Group Ring Module

Hidetsune Kobayashi, L. Chen, H. Murao

March 17, 2025

Abstract

The theory of groups, rings and modules is developed to a great depth. Group theory results include Zassenhaus's theorem and the Jordan-Hoelder theorem. The ring theory development includes ideals, quotient rings and the Chinese remainder theorem. The module development includes the Nakayama lemma, exact sequences and Tensor products.

Contents

theory *Algebra1*
imports *Main HOL−Library.FuncSet*
begin

Chapter 1

Preliminaries

Some of the lemmas of this section are proved in src/HOL/Integ of Isabelle version 2003.

1.1 Lemmas for logical manipulation

lemma *True-then*:$True \longrightarrow P \Longrightarrow P$
⟨*proof*⟩

lemma *ex-conjI*:$[\![P\ c;\ Q\ c]\!] \Longrightarrow \exists c.\ P\ c \wedge Q\ c$
⟨*proof*⟩

lemma *forall-spec*:$[\![\ \forall b.\ P\ b \longrightarrow Q\ b;\ P\ a]\!] \Longrightarrow Q\ a$
⟨*proof*⟩

lemma *a-b-exchange*:$[\![a;\ a = b]\!] \Longrightarrow b$
⟨*proof*⟩

lemma *eq-prop*:$[\![\ P;\ P = Q]\!] \Longrightarrow Q$
⟨*proof*⟩

lemma *forball-contra*:$[\![\forall y{\in}A.\ P\ x\ y \longrightarrow \neg\ Q\ y;\ \forall y{\in}A.\ Q\ y \vee R\ y]\!] \Longrightarrow$
$\forall y{\in}A.\ (\neg\ P\ x\ y) \vee R\ y$
⟨*proof*⟩

lemma *forball-contra1*:$[\![\forall y{\in}A.\ P\ x\ y \longrightarrow Q\ y;\ \forall y{\in}A.\ \neg\ Q\ y]\!] \Longrightarrow \forall y{\in}A.\ \neg\ P\ x\ y$
⟨*proof*⟩

1.2 Natural numbers and Integers

Elementary properties of natural numbers and integers

lemma *nat-nonzero-pos*:$(a{::}nat) \neq 0 \Longrightarrow 0 < a$
⟨*proof*⟩

lemma *add-both*:$(a::nat) = b \Longrightarrow a + c = b + c$
⟨*proof*⟩

lemma *add-bothl*:$a = b \Longrightarrow c + a = c + b$
⟨*proof*⟩

lemma *diff-Suc*:$(n::nat) \leq m \Longrightarrow m - n + Suc\ 0 = Suc\ m - n$
⟨*proof*⟩

lemma *le-convert*:$[\![a = b;\ a \leq c]\!] \Longrightarrow b \leq c$
⟨*proof*⟩

lemma *ge-convert*:$[\![a = b;\ c \leq a]\!] \Longrightarrow c \leq b$
⟨*proof*⟩

lemma *less-convert*:$[\![\ a = b;\ c < b\ ]\!] \Longrightarrow c < a$
⟨*proof*⟩

lemma *ineq-conv1*:$[\![a = b;\ a < c]\!] \Longrightarrow b < c$
⟨*proof*⟩

lemma *diff-Suc-pos*:$0 < a - Suc\ 0 \Longrightarrow\ 0 < a$
⟨*proof*⟩

lemma *minus-SucSuc*:$a - Suc\ (Suc\ 0) = a - Suc\ 0 - Suc\ 0$
⟨*proof*⟩

lemma *Suc-Suc-Tr*:$Suc\ (Suc\ 0) \leq n \Longrightarrow Suc\ (n - Suc\ (Suc\ 0)) = n - Suc\ 0$
⟨*proof*⟩

lemma *Suc-Suc-less*:$Suc\ 0 < a \Longrightarrow Suc\ (a - Suc\ (Suc\ 0)) < a$
⟨*proof*⟩

lemma *diff-zero-eq*:$n = (0::nat) \Longrightarrow m = m - n$
⟨*proof*⟩

lemma *Suc-less-le*:$x < Suc\ n \Longrightarrow x \leq n$
⟨*proof*⟩

lemma *less-le-diff*:$x < n \Longrightarrow x \leq n - Suc\ 0$
⟨*proof*⟩

lemma *le-pre-le*:$x \leq n - Suc\ 0 \Longrightarrow x \leq n$
⟨*proof*⟩

lemma *nat-not-less*:$\neg\ (m::nat) < n \Longrightarrow n \leq m$
⟨*proof*⟩

lemma *less-neq*:$n < (m::nat) \Longrightarrow n \neq m$
⟨*proof*⟩

lemma *less-le-diff1*:$n \neq 0 \Longrightarrow ((m::nat) < n) = (m \leq (n - Suc\ 0))$
⟨*proof*⟩

lemma *nat-not-less1*:$n \neq 0 \Longrightarrow (\neg\ (m::nat) < n) = (\neg\ m \leq (n - Suc\ 0))$
⟨*proof*⟩

lemma *nat-eq-le*:$m = (n::nat) \Longrightarrow m \leq n$
⟨*proof*⟩

1.2.1 Integers

lemma *non-zero-int*: $(n::int) \neq 0 \Longrightarrow 0 < n \vee n < 0$
⟨*proof*⟩

lemma *zgt-0-zge-1*:$(0::int) < z \Longrightarrow 1 \leq z$
⟨*proof*⟩

lemma *not-zle*:$(\neg\ (n::int) \leq m) = \ (m < n)$
⟨*proof*⟩

lemma *not-zless*:$(\neg\ (n::int) < m) = (m \leq n)$
⟨*proof*⟩

lemma *zle-imp-zless-or-eq*:$(n::int) \leq m \Longrightarrow n < m \vee n = m$
⟨*proof*⟩

lemma *zminus-zadd-cancel*: $-\ z + (z + w) = (w::int)$
⟨*proof*⟩

lemma *int-neq-iff*:$((w::int) \neq z) = (w < z) \vee (z < w)$
⟨*proof*⟩

lemma *zless-imp-zle*:$(z::int) < z' \Longrightarrow z \leq z'$
⟨*proof*⟩

lemma *zdiff*:$z - (w::int) = z + (-\ w)$
⟨*proof*⟩

lemma *zle-zless-trans*:$[\![\ (i::int) \leq j; j < k]\!] \Longrightarrow i < k$
⟨*proof*⟩

lemma *zless-zle-trans*:$[\![\ (i::int) < j; j \leq k]\!] \Longrightarrow i < k$
⟨*proof*⟩

lemma *zless-neq*:$(i::int) < j \Longrightarrow i \neq j$
⟨*proof*⟩

lemma *int-mult-mono*:$[\![\ i < j;\ (0::int) < k\ ]\!] \Longrightarrow k * i < k * j$
⟨*proof*⟩

lemma *int-mult-le*:$[\![i \leq j;\ (0::int) \leq k]\!] \Longrightarrow k * i \leq k * j$
⟨*proof*⟩

lemma *int-mult-le1*:$[\![i \leq j;\ (0::int) \leq k]\!] \Longrightarrow i * k \leq j * k$
⟨*proof*⟩

lemma *zmult-zminus-right*:$(w::int) * (-\ z) = -\ (w * z)$
⟨*proof*⟩

lemma *zmult-zle-mono1-neg*:$[\![(i::int) \leq j;\ k \leq 0]\!] \Longrightarrow j * k \leq i * k$
⟨*proof*⟩

lemma *zmult-zless-mono-neg*:$[\![(i::int) < j;\ k < 0]\!] \Longrightarrow j * k < i * k$
⟨*proof*⟩

lemma *zmult-neg-neg*:$[\![i < (0::int);\ j < 0\ ]\!] \Longrightarrow 0 < i * j$
⟨*proof*⟩

lemma *zmult-pos-pos*:$[\![(0::int) < i;\ 0 < j]\!] \Longrightarrow 0 < i * j$
⟨*proof*⟩

lemma *zmult-pos-neg*:$[\![(0::int) < i;\ j < 0]\!] \Longrightarrow i * j < 0$
⟨*proof*⟩

lemma *zmult-neg-pos*:$[\![i < (0::int);\ 0 < j]\!] \Longrightarrow i * j < 0$
⟨*proof*⟩

lemma *zle*:$((z::int) \leq w) = (\neg\ (w < z))$
⟨*proof*⟩

lemma *times-1-both*:$[\![(0::int) < z;\ z * z' = 1]\!] \Longrightarrow z = 1 \wedge z' = 1$
⟨*proof*⟩

lemma *zminus-minus*:$i - - (j::int) = i + j$
⟨*proof*⟩

lemma *zminus-minus-pos*:$(n::int) < 0 \Longrightarrow 0 < - n$
⟨*proof*⟩

lemma *zadd-zle-mono*:$[\![w' \leq w;\ z' \leq (z::int)]\!] \Longrightarrow w' + z' \leq w + z$
⟨*proof*⟩

lemma *zmult-zle-mono*:$[\![i \leq (j::int);\ 0 < k]\!] \Longrightarrow k * i \leq\ k * j$
⟨*proof*⟩

lemma *zmult-zle-mono-r*:$[\![i \leq (j::int);\ 0 < k]\!] \Longrightarrow i * k \leq j * k$
⟨*proof*⟩

lemma *pos-zmult-pos*:$[\![\ 0 \leq (a::int);\ 0 < (b::int)]\!] \Longrightarrow a \leq a * b$
⟨*proof*⟩

lemma *pos-mult-l-gt*:$[\![(0::int) < w;\ i \leq j;\ 0 \leq i]\!] \Longrightarrow i \leq w * j$
⟨*proof*⟩

lemma *pos-mult-r-gt*:$[\![(0::int) < w;\ i \leq j;\ 0 \leq i]\!] \Longrightarrow i \leq j * w$
⟨*proof*⟩

lemma *mult-pos-iff*:$[\![(0::int) < i;\ 0 \leq i * j\ ]\!] \Longrightarrow 0 \leq j$
⟨*proof*⟩

lemma *zmult-eq*:$[\![(0::int) < w;\ z = z']\!] \Longrightarrow w * z = w * z'$
⟨*proof*⟩

lemma *zmult-eq-r*:$[\![(0::int) < w;\ z = z']\!] \Longrightarrow z * w = z' * w$
⟨*proof*⟩

lemma *zdiv-eq-l*:$[\![(0::int) < w;\ z * w = z' * w\ ]\!] \Longrightarrow z = z'$
⟨*proof*⟩

lemma *zdiv-eq-r*:$[\![(0::int) < w;\ w * z = w * z'\ ]\!] \Longrightarrow z = z'$
⟨*proof*⟩

lemma *int-nat-minus*:$0 < (n::int) \Longrightarrow nat\ (n - 1) = (nat\ n) - 1$
⟨*proof*⟩

lemma *int-nat-add*:$[\![0 < (n::int);\ 0 < (m::int)]\!] \Longrightarrow (nat\ (n - 1)) + (nat\ (m - 1)) + (Suc\ 0) = nat\ (n + m - 1)$
⟨*proof*⟩

lemma *int-equation*:$(x::int) = y + z \Longrightarrow x - y = z$
⟨*proof*⟩

lemma *int-pos-mult-monor*:$[\![\ 0 < (n::int);\ 0 \leq n * m\ ]\!] \Longrightarrow 0 \leq m$
⟨*proof*⟩

lemma *int-pos-mult-monol*:$[\![\ 0 < (m::int);\ 0 \leq n * m\ ]\!] \Longrightarrow 0 \leq n$
⟨*proof*⟩

lemma *zdiv-positive*:$[\![(0::int) \leq a;\ 0 < b]\!] \Longrightarrow 0 \leq a\ div\ b$
⟨*proof*⟩

lemma *zdiv-pos-mono-r*:$[\![\ (0::int) < w;\ w * z \leq w * z']\!] \Longrightarrow z \leq z'$
⟨*proof*⟩

lemma *zdiv-pos-mono-l*:$[\![(0::int) < w;\ z * w \leq z' * w]\!] \Longrightarrow z \leq z'$
⟨*proof*⟩

lemma *zdiv-pos-pos-l*:$[\![(0::int) < w;\ 0 \leq z * w]\!] \Longrightarrow 0 \leq z$
⟨*proof*⟩

1.3 Sets

1.3.1 A short notes for proof steps

1.3.2 Sets

lemma *inEx*:$x \in A \Longrightarrow \exists\, y{\in}A.\ y = x$
⟨*proof*⟩

lemma *inEx-rev*: $\exists\, y{\in}A.\ y = x \Longrightarrow x \in A$
⟨*proof*⟩

lemma *nonempty-ex*:$A \neq \{\} \Longrightarrow \exists\, x.\ x \in A$
⟨*proof*⟩

lemma *ex-nonempty*:$\exists\, x.\ x \in A \Longrightarrow A \neq \{\}$
⟨*proof*⟩

lemma *not-eq-outside*:$a \notin A \Longrightarrow \forall\, b{\in}A.\ b \neq a$
⟨*proof*⟩

lemma *ex-nonempty-set*:$\exists\, a.\ P\ a \Longrightarrow \{x.\ P\ x\} \neq \{\}$
⟨*proof*⟩

lemma *nonempty*: $x \in A \Longrightarrow A \neq \{\}$
⟨*proof*⟩

lemma *subset-self*:$A \subseteq A$
⟨*proof*⟩

lemma *conditional-subset*:$\{x{\in}A.\ P\ x\} \subseteq A$
⟨*proof*⟩

lemma *bsubsetTr*:$\{x.\ x \in A \wedge P\ x\} \subseteq A$
⟨*proof*⟩

lemma *sets-not-eq*:$[\![A \neq B;\ B \subseteq A]\!] \Longrightarrow \exists\, a{\in}A.\ a \notin B$
⟨*proof*⟩

lemma *diff-nonempty*:$[\![A \neq B;\ B \subseteq A]\!] \Longrightarrow A - B \neq \{\}$
⟨*proof*⟩

lemma *sub-which1*:$[\![A \subseteq B \vee B \subseteq A;\ x \in A;\ x \notin B]\!] \Longrightarrow B \subseteq A$
⟨*proof*⟩

lemma *sub-which2*:$[\![A \subseteq B \vee B \subseteq A;\ x \notin A;\ x \in B]\!] \Longrightarrow A \subseteq B$
⟨*proof*⟩

lemma *nonempty-int*: $A \cap B \neq \{\} \Longrightarrow \exists\, x.\ x \in A \cap B$
⟨*proof*⟩

lemma *no-meet1*:$A \cap B = \{\} \Longrightarrow \forall\, a \in A.\ a \notin B$
⟨*proof*⟩

lemma *no-meet2*:$A \cap B = \{\} \Longrightarrow \forall\, a \in B.\ a \notin A$
⟨*proof*⟩

lemma *elem-some*:$x \in A \Longrightarrow \exists\, y{\in}A.\ x = y$
⟨*proof*⟩

lemma *singleton-sub*:$a \in A \Longrightarrow \{a\} \subseteq A$
⟨*proof*⟩

lemma *eq-elem-in*: $[\![\ a \in A;\ a = b\ ]\!] \Longrightarrow b \in A$
⟨*proof*⟩

lemma *eq-set-inc*: $[\![\ a \in A;\ A = B\ ]\!] \Longrightarrow a \in B$
⟨*proof*⟩

lemma *eq-set-not-inc*:$[\![a \notin A;\ A = B\ ]\!] \Longrightarrow a \notin B$
⟨*proof*⟩

lemma *int-subsets*: $[\![\ A1 \subseteq A;\ B1 \subseteq B\ ]\!] \Longrightarrow A1 \cap B1 \subseteq A \cap B$
⟨*proof*⟩

lemma *inter-mono*:$A \subseteq B \Longrightarrow A \cap C \subseteq B \cap C$
⟨*proof*⟩

lemma *sub-Un1*:$B \subseteq\ B \cup C$
⟨*proof*⟩

lemma *sub-Un2*:$C \subseteq\ B \cup C$
⟨*proof*⟩

lemma *subset-contr*:$[\![\ A \subset B;\ B \subseteq A\ ]\!] \Longrightarrow False$
⟨*proof*⟩

lemma *psubset-contr*:$[\![\ A \subset B;\ B \subset A\ ]\!] \Longrightarrow False$
⟨*proof*⟩

lemma *eqsets-sub*:$A = B \Longrightarrow A \subseteq B$

⟨*proof*⟩

lemma *not-subseteq*: $\neg\ A \subseteq B \Longrightarrow \exists\, a \in A.\ a \notin B$
⟨*proof*⟩

lemma *in-un1*:$[\![\ x \in A \cup B;\ x \notin B\ ]\!] \Longrightarrow x \in A$
⟨*proof*⟩

lemma *proper-subset*:$[\![A \subseteq B;\ x \notin A;\ x \in B]\!] \Longrightarrow A \neq B$
⟨*proof*⟩

lemma *in-un2*:$[\![\ x \in A \cup B;\ x \notin A\ ]\!] \Longrightarrow x \in B$
⟨*proof*⟩

lemma *diff-disj*:$x \notin A \Longrightarrow A - \{x\} = A$
⟨*proof*⟩

lemma *in-diff*:$[\![x \neq a;\ x \in A]\!] \Longrightarrow x \in A - \{a\}$
⟨*proof*⟩

lemma *in-diff1*:$x \in A - \{a\} \Longrightarrow x \neq a$
⟨*proof*⟩

lemma *sub-inserted1*:$[\![Y \subseteq \mathit{insert}\ a\ X;\ \neg\ Y \subseteq X]\!] \Longrightarrow a \notin X \wedge a \in Y$
⟨*proof*⟩

lemma *sub-inserted2*:$[\![Y \subseteq \mathit{insert}\ a\ X;\ \neg\ Y \subseteq X]\!] \Longrightarrow Y = (Y - \{a\}) \cup \{a\}$
⟨*proof*⟩

lemma *insert-sub*:$[\![\ A \subseteq B;\ a \in B]\!] \Longrightarrow (\mathit{insert}\ a\ A) \subseteq B$
⟨*proof*⟩

lemma *insert-diff*:$A \subseteq (\mathit{insert}\ b\ B) \Longrightarrow A - \{b\} \subseteq B$
⟨*proof*⟩

lemma *insert-inc1*:$A \subseteq \mathit{insert}\ a\ A$
⟨*proof*⟩

lemma *insert-inc2*:$a \in \mathit{insert}\ a\ A$
⟨*proof*⟩

lemma *nonempty-some*:$A \neq \{\} \Longrightarrow (\mathit{SOME}\ x.\ x \in A) \in A$
⟨*proof*⟩

lemma *mem-family-sub-Un*:$A \in C \Longrightarrow A \subseteq \bigcup\ C$
⟨*proof*⟩

lemma *sub-Union*:$\exists\, X{\in}C.\ A \subseteq X \Longrightarrow A \subseteq \bigcup\ C$
⟨*proof*⟩

lemma *family-subset-Un-sub*:$\forall\, A{\in}C.\ A \subseteq B \Longrightarrow \bigcup\ C \subseteq B$
⟨*proof*⟩

lemma *in-set-with-P*:$P\ x \Longrightarrow x \in \{y.\ P\ y\}$
⟨*proof*⟩

lemma *sub-single*:$[\![A \neq \{\};\ A \subseteq \{a\}]\!] \Longrightarrow A = \{a\}$
⟨*proof*⟩

lemma *not-sub-single*:$[\![A \neq \{\};\ A \neq \{a\}]\!] \Longrightarrow \neg\ A \subseteq \{a\}$
⟨*proof*⟩

lemma *not-sub*:$\neg\ A \subseteq B \Longrightarrow \exists\, a.\ a{\in}A \wedge a \notin B$
⟨*proof*⟩

1.4 Functions

definition
cmp :: $['b \Rightarrow 'c,\ 'a \Rightarrow 'b] \Rightarrow ('a \Rightarrow 'c)$ **where**
$cmp\ g\ f = (\lambda x.\ g\ (f\ x))$

definition
idmap :: $'a\ set \Rightarrow ('a \Rightarrow 'a)$ **where**
$idmap\ A = (\lambda x{\in}A.\ x)$

definition
constmap :: $['a\ set,\ 'b\ set] \Rightarrow ('a \Rightarrow 'b)$ **where**
$constmap\ A\ B = (\lambda x{\in}A.\ SOME\ y.\ y \in B)$

definition
invfun :: $['a\ set,\ 'b\ set,\ 'a \Rightarrow 'b] \Rightarrow ('b \Rightarrow 'a)$ **where**
$invfun\ A\ B\ (f :: 'a \Rightarrow 'b) = (\lambda y{\in}B.(SOME\ x.\ (x \in A \wedge f\ x = y)))$

abbreviation
INVFUN :: $['a \Rightarrow 'b,\ 'b\ set,\ 'a\ set] \Rightarrow ('b \Rightarrow 'a)$ $(\langle(3\text{-}^{-1}\text{-},\text{-})\rangle\ [82,82,83]82)$ **where**
${f^{-1}}_{B,A} == invfun\ A\ B\ f$

lemma *eq-fun*:$[\![\ f \in A \to B;\ f = g\ ]\!] \Longrightarrow g \in A \to B$
⟨*proof*⟩

lemma *eq-fun-eq-val*: $f = g \Longrightarrow f\ x = g\ x$
⟨*proof*⟩

lemma *eq-elems-eq-val*:$x = y \Longrightarrow f\ x = f\ y$
⟨*proof*⟩

lemma *cmp-fun*:$[\![f \in A \to B;\ g \in B \to C\ ]\!] \Longrightarrow cmp\ g\ f \in A \to C$
⟨*proof*⟩

lemma *cmp-fun-image*:$[\![f \in A \to B;\ g \in B \to C\ ]\!] \Longrightarrow$
$(cmp\ g\ f)\ `\ A = \ g\ `\ (f\ `\ A)$
⟨*proof*⟩

lemma *cmp-fun-sub-image*:$[\![f \in A \to B;\ g \in B \to C;\ A1 \subseteq A]\!] \Longrightarrow$
$(cmp\ g\ f)\ `\ A1 = \ g\ `\ (f\ `\ A1)$
⟨*proof*⟩

lemma *restrict-fun-eq*:$\forall\, x{\in}A.\ f\ x = g\ x \Longrightarrow (\lambda x{\in}A.\ f\ x) = (\lambda x{\in}A.\ g\ x)$
⟨*proof*⟩

lemma *funcset-mem*: $[\![f \in A \to B;\ x \in A]\!] \Longrightarrow f\ x \in B$
⟨*proof*⟩

lemma *img-subset*:$f \in A \to B \Longrightarrow f\ `\ A \subseteq B$
⟨*proof*⟩

lemma *funcset-mem1*:$[\![\forall\, l{\in}A.\ f\ l \in B;\ x \in A]\!] \Longrightarrow f\ x \in B$
⟨*proof*⟩

lemma *func-to-img*:$f \in A \to B \Longrightarrow f \in A \to f\ `\ A$
⟨*proof*⟩

lemma *funcset-eq*:$[\![\ f \in extensional\ A;\ g \in extensional\ A;\ \forall\, x{\in}A.\ f\ x = g\ x\ ]\!] \Longrightarrow$
$f = g$
⟨*proof*⟩

lemma *eq-funcs*:$[\![f \in A \to B;\ g \in A \to B;\ f = g;\ x \in A]\!] \Longrightarrow f\ x = g\ x$
⟨*proof*⟩

lemma *restriction-of-domain*:$[\![\ f \in A \to B;\ A1 \subseteq A\ ]\!] \Longrightarrow$
$restrict\ f\ A1 \in A1 \to B$
⟨*proof*⟩

lemma *restrict-restrict*:$[\![\ restrict\ f\ A \in A \to B;\ A1 \subseteq A\ ]\!] \Longrightarrow$
$restrict\ (restrict\ f\ A)\ A1 = restrict\ f\ A1$
⟨*proof*⟩

lemma *restr-restr-eq*:$[\![\ restrict\ f\ A \in A \to B;\ restrict\ f\ A = restrict\ g\ A;$
$A1 \subseteq A\ ]\!] \Longrightarrow restrict\ f\ A1 = restrict\ g\ A1$
⟨*proof*⟩

lemma *funcTr*:$[\![\ f \in A \to B;\ g \in A \to B;\ f = g;\ a \in A]\!] \Longrightarrow f\ a = g\ a$
⟨*proof*⟩

lemma *funcTr1*:$[\![f = g;\ a \in A]\!] \Longrightarrow f\ a = g\ a$
⟨*proof*⟩

lemma *restrictfun-im*:⟦ (*restrict f A*) ∈ *A* → *B*; *A1* ⊆ *A* ⟧ ⟹
(*restrict f A*) ‘ *A1* = *f* ‘ *A1*
⟨*proof*⟩

lemma *mem-in-image*:⟦ *f* ∈ *A* → *B*; *a* ∈ *A*⟧ ⟹ *f a* ∈ *f* ‘ *A*
⟨*proof*⟩

lemma *mem-in-image1*:⟦ ∀ *l*∈*A*. *f l* ∈ *B*; *a* ∈ *A*⟧ ⟹ *f a* ∈ *f* ‘ *A*
⟨*proof*⟩

lemma *mem-in-image2*:*a* ∈ *A* ⟹ *f a* ∈ *f* ‘ *A*
⟨*proof*⟩

lemma *mem-in-image3*:*b* ∈ *f* ‘ *A* ⟹ ∃ *a* ∈ *A*. *b* = *f a*
⟨*proof*⟩

lemma *elem-in-image2*: ⟦ *f* ∈ *A* → *B*; *A1* ⊆ *A*; *x* ∈ *A1*⟧ ⟹ *f x* ∈ *f*‘ *A1*
⟨*proof*⟩

lemma *funcs-nonempty*:⟦ *A* ≠ {}; *B* ≠ {} ⟧ ⟹ (*A* → *B*) ≠ {}
⟨*proof*⟩

lemma *idmap-funcs*: *idmap A* ∈ *A* → *A*
⟨*proof*⟩

lemma *l-idmap-comp*: ⟦*f* ∈ *extensional A*; *f* ∈ *A* → *B*⟧ ⟹
compose A (*idmap B*) *f* = *f*
⟨*proof*⟩

lemma *r-idmap-comp*:⟦*f* ∈ *extensional A*; *f* ∈ *A* → *B*⟧ ⟹
compose A f (*idmap A*) = *f*
⟨*proof*⟩

lemma *extend-fun*: ⟦ *f* ∈ *A* → *B*; *B* ⊆ *B1* ⟧ ⟹ *f* ∈ *A* → *B1*
⟨*proof*⟩

lemma *restrict-fun*: ⟦ *f* ∈ *A* → *B*; *A1* ⊆ *A* ⟧ ⟹ *restrict f A1* ∈ *A1* → *B*
⟨*proof*⟩

lemma *set-of-hom*: ∀ *x* ∈ *A*. *f x* ∈ *B* ⟹ *restrict f A* ∈ *A* → *B*
⟨*proof*⟩

lemma *composition* : ⟦ *f* ∈ *A* → *B*; *g* ∈ *B* → *C*⟧ ⟹ (*compose A g f*) ∈ *A* → *C*
⟨*proof*⟩

lemma *comp-assoc*:⟦*f* ∈ *A* → *B*; *g* ∈ *B* → *C*; *h* ∈ *C* → *D* ⟧ ⟹
compose A h (*compose A g f*) = *compose A* (*compose B h g*) *f*

⟨*proof*⟩

lemma *restrictfun-inj*: ⟦ *inj-on f A*; *A1* ⊆ *A* ⟧ ⟹ *inj-on* (*restrict f A1*) *A1*
⟨*proof*⟩

lemma *restrict-inj*:⟦*inj-on f A*; *A1* ⊆ *A*⟧ ⟹ *inj-on f A1*
⟨*proof*⟩

lemma *injective*:⟦ *inj-on f A*; *x* ∈ *A*; *y* ∈ *A*; *x* ≠ *y* ⟧ ⟹ *f x* ≠ *f y*
⟨*proof*⟩

lemma *injective-iff*:⟦ *inj-on f A*; *x* ∈ *A*; *y* ∈ *A*⟧ ⟹
(*x* = *y*) = (*f x* = *f y*)
⟨*proof*⟩

lemma *injfun-elim-image*:⟦*f* ∈ *A* → *B*; *inj-on f A*; *x* ∈ *A*⟧ ⟹
f ‘ (*A* − {*x*}) = (*f* ‘ *A*) − {*f x*}
⟨*proof*⟩

lemma *cmp-inj*:⟦*f* ∈ *A* → *B*; *g* ∈ *B* → *C*; *inj-on f A*; *inj-on g B* ⟧ ⟹
inj-on (*cmp g f*) *A*
⟨*proof*⟩

lemma *cmp-assoc*:⟦*f* ∈ *A* → *B*; *g* ∈ *B* → *C*; *h* ∈ *C* → *D*; *x* ∈ *A*⟧ ⟹
(*cmp h* (*cmp g f*)) *x* = (*cmp* (*cmp h g*) *f*) *x*
⟨*proof*⟩

lemma *bivar-fun*: ⟦ *f* ∈ *A* → (*B* → *C*); *a* ∈ *A* ⟧ ⟹ *f a* ∈ *B* → *C*
⟨*proof*⟩

lemma *bivar-fun-mem*: ⟦ *f* ∈ *A* → (*B* → *C*); *a* ∈ *A*; *b* ∈ *B* ⟧ ⟹ *f a b* ∈ *C*
⟨*proof*⟩

lemma *bivar-func-eq*:⟦∀ *a*∈*A*. ∀ *b*∈*B*. *f a b* = *g a b* ⟧ ⟹
(λ*x*∈*A*. λ*y*∈*B*. *f x y*) = (λ*x*∈*A*. λ*y*∈*B*. *g x y*)
⟨*proof*⟩

lemma *set-image*: ⟦ *f* ∈ *A* → *B*; *A1* ⊆ *A*; *A2* ⊆ *A* ⟧ ⟹
f‘(*A1* ∩ *A2*) ⊆ (*f*‘ *A1*) ∩ (*f*‘ *A2*)
⟨*proof*⟩

lemma *image-sub*: ⟦*f* ∈ *A* → *B*; *A1* ⊆ *A* ⟧ ⟹ (*f*‘*A1*) ⊆ *B*
⟨*proof*⟩

lemma *image-sub0*: *f* ∈ *A* → *B* ⟹ (*f*‘*A*) ⊆ *B*
⟨*proof*⟩

lemma *image-nonempty*:⟦*f* ∈ *A* → *B*; *A1* ⊆ *A*; *A1* ≠ {} ⟧ ⟹ *f* ‘ *A1* ≠ {}
⟨*proof*⟩

lemma *im-set-mono*: $[\![f \in A \to B;\ A1 \subseteq A2;\ A2 \subseteq A\ ]\!] \Longrightarrow (f\ `\ A1) \subseteq (f\ `\ A2)$
⟨*proof*⟩

lemma *im-set-un*:$[\![f \in A \to B;\ A1 \subseteq A;\ A2 \subseteq A\ ]\!] \Longrightarrow$
$f`(A1 \cup A2) = (f`A1) \cup (f`A2)$
⟨*proof*⟩

lemma *im-set-un1*:$[\![\forall l \in A.\ f\ l \in B;\ A = A1 \cup A2]\!] \Longrightarrow$
$f\ `(A1 \cup A2) = f\ `(A1) \cup f\ `(A2)$
⟨*proof*⟩

lemma *im-set-un2*:$A = A1 \cup A2 \Longrightarrow\ f\ `A = f\ `(A1) \cup f\ `(A2)$
⟨*proof*⟩

definition
invim::$['a \Rightarrow 'b,\ 'a\ set,\ 'b\ set] \Rightarrow 'a\ set$ **where**
$invim\ f\ A\ B = \{x.\ x \in A \wedge f\ x \in B\}$

lemma *invim*: $[\![f{:}A \to B;\ B1 \subseteq B\ ]\!] \Longrightarrow invim\ f\ A\ B1 \subseteq A$
⟨*proof*⟩

lemma *setim-cmpfn*: $[\![f{:}A \to B;\ g{:}B \to C;\ A1 \subseteq A\ ]\!] \Longrightarrow$
$(compose\ A\ g\ f)`\ A1 = g`(f`\ A1)$
⟨*proof*⟩

definition
surj-to :: $['a \Rightarrow 'b,\ 'a\ set,\ 'b\ set] \Rightarrow bool$ **where**
$surj\text{-}to\ f\ A\ B \longleftrightarrow f`A = B$

lemma *surj-to-test*:$[\![f \in A \to B;\ \forall b \in B.\ \exists a \in A.\ f\ a = b\ ]\!] \Longrightarrow$
$surj\text{-}to\ f\ A\ B$
⟨*proof*⟩

lemma *surj-to-image*:$f \in A \to B \Longrightarrow surj\text{-}to\ f\ A\ (f\ `\ A)$
⟨*proof*⟩

lemma *surj-to-el*:$[\![f \in A \to B;\ surj\text{-}to\ f\ A\ B\ ]\!] \Longrightarrow \forall b \in B.\ \exists a \in A.\ f\ a = b$
⟨*proof*⟩

lemma *surj-to-el1*:$[\![f \in A \to B;\ surj\text{-}to\ f\ A\ B;\ b \in B]\!] \Longrightarrow \exists a \in A.\ f\ a = b$
⟨*proof*⟩

lemma *surj-to-el2*:$[\![surj\text{-}to\ f\ A\ B;\ b \in B]\!] \Longrightarrow \exists a \in A.\ f\ a = b$
⟨*proof*⟩

lemma *compose-surj*: $[\![f{:}A \to B;\ surj\text{-}to\ f\ A\ B;\ g : B \to C;\ surj\text{-}to\ g\ B\ C\ ]\!]$
$\Longrightarrow surj\text{-}to\ (compose\ A\ g\ f)\ A\ C$
⟨*proof*⟩

lemma *cmp-surj*: $[\![f{:}A \to B;\ surj\text{-}to\ f\ A\ B;\ g : B \to C;\ surj\text{-}to\ g\ B\ C\ ]\!]$
$\Longrightarrow surj\text{-}to\ (cmp\ g\ f)\ A\ C$
⟨*proof*⟩

lemma *inj-onTr0*:$[\![\ f \in A \to B;\ x \in A;\ y \in A;\ inj\text{-}on\ f\ A;\ f\ x = f\ y]\!] \Longrightarrow x = y$
⟨*proof*⟩

lemma *inj-onTr1*:$[\![inj\text{-}on\ f\ A;\ x \in A;\ y \in A;\ f\ x = f\ y]\!] \Longrightarrow x = y$
⟨*proof*⟩

lemma *inj-onTr2*:$[\![inj\text{-}on\ f\ A;\ x \in A;\ y \in A;\ f\ x \neq f\ y]\!] \Longrightarrow x \neq y$
⟨*proof*⟩

lemma *comp-inj*: $[\![\ f \in A \to B;\ inj\text{-}on\ f\ A;\ g \in B \to C;\ inj\text{-}on\ g\ B\ ]\!]$
$\Longrightarrow inj\text{-}on\ (compose\ A\ g\ f)\ A$
⟨*proof*⟩

lemma *cmp-inj-1*: $[\![\ f \in A \to B;\ inj\text{-}on\ f\ A;\ g \in B \to C;\ inj\text{-}on\ g\ B\ ]\!]$
$\Longrightarrow inj\text{-}on\ (cmp\ g\ f)\ A$
⟨*proof*⟩

lemma *cmp-inj-2*: $[\![\forall\, l{\in}A.\ f\ l \in B;\ inj\text{-}on\ f\ A;\ \forall\, k{\in}B.\ g\ k \in C;\ inj\text{-}on\ g\ B\ ]\!]$
$\Longrightarrow inj\text{-}on\ (cmp\ g\ f)\ A$
⟨*proof*⟩

lemma *invfun-mem*:$[\![\ f \in A \to B;\ inj\text{-}on\ f\ A;\ surj\text{-}to\ f\ A\ B;\ b \in B\ ]\!]$
$\Longrightarrow\ (invfun\ A\ B\ f)\ b \in A$
⟨*proof*⟩

lemma *inv-func*:$[\![\ f \in A \to B;\ inj\text{-}on\ f\ A;\ surj\text{-}to\ f\ A\ B]\!]$
$\Longrightarrow\ (invfun\ A\ B\ f) \in B \to A$
⟨*proof*⟩

lemma *invfun-r*:$[\![\ f{\in}A \to B;\ inj\text{-}on\ f\ A;\ surj\text{-}to\ f\ A\ B;\ b \in B\ ]\!]$
$\Longrightarrow f\ ((invfun\ A\ B\ f)\ b) = b$
⟨*proof*⟩

lemma *invfun-l*:$[\![f \in A \to B;\ inj\text{-}on\ f\ A;\ surj\text{-}to\ f\ A\ B;\ a \in A]\!]$
$\Longrightarrow (invfun\ A\ B\ f)\ (f\ a) = a$
⟨*proof*⟩

lemma *invfun-inj*:$[\![f \in A \to B;\ inj\text{-}on\ f\ A;\ surj\text{-}to\ f\ A\ B]\!]$
$\Longrightarrow\ inj\text{-}on\ (invfun\ A\ B\ f)\ B$
⟨*proof*⟩

lemma *invfun-surj*:$[\![f \in A \to B;\ inj\text{-}on\ f\ A;\ surj\text{-}to\ f\ A\ B]\!]$

$\Longrightarrow$ *surj-to* (*invfun A B f*) *B A*

⟨*proof*⟩

definition
bij-to :: [*'a* ⇒ *'b*, *'a set*, *'b set*] ⇒ *bool* **where**
bij-to f A B ⟷ *surj-to f A B* ∧ *inj-on f A*

lemma *idmap-bij*:*bij-to* (*idmap A*) *A A*
⟨*proof*⟩

lemma *bij-invfun*:⟦*f* ∈ *A* → *B*; *bij-to f A B*⟧ ⟹
bij-to (*invfun A B f*) *B A*
⟨*proof*⟩

lemma *l-inv-invfun*:⟦ *f* ∈ *A* → *B*; *inj-on f A*; *surj-to f A B*⟧
⟹ *compose A* (*invfun A B f*) *f* = *idmap A*
⟨*proof*⟩

lemma *invfun-mem1*:⟦*f* ∈ *A* → *B*; *bij-to f A B*; *b* ∈ *B*⟧ ⟹
(*invfun A B f*) *b* ∈ *A*
⟨*proof*⟩

lemma *invfun-r1*:⟦ *f*∈*A* → *B*; *bij-to f A B*; *b* ∈ *B* ⟧
⟹ *f* ((*invfun A B f*) *b*) = *b*
⟨*proof*⟩

lemma *invfun-l1*:⟦*f* ∈ *A* → *B*; *bij-to f A B*; *a* ∈ *A*⟧
⟹ (*invfun A B f*) (*f a*) = *a*
⟨*proof*⟩

lemma *compos-invfun-r*:⟦*f* ∈ *A* → *B*; *bij-to f A B*; *g* ∈ *A* → *C*; *h* ∈ *B* → *C*;
g ∈ *extensional A*; *compose B g* (*invfun A B f*) = *h*⟧ ⟹
g = *compose A h f*
⟨*proof*⟩

lemma *compos-invfun-l*:⟦*f* ∈ *A* → *B*; *bij-to f A B*; *g* ∈ *C* → *B*; *h* ∈ *C* → *A*;
compose C (*invfun A B f*) *g* = *h*; *g* ∈ *extensional C* ⟧ ⟹
g = *compose C f h*
⟨*proof*⟩

lemma *invfun-set*:⟦*f* ∈ *A* → *B*; *bij-to f A B*; *C* ⊆ *B*⟧ ⟹
f ‘ ((*invfun A B f*)‘ *C*) = *C*
⟨*proof*⟩

lemma *compos-bij*:⟦*f* ∈ *A* → *B*; *bij-to f A B*; *g* ∈ *B* → *C*; *bij-to g B C*⟧ ⟹
bij-to (*compose A g f*) *A C*
⟨*proof*⟩

1.5 Nsets

definition
nset :: [*nat*, *nat*] ⇒ (*nat*) *set* **where**
nset i j = {*k*. *i* ≤ *k* ∧ *k* ≤ *j*}

definition
slide :: *nat* ⇒ *nat* ⇒ *nat* **where**
slide i j == *i* + *j*

definition
sliden :: *nat* ⇒ *nat* ⇒ *nat* **where**
sliden i j == *j* − *i*

definition
jointfun :: [*nat*, *nat* ⇒ 'a, *nat*, *nat* ⇒ 'a] ⇒ (*nat* ⇒ 'a) **where**
(*jointfun n f m g*) = (λ*i*. *if i* ≤ *n then f i else g* ((*sliden* (*Suc n*)) *i*))

definition
skip :: *nat* ⇒ (*nat* ⇒ *nat*) **where**
skip i = (λ*x*. (*if i* = *0 then Suc x else*
(*if x* ∈ {*j*. *j* ≤ (*i* − *Suc 0*)} *then x else Suc x*)))

lemma *nat-pos*:*0* ≤ (*l*::*nat*)
⟨*proof*⟩

lemma *Suc-pos*:*Suc k* ≤ *r* ⟹ *0* < *r*
⟨*proof*⟩

lemma *nat-pos2*:(*k*::*nat*) < *r* ⟹ *0* < *r*
⟨*proof*⟩

lemma *eq-le-not*:⟦(*a*::*nat*) ≤ *b*; ¬ *a* < *b* ⟧ ⟹ *a* = *b*
⟨*proof*⟩

lemma *im-of-constmap*:(*constmap* {*0*} {*a*}) ‘ {*0*} = {*a*}
⟨*proof*⟩

lemma *noteq-le-less*:⟦ *m* ≤ (*n*::*nat*); *m* ≠ *n* ⟧ ⟹ *m* < *n*
⟨*proof*⟩

lemma *nat-not-le-less*:(¬ (*n*::*nat*) ≤ *m*) = (*m* < *n*)
⟨*proof*⟩

lemma *self-le*:(*n*::*nat*) ≤ *n*
⟨*proof*⟩

lemma *n-less-Suc*:(*n*::*nat*) < *Suc n*
⟨*proof*⟩

lemma *less-diff-pos*:$i < (n::nat) \Longrightarrow 0 < n - i$
⟨*proof*⟩

lemma *less-diff-Suc*:$i < (n::nat) \Longrightarrow n - (Suc\ i) = (n - i) - (Suc\ 0)$
⟨*proof*⟩

lemma *less-pre-n*:$0 < n \Longrightarrow n - Suc\ 0 < n$
⟨*proof*⟩

lemma *Nset-inc-0*:$(0::nat) \in \{i.\ i \leq n\}$
⟨*proof*⟩

lemma *Nset-1*:$\{i.\ i \leq Suc\ 0\} = \{0,\ Suc\ 0\}$
⟨*proof*⟩

lemma *Nset-1-1*:$(k \leq Suc\ 0) = (k = 0 \vee k = Suc\ 0)$
⟨*proof*⟩

lemma *Nset-2*:$\{i,\ j\} = \{j,\ i\}$
⟨*proof*⟩

lemma *Nset-nonempty*:$\{i.\ i \leq (n::nat)\} \neq \{\}$
⟨*proof*⟩

lemma *Nset-le*:$x \in \{i.\ i \leq n\} \Longrightarrow x \leq n$
⟨*proof*⟩

lemma *n-in-Nsetn*:$(n::nat) \in \{i.\ i \leq n\}$
⟨*proof*⟩

lemma *Nset-pre*:$[\![\ (x::nat) \in \{i.\ i \leq (Suc\ n)\};\ x \neq Suc\ n\ ]\!] \Longrightarrow x \in \{i.\ i \leq n\}$
⟨*proof*⟩

lemma *Nset-pre1*:$\{i.\ i \leq (Suc\ n)\} - \{Suc\ n\} = \{i.\ i \leq n\}$
⟨*proof*⟩

lemma *le-Suc-mem-Nsetn*:$x \leq Suc\ n \Longrightarrow x - Suc\ 0 \in \{i.\ i \leq n\}$
⟨*proof*⟩

lemma *le-Suc-diff-le*:$x \leq Suc\ n \Longrightarrow x - Suc\ 0 \leq n$
⟨*proof*⟩

lemma *Nset-not-pre*:$[\![\ x \notin \{i.\ i \leq n\};\ x \in \{i.\ i \leq (Suc\ n)\}]\!] \Longrightarrow x = Suc\ n$
⟨*proof*⟩

lemma *mem-of-Nset*:$x \leq (n::nat) \Longrightarrow x \in \{i.\ i \leq n\}$
⟨*proof*⟩

lemma *less-mem-of-Nset*:$x < (n::nat) \Longrightarrow x \in \{i.\ i \leq n\}$
⟨*proof*⟩

lemma *Nset-nset*:$\{i.\ i \leq (Suc\ (n + m))\} = \{i.\ i \leq n\} \cup$
$nset\ (Suc\ n)\ (Suc\ (n + m))$
⟨*proof*⟩

lemma *Nset-nset-1*:$[\![0 < n;\ i < n]\!] \Longrightarrow \{j.\ j \leq n\} = \{j.\ j \leq i\} \cup$
$nset\ (Suc\ i)\ n$
⟨*proof*⟩

lemma *Nset-img0*:$[\![f \in \{j.\ j \leq Suc\ n\} \to B;\ (f\ (Suc\ n)) \in f\ `\ \{j.\ j \leq n\}]\!] \Longrightarrow$
$f\ `\ \{j.\ j \leq Suc\ n\} = f\ `\ \{j.\ j \leq n\}$
⟨*proof*⟩

lemma *Nset-img*:$f \in \{j.\ j \leq Suc\ n\} \to B \Longrightarrow$
$insert\ (f\ (Suc\ n))\ (f\ `\ \{j.\ j \leq n\}) = f\ `\ \{j.\ j \leq Suc\ n\}$
⟨*proof*⟩

primrec *nasc-seq* :: $[nat\ set,\ nat,\ nat] \Rightarrow nat$
where
dec-seq-0: $nasc\text{-}seq\ A\ a\ 0 = a$
| *dec-seq-Suc*: $nasc\text{-}seq\ A\ a\ (Suc\ n) =$
$(SOME\ b.\ ((b \in A) \wedge (nasc\text{-}seq\ A\ a\ n) < b))$

lemma *nasc-seq-mem*:$[\![(a::nat) \in A;\ \neg\ (\exists\, m.\ m{\in}A \wedge (\forall\, x{\in}A.\ x \leq m))]\!] \Longrightarrow$
$(nasc\text{-}seq\ A\ a\ n) \in A$
⟨*proof*⟩

lemma *nasc-seqn*:$[\![(a::nat) \in A;\ \neg\ (\exists\, m.\ m{\in}A \wedge (\forall\, x{\in}A.\ x \leq m))]\!] \Longrightarrow$
$(nasc\text{-}seq\ A\ a\ n) < (nasc\text{-}seq\ A\ a\ (Suc\ n))$
⟨*proof*⟩

lemma *nasc-seqn1*:$[\![(a::nat) \in A;\ \neg\ (\exists\, m.\ m{\in}A \wedge (\forall\, x{\in}A.\ x \leq m))]\!] \Longrightarrow$
$Suc\ (nasc\text{-}seq\ A\ a\ n) \leq (nasc\text{-}seq\ A\ a\ (Suc\ n))$
⟨*proof*⟩

lemma *ubs-ex-n-maxTr*:$[\![(a::nat) \in A;\ \neg\ (\exists\, m.\ m{\in}A \wedge (\forall\, x{\in}A.\ x \leq m))]\!]$
$\Longrightarrow\ (a + n) \leq (nasc\text{-}seq\ A\ a\ n)$
⟨*proof*⟩

lemma *ubs-ex-n-max*:$[\![A \neq \{\};\ A \subseteq \{i.\ i \leq (n::nat)\}]\!] \Longrightarrow$
$\exists!m.\ m{\in}A \wedge (\forall\, x{\in}A.\ x \leq m)$
⟨*proof*⟩

definition
n-max :: $nat\ set \Rightarrow nat$ **where**
$n\text{-}max\ A = (THE\ m.\ m \in A \wedge (\forall\, x{\in}A.\ x \leq m))$

lemma *n-max*:$[\![A \subseteq \{i.\ i \leq (n::nat)\};\ A \neq \{\}]\!] \Longrightarrow$
$(n\text{-}max\ A) \in A \wedge (\forall x{\in}A.\ x \leq (n\text{-}max\ A))$
⟨*proof*⟩

lemma *n-max-eq-sets*:$[\![A = B;\ A \neq \{\};\ \exists n.\ A \subseteq \{j.\ j \leq n\}]\!] \Longrightarrow$
$n\text{-}max\ A = n\text{-}max\ B$
⟨*proof*⟩

lemma *skip-mem*:$l \in \{i.\ i \leq n\} \Longrightarrow (skip\ i\ l) \in \{i.\ i \leq (Suc\ n)\}$
⟨*proof*⟩

lemma *skip-fun*:$(skip\ i) \in \{i.\ i \leq n\} \to \{i.\ i \leq (Suc\ n)\}$
⟨*proof*⟩

lemma *skip-im-Tr0*:$x \in \{i.\ i \leq n\} \Longrightarrow skip\ 0\ x = Suc\ x$
⟨*proof*⟩

lemma *skip-im-Tr0-1*:$0 < y \Longrightarrow skip\ 0\ (y - Suc\ 0) = y$
⟨*proof*⟩

lemma *skip-im-Tr1*:$[\![\ i \in \{i.\ i \leq (Suc\ n)\};\ 0 < i;\ x \leq i - Suc\ 0\ ]\!] \Longrightarrow$
$skip\ i\ x = x$
⟨*proof*⟩

lemma *skip-im-Tr1-1*:$[\![\ i \in \{i.\ i \leq (Suc\ n)\};\ 0 < i;\ x < i]\!] \Longrightarrow$
$skip\ i\ x = x$
⟨*proof*⟩

lemma *skip-im-Tr1-2*:$[\![\ i \leq (Suc\ n);\ x < i]\!] \Longrightarrow\ skip\ i\ x = x$
⟨*proof*⟩

lemma *skip-im-Tr2*:$[\![\ 0 < i;\ i \in \{i.\ i \leq (Suc\ n)\};\ i \leq x]\!] \Longrightarrow$
$skip\ i\ x = Suc\ x$
⟨*proof*⟩

lemma *skip-im-Tr2-1*:$[\![i \in \{i.\ i \leq (Suc\ n)\};\ i \leq x]\!] \Longrightarrow$
$skip\ i\ x = Suc\ x$
⟨*proof*⟩

lemma *skip-im-Tr3*:$x \in \{i.\ i \leq n\} \Longrightarrow skip\ (Suc\ n)\ x = x$
⟨*proof*⟩

lemma *skip-im-Tr4*:$[\![x \leq Suc\ n;\ 0 < x]\!] \Longrightarrow x - Suc\ 0 \leq n$
⟨*proof*⟩

lemma *skip-fun-im*:$i \in \{j.\ j \leq (Suc\ n)\} \Longrightarrow$
$(skip\ i)\ `\ \{j.\ j \leq n\} = (\{j.\ j \leq (Suc\ n)\} - \{i\})$
⟨*proof*⟩

lemma *skip-fun-im1*:$[\![i \in \{j.\ j \leq (Suc\ n)\};\ x \in \{j.\ j \leq n\}]\!] \Longrightarrow$
$(skip\ i)\ x \in (\{j.\ j \leq (Suc\ n)\} - \{i\})$
⟨*proof*⟩

lemma *skip-id*:$l < i \Longrightarrow skip\ i\ l = l$
⟨*proof*⟩

lemma *Suc-neq*:$[\![0 < i;\ i - Suc\ 0 < l]\!] \Longrightarrow i \neq Suc\ l$
⟨*proof*⟩

lemma *skip-il-neq-i*:$skip\ i\ l \neq i$
⟨*proof*⟩

lemma *skip-inj*:$[\![i \in \{k.\ k \leq n\};\ j \in \{k.\ k \leq n\};\ i \neq j]\!] \Longrightarrow$
$skip\ k\ i \neq skip\ k\ j$
⟨*proof*⟩

lemma *le-imp-add-int*: $i \leq (j{::}nat) \Longrightarrow \exists k.\ j = i + k$
⟨*proof*⟩

lemma *jointfun-hom0*:$[\![\ f \in \{j.\ j \leq n\} \to A;\ g \in \{k.\ k \leq m\} \to B\ ]\!] \Longrightarrow$
$(jointfun\ n\ f\ m\ g) \in \{l.\ l \leq (Suc\ (n + m))\} \to\ (A \cup B)$
⟨*proof*⟩

lemma *jointfun-mem*:$[\![\forall j \leq (n{::}nat).\ f\ j \in A;\ \forall j \leq m.\ g\ j \in B;$
$l \leq (Suc\ (n + m))]\!] \Longrightarrow (jointfun\ n\ f\ m\ g)\ l \in (A \cup B)$
⟨*proof*⟩

lemma *jointfun-inj*:$[\![f \in \{j.\ j \leq n\} \to B;\ inj\text{-}on\ f\ \{j.\ j \leq n\};$
$b \notin f\ `\ \{j.\ j \leq n\}]\!] \Longrightarrow$
$inj\text{-}on\ (jointfun\ n\ f\ 0\ (\lambda k{\in}\{0{::}nat\}.\ b))\ \{j.\ j \leq Suc\ n\}$
⟨*proof*⟩

lemma *slide-hom*:$i \leq j \Longrightarrow (slide\ i) \in \{l.\ l \leq (j - i)\} \to nset\ i\ j$
⟨*proof*⟩

lemma *slide-mem*:$[\![\ i \leq j;\ l \in \{k.\ k \leq (j - i)\}]\!] \Longrightarrow slide\ i\ l \in nset\ i\ j$
⟨*proof*⟩

lemma *slide-iM*:$(slide\ i)\ `\ \{l.\ 0 \leq l\} = \{k.\ i \leq k\}$
⟨*proof*⟩

lemma *jointfun-hom*:$[\![\ f \in \{i.\ i \leq n\} \to A;\ g \in \{j.\ j \leq m\} \to B\ ]\!] \Longrightarrow$
$(jointfun\ n\ f\ m\ g) \in \{j.\ j \leq (Suc\ (n + m))\} \to A \cup B$
⟨*proof*⟩

lemma *im-jointfunTr1*:$(jointfun\ n\ f\ m\ g)\ `\ \{i.\ i \leq n\} = f\ `\ \{i.\ i \leq n\}$
⟨*proof*⟩

lemma *im-jointfunTr2*:$(jointfun\ n\ f\ m\ g)\ `\ (nset\ (Suc\ n)\ (Suc\ (n + m))) =$
$g\ `\ (\{j.\ j \leq m\})$
⟨*proof*⟩

lemma *im-jointfun*:$[\![f \in \{j.\ j \leq n\} \to A;\ g \in \{j.\ j \leq m\} \to B]\!] \Longrightarrow$
$(jointfun\ n\ f\ m\ g)\ `(\{j.\ j \leq (Suc\ (n + m))\}) =$
$f\ `\{j.\ j \leq n\} \cup g\ `\{j.\ j \leq\ m\}$
⟨*proof*⟩

lemma *im-jointfun1*:$(jointfun\ n\ f\ m\ g)\ `(\{j.\ j \leq (Suc\ (n + m))\}) =$
$f\ `\{j.\ j \leq n\} \cup g\ `\ \{j.\ j \leq m\}$
⟨*proof*⟩

lemma *jointfun-surj*:$[\![f \in \{j.\ j \leq n\} \to A;\ surj\text{-}to\ f\ \{j.\ j \leq (n::nat)\}\ A;$
$g \in \{j.\ j \leq (m::nat)\} \to B;\ surj\text{-}to\ g\ \{j.\ j \leq m\}\ B]\!] \Longrightarrow$
$surj\text{-}to\ (jointfun\ n\ f\ m\ g)\ \{j.\ j \leq Suc\ (n + m)\}\ (A \cup B)$
⟨*proof*⟩

lemma *Nset-un*:$\{j.\ j \leq (Suc\ n)\} = \{j.\ j \leq n\} \cup \{Suc\ n\}$
⟨*proof*⟩

lemma *Nsetn-sub*: $\{j.\ j \leq n\} \subseteq \{j.\ j \leq (Suc\ n)\}$
⟨*proof*⟩

lemma *Nset-pre-sub*:$(0::nat) < k \Longrightarrow \{j.\ j \leq (k - Suc\ 0)\} \subseteq \{j.\ j \leq k\}$
⟨*proof*⟩

lemma *Nset-pre-un*:$(0::nat) < k \Longrightarrow \{j.\ j \leq k\} = \{j.\ j \leq (k - Suc\ 0)\} \cup \{k\}$
⟨*proof*⟩

lemma *Nsetn-sub-mem*: $l \in \{j.\ j \leq n\} \Longrightarrow l \in \{j.\ j \leq (Suc\ n)\}$
⟨*proof*⟩

lemma *Nsetn-sub-mem1*:$\forall j.\ j \in \{j.\ j \leq n\} \longrightarrow j \in \{j.\ j \leq (Suc\ n)\}$
⟨*proof*⟩

lemma *Nset-Suc*:$\{j.\ j \leq (Suc\ n)\} = insert\ (Suc\ n)\ \{j.\ j \leq n\}$
⟨*proof*⟩

lemma *nsetnm-sub-mem*:$\forall j.\ j \in nset\ n\ (n + m) \longrightarrow j \in nset\ n\ (Suc\ (n + m))$
⟨*proof*⟩

lemma *Nset-0*:$\{j.\ j \leq (0::nat)\} = \{0\}$
⟨*proof*⟩

lemma *Nset-Suc0*:$\{i.\ i \leq (Suc\ 0)\} = \{0,\ Suc\ 0\}$
⟨*proof*⟩

lemma *Nset-Suc-Suc*:$Suc\ (Suc\ 0) \leq n \Longrightarrow$
$\{j.\ j \leq (n - Suc\ (Suc\ 0))\} = \{j.\ j \leq n\} - \{n - Suc\ 0,\ n\}$
⟨*proof*⟩

lemma *func-pre*:$f \in \{j.\ j \leq (Suc\ n)\} \to A \Longrightarrow f \in \{j.\ j \leq n\} \to A$
⟨*proof*⟩

lemma *image-Nset-Suc*:$f\ `\ (\{j.\ j \leq (Suc\ n)\}) =$
$insert\ (f\ (Suc\ n))\ (f\ `\ \{j.\ j \leq n\})$
⟨*proof*⟩

definition
$Nleast :: nat\ set \Rightarrow nat$ **where**
$Nleast\ A = (THE\ a.\ (a \in A \wedge (\forall x{\in}A.\ a \leq x)))$

definition
$Nlb :: [nat\ set,\ nat] \Rightarrow bool$ **where**
$Nlb\ A\ n \longleftrightarrow (\forall a{\in}A.\ n \leq a)$

primrec *ndec-seq* :: $[nat\ set,\ nat,\ nat] \Rightarrow nat$ **where**
ndec-seq-0 :$ndec\text{-}seq\ A\ a\ 0 = a$
| *ndec-seq-Suc*:$ndec\text{-}seq\ A\ a\ (Suc\ n) =$
$(SOME\ b.\ ((b \in A) \wedge b < (ndec\text{-}seq\ A\ a\ n)))$

lemma *ndec-seq-mem*:$[\![a \in (A{::}nat\ set);\ \neg\ (\exists\, m.\ m{\in}A \wedge (\forall x{\in}A.\ m \leq x))]\!] \Longrightarrow$
$(ndec\text{-}seq\ A\ a\ n) \in A$
⟨*proof*⟩

lemma *ndec-seqn*:$[\![a \in (A{::}nat\ set);\neg\ (\exists\, m.\ m{\in}A \wedge (\forall x{\in}A.\ m \leq x))]\!] \Longrightarrow$
$(ndec\text{-}seq\ A\ a\ (Suc\ n)) < (ndec\text{-}seq\ A\ a\ n)$
⟨*proof*⟩

lemma *ndec-seqn1*:$[\![a \in (A{::}nat\ set);\ \neg\ (\exists\, m.\ m{\in}A \wedge (\forall x{\in}A.\ m \leq x))]\!] \Longrightarrow$
$(ndec\text{-}seq\ A\ a\ (Suc\ n)) \leq (ndec\text{-}seq\ A\ a\ n) - 1$
⟨*proof*⟩

lemma *ex-NleastTr*:$[\![a \in (A{::}nat\ set);\ \neg\ (\exists\, m.\ m{\in}A \wedge (\forall x{\in}A.\ m \leq x))]\!] \Longrightarrow$
$(ndec\text{-}seq\ A\ a\ n) \leq (a - n)$
⟨*proof*⟩

lemma *nat-le*:$((a{::}nat) - (a + 1)) \leq 0$
⟨*proof*⟩

lemma *ex-Nleast*:$(A{::}nat\ set) \neq \{\} \Longrightarrow \exists!m.\ m{\in}A \wedge (\forall x{\in}A.\ m \leq x)$
⟨*proof*⟩

lemma *Nleast*:$(A{::}nat\ set) \neq \{\} \Longrightarrow Nleast\ A \in A \wedge (\forall x{\in}A.\ (Nleast\ A) \leq x)$
⟨*proof*⟩

1.5.1 Lemmas for existence of reduced chain.

lemma *jointgd-tool1*: $0 < i \Longrightarrow 0 \leq i - Suc\ 0$
⟨*proof*⟩

lemma *jointgd-tool2*: $0 < i \Longrightarrow i = Suc\ (i - Suc\ 0)$
⟨*proof*⟩

lemma *jointgd-tool3*:$[\![0 < i;\ \ i \leq m]\!] \Longrightarrow i - Suc\ 0 \leq (m - Suc\ 0)$
⟨*proof*⟩

lemma *jointgd-tool4*:$n < i \Longrightarrow i - n = Suc(\ i - Suc\ n)$
⟨*proof*⟩

lemma *pos-prec-less*:$0 < i \Longrightarrow i - Suc\ 0 < i$
⟨*proof*⟩

lemma *Un-less-Un*:$[\![f \in \{j.\ j \leq (Suc\ n)\} \to (X::'a\ set\ set);$
$A \subseteq \bigcup (f\ `\ \{j.\ j \leq (Suc\ n)\});$
$i \in \{j.\ j \leq (Suc\ n)\};\ j \in \{l.\ l \leq (Suc\ n)\};\ i \neq j \wedge f\ i \subseteq f\ j]\!]$
$\Longrightarrow A \subseteq \bigcup (compose\ \{j.\ j \leq n\}\ f\ (skip\ i)\ `\ \{j.\ j \leq n\})$
⟨*proof*⟩

1.6 Lower bounded set of integers

definition $Zset = \{x.\ \exists (n::int).\ x = n\}$

definition
$Zleast :: int\ set \Rightarrow int$ **where**
$Zleast\ A = (THE\ a.\ (a \in A \wedge (\forall x \in A.\ a \leq x)))$

definition
$LB :: [int\ set,\ int] \Rightarrow bool$ **where**
$LB\ A\ n = (\forall a \in A.\ n \leq a)$

lemma *linorder-linear1*:$(m::int) < n \vee n \leq m$
⟨*proof*⟩

primrec *dec-seq* :: $[int\ set,\ int,\ nat] \Rightarrow int$
where
dec-seq-0: $dec\text{-}seq\ A\ a\ 0 = a$
| *dec-seq-Suc*: $dec\text{-}seq\ A\ a\ (Suc\ n) = (SOME\ b.\ ((b \in A) \wedge b < (dec\text{-}seq\ A\ a\ n)))$

lemma *dec-seq-mem*:$[\![a \in A;\ A \subseteq Zset;\neg\ (\exists m.\ m \in A \wedge (\forall x \in A.\ m \leq x))]\!] \Longrightarrow$
$(dec\text{-}seq\ A\ a\ n) \in A$
⟨*proof*⟩

lemma *dec-seqn*:$[\![a \in A;\ A \subseteq Zset;\neg\ (\exists m.\ m \in A \wedge (\forall x \in A.\ m \leq x))]\!] \Longrightarrow$
$(dec\text{-}seq\ A\ a\ (Suc\ n)) < (dec\text{-}seq\ A\ a\ n)$

⟨*proof*⟩

lemma *dec-seqn1*:⟦$a \in A$; $A \subseteq Zset$;¬ $(\exists\, m.\ m{\in}A \wedge (\forall\, x{\in}A.\ m \leq x))$⟧ ⟹
$(dec\text{-}seq\ A\ a\ (Suc\ n)) \leq (dec\text{-}seq\ A\ a\ n) - 1$
⟨*proof*⟩

lemma *lbs-ex-ZleastTr*:⟦$a \in A$; $A \subseteq Zset$;¬ $(\exists\, m.\ m{\in}A \wedge (\forall\, x{\in}A.\ m \leq x))$⟧ ⟹
$(dec\text{-}seq\ A\ a\ n) \leq (a - int(n))$
⟨*proof*⟩

lemma *big-int-less*:$a - int(nat(abs(a) + abs(N) + 1)) < N$
⟨*proof*⟩

lemma *lbs-ex-Zleast*:⟦$A \neq \{\}$; $A \subseteq Zset$; $LB\ A\ n$⟧ ⟹ $\exists !m.\ m{\in}A \wedge (\forall\, x{\in}A.\ m \leq x)$
⟨*proof*⟩

lemma *Zleast*:⟦$A \neq \{\}$; $A \subseteq Zset$; $LB\ A\ n$⟧ ⟹ $Zleast\ A \in A \wedge$
$(\forall\, x{\in}A.\ (Zleast\ A) \leq x)$
⟨*proof*⟩

lemma *less-convert1*:⟦ $a = c$; $a < b$ ⟧ ⟹ $c < b$
⟨*proof*⟩

lemma *less-convert2*:⟦$a = b$; $b < c$⟧ ⟹ $a < c$
⟨*proof*⟩

1.7 Augmented integer: integer and $\infty{-}\infty$

definition
$zag :: (int * int)\ set$ **where**
$zag = \{(x,y) \mid x\ y.\ x * y = (0{::}int) \wedge (y = -1 \vee y = 0 \vee y = 1)\}$

definition
$zag\text{-}pl{::}[(int * int), (int * int)] \Rightarrow (int * int)$ **where**
$zag\text{-}pl\ x\ y == if\ (snd\ x + snd\ y) = 2\ then\ (0, 1)$
$else\ if\ (snd\ x + snd\ y) = 1\ then\ (0, 1)$
$else\ if\ (snd\ x + snd\ y) = 0\ then\ (fst\ x + fst\ y, 0)$
$else\ if\ (snd\ x + snd\ y) = -1\ then\ (0, -1)$
$else\ if\ (snd\ x + snd\ y) = -2\ then\ (0, -1)\ else\ undefined$

definition
$zag\text{-}t :: [(int * int), (int * int)] \Rightarrow (int * int)$ **where**
$zag\text{-}t\ x\ y = (if\ (snd\ x){*}(snd\ y) = 0\ then$
$(if\ 0 < (fst\ x){*}(snd\ y) + (snd\ x){*}(fst\ y)\ then\ (0,1)$
$else\ (if\ (fst\ x){*}(snd\ y) + (snd\ x){*}(fst\ y) = 0$
$then\ ((fst\ x){*}(fst\ y), 0)\ else\ (0, -1)))$
$else\ (if\ 0 < (snd\ x){*}(snd\ y)\ then\ (0, 1)\ else\ (0, -1)))$

definition *Ainteg* = *zag*

typedef *ant* = *Ainteg*
morphisms *Rep-Ainteg Abs-Ainteg*
⟨*proof*⟩

definition
ant :: *int* ⇒ *ant* **where**
ant m = *Abs-Ainteg*((*m*, *0*))

definition
tna :: *ant* ⇒ *int* **where**
tna z = (*if Rep-Ainteg*(*z*) ≠ (*0*,*1*) ∧ *Rep-Ainteg*(*z*) ≠ (*0*,−*1*) *then*
fst (*Rep-Ainteg*(*z*)) *else undefined*)

instantiation *ant* :: {*zero*, *one*, *plus*, *uminus*, *minus*, *times*, *ord*}
begin

definition
Zero-ant-def : *0* == *ant 0*

definition
One-ant-def : *1* == *ant 1*

definition
add-ant-def:
z + *w* ==
Abs-Ainteg (*zag-pl* (*Rep-Ainteg z*) (*Rep-Ainteg w*))

definition
minus-ant-def : − *z* ==
Abs-Ainteg((− (*fst* (*Rep-Ainteg z*)), − (*snd* (*Rep-Ainteg z*))))

definition
diff-ant-def: *z* − (*w*::*ant*) == *z* + (−*w*)

definition
mult-ant-def:
z ∗ *w* ==
Abs-Ainteg (*zag-t* (*Rep-Ainteg z*) (*Rep-Ainteg w*))

definition
le-ant-def:
(*z*::*ant*) ≤ *w* == *if* (*snd* (*Rep-Ainteg w*)) = *1 then True*
else (*if* (*snd* (*Rep-Ainteg w*)) = *0 then* (*if* (*snd* (*Rep-Ainteg z*)) = *1*
then False else (*if* (*snd* (*Rep-Ainteg z*)) = *0 then*
(*fst* (*Rep-Ainteg z*)) ≤ (*fst* (*Rep-Ainteg w*)) *else True*))
else (*if snd* (*Rep-Ainteg z*) = −*1 then True else False*))

definition
less-ant-def: $((z::ant) < (w::ant)) == (z \leq w \wedge z \neq w)$

instance ⟨*proof*⟩

end

definition
$inf\text{-}ant :: ant \ \ (\langle\infty\rangle)$ **where**
$\infty = Abs\text{-}Ainteg((0,1))$

definition
$an :: nat \Rightarrow ant$ **where**
$an\ m = ant\ (int\ m)$

definition
$na :: ant \Rightarrow nat$ **where**
$na\ x = (if\ (x < 0)\ then\ 0\ else$
$\quad if\ x \neq \infty\ then\ (nat\ (tna\ x))\ else\ undefined)$

definition
$UBset :: ant \Rightarrow ant\ set$ **where**
$UBset\ z = \{(x::ant).\ \ x \leq z\}$

definition
$LBset :: ant \Rightarrow ant\ set$ **where**
$LBset\ z = \{(x::ant).\ z \leq x\}$

lemma *ant-z-in-Ainteg*:$(z::int,\ 0) \in Ainteg$
⟨*proof*⟩

lemma *ant-inf-in-Ainteg*:$((0::int),\ 1) \in Ainteg$
⟨*proof*⟩

lemma *ant-minf-in-Ainteg*:$((0::int),\ -1) \in Ainteg$
⟨*proof*⟩

lemma *ant-0-in-Ainteg*:$((0::int),\ 0) \in Ainteg$
⟨*proof*⟩

lemma *an-0*[*simp*]:$an\ 0 = 0$
⟨*proof*⟩

lemma *an-1*[*simp*]:$an\ 1 = 1$
⟨*proof*⟩

lemma *mem-ant*:$(a::ant) = -\infty \vee (\exists (z::int).\ a = ant\ z) \vee a = \infty$
⟨*proof*⟩

lemma *minf*:$-\infty = Abs\text{-}Ainteg((0,-1))$
⟨*proof*⟩

lemma *z-neq-inf*[*simp*]:$(ant\ z) \neq \infty$
⟨*proof*⟩

lemma *z-neq-minf*[*simp*]:$(ant\ z) \neq -\infty$
⟨*proof*⟩

lemma *minf-neq-inf*[*simp*]:$-\infty \neq \infty$
⟨*proof*⟩

lemma *a-ipi*[*simp*]:$\infty + \infty = \infty$
⟨*proof*⟩

lemma *a-zpi*[*simp*]:$(ant\ z) + \infty\ = \infty$
⟨*proof*⟩

lemma *a-ipz*[*simp*]: $\infty + (ant\ z) = \infty$
⟨*proof*⟩

lemma *a-zpz*:$(ant\ m) + (ant\ n) = ant\ (m + n)$
⟨*proof*⟩

lemma *a-mpi*[*simp*]:$-\infty + \infty\ = 0$
⟨*proof*⟩

lemma *a-ipm*[*simp*]:$\infty + (-\infty) = 0$
⟨*proof*⟩

lemma *a-mpm*[*simp*]:$-\infty + (-\infty) = -\infty$
⟨*proof*⟩

lemma *a-mpz*[*simp*]:$-\infty + (ant\ m) = -\infty$
⟨*proof*⟩

lemma *a-zpm*[*simp*]:$(ant\ m) + (-\infty) = -\infty$
⟨*proof*⟩

lemma *a-mdi*[*simp*]:$-\infty - \infty\ = -\infty$
⟨*proof*⟩

lemma *a-zdz*:$(ant\ m) - (ant\ n) = ant\ (m - n)$
⟨*proof*⟩

lemma *a-i-i*[*simp*]:$\infty * \infty = \infty$
⟨*proof*⟩

lemma *a-0-i*[*simp*]:$0 * \infty = 0$
⟨*proof*⟩

lemma *a-i-0*[*simp*]:$\infty * 0 = 0$
⟨*proof*⟩

lemma *a-0-m*[*simp*]:$0 * (-\infty) = 0$
⟨*proof*⟩

lemma *a-m-0*[*simp*]:$(-\infty) * 0 = 0$
⟨*proof*⟩

lemma *a-m-i*[*simp*]:$(-\infty) * \infty = -\infty$
⟨*proof*⟩

lemma *a-i-m*[*simp*]:$\infty * (-\infty) = - \infty$
⟨*proof*⟩

lemma *a-pos-i*[*simp*]:$0 < m \Longrightarrow (ant\ m) * \infty = \infty$
⟨*proof*⟩

lemma *a-i-pos*[*simp*]:$0 < m \Longrightarrow \infty * (ant\ m) = \infty$
⟨*proof*⟩

lemma *a-neg-i*[*simp*]:$m < 0 \Longrightarrow (ant\ m) * \infty = -\infty$
⟨*proof*⟩

lemma *a-i-neg*[*simp*]:$m < 0 \Longrightarrow \infty * (ant\ m) = -\infty$
⟨*proof*⟩

lemma *a-z-z*:$(ant\ m) * (ant\ n) = ant\ (m*n)$
⟨*proof*⟩

lemma *a-pos-m*[*simp*]:$0 < m \Longrightarrow (ant\ m) * (-\infty) = -\infty$
⟨*proof*⟩

lemma *a-m-pos*[*simp*]:$0 < m \Longrightarrow (-\infty) * (ant\ m) = -\infty$
⟨*proof*⟩

lemma *a-neg-m*[*simp*]:$m < 0 \Longrightarrow (ant\ m) * (-\infty) = \infty$
⟨*proof*⟩

lemma *neg-a-m*[*simp*]:$m < 0 \Longrightarrow (-\infty) * (ant\ m) = \infty$
⟨*proof*⟩

lemma *a-m-m*[*simp*]:$(-\infty) * (-\infty) = \infty$
⟨*proof*⟩

lemma *inj-on-Abs-Ainteg*:*inj-on Abs-Ainteg Ainteg*
⟨*proof*⟩

lemma *an-Suc*:*an* (*Suc n*) = *an n* + *1*
⟨*proof*⟩

lemma *aeq-zeq* [*iff*]: (*ant m* = *ant n*) = (*m* = *n*)
⟨*proof*⟩

lemma *aminus*:− *ant m* = *ant* (−*m*)
⟨*proof*⟩

lemma *aminusZero*:− *ant 0* = *ant 0*
⟨*proof*⟩

lemma *ant-0*: *ant 0* = (*0*::*ant*)
⟨*proof*⟩

lemma *inf-neq-0*[*simp*]:∞ ≠ *0*
⟨*proof*⟩

lemma *zero-neq-inf*[*simp*]:*0* ≠ ∞
⟨*proof*⟩

lemma *minf-neq-0*[*simp*]:−∞ ≠ *0*
⟨*proof*⟩

lemma *zero-neq-minf*[*simp*]:*0* ≠ −∞
⟨*proof*⟩

lemma *a-minus-zero*[*simp*]:−(*0*::*ant*) = *0*
⟨*proof*⟩

lemma *a-minus-minus*: − (− *z*) = (*z*::*ant*)
⟨*proof*⟩

lemma *aminus-0*: − (− *0*) = (*0*::*ant*)
⟨*proof*⟩

lemma *a-a-z-0*:⟦ *0* < *z*; *a* ∗ *ant z* = *0*⟧ ⟹ *a* = *0*
⟨*proof*⟩

lemma *adiv-eq*:⟦ *z* ≠ *0*; *a* ∗ (*ant z*) = *b* ∗ (*ant z*)⟧ ⟹ *a* = *b*
⟨*proof*⟩

lemma *aminus-add-distrib*: − (*z* + *w*) = (− *z*) + (− *w*::*ant*)
⟨*proof*⟩

lemma *aadd-commute*:$(x::ant) + y = y + x$
⟨*proof*⟩

definition
aug-inf :: *ant set* (‹Z_∞›) **where**
$Z_\infty = \{(z::ant).\ z \neq -\infty\ \}$

definition
aug-minf :: *ant set* (‹$Z_{-\infty}$›) **where**
$Z_{-\infty} = \{(z::ant).\ z \neq \infty\ \}$

lemma *z-in-aug-inf*:*ant* $z \in Z_\infty$
⟨*proof*⟩

lemma *Zero-in-aug-inf*:$0 \in Z_\infty$
⟨*proof*⟩

lemma *z-in-aug-minf*:*ant* $z \in Z_{-\infty}$
⟨*proof*⟩

lemma *mem-aug-minf*:$a \in Z_{-\infty} \Longrightarrow a = -\infty \vee (\exists z.\ a = ant\ z)$
⟨*proof*⟩

lemma *minus-an-in-aug-minf*: $-$ *an* $n \in\ Z_{-\infty}$
⟨*proof*⟩

lemma *Zero-in-aug-minf*:$0 \in Z_{-\infty}$
⟨*proof*⟩

lemma *aadd-assoc-i*: $[\![x \in Z_\infty;\ y \in Z_\infty;\ z \in Z_\infty]\!] \Longrightarrow (x + y) + z = x + (y + z)$
⟨*proof*⟩

lemma *aadd-assoc-m*: $[\![x \in Z_{-\infty};\ y \in Z_{-\infty};\ z \in Z_{-\infty}]\!] \Longrightarrow$
$(x + y) + z = x + (y + z)$
⟨*proof*⟩

lemma *aadd-0-r*: $x + (0::ant) = x$
⟨*proof*⟩

lemma *aadd-0-l*: $(0::ant) + x = x$
⟨*proof*⟩

lemma *aadd-minus-inv*: $(-\ x) + x = (0::ant)$
⟨*proof*⟩

lemma *aadd-minus-r*: $x + (-\ x) = (0::ant)$
⟨*proof*⟩

lemma *ant-minus-inj*:$ant\ z \neq ant\ w \Longrightarrow -\ ant\ z \neq -\ ant\ w$

⟨*proof*⟩

lemma *aminus-mult-minus*: $(-\ (ant\ z)) * (ant\ w) = -\ ((ant\ z) * (ant\ w))$
⟨*proof*⟩

lemma *amult-commute*: $(x::ant) * y = y * x$
⟨*proof*⟩

lemma *z-le-i*[*simp*]:$(ant\ x) \leq \infty$
⟨*proof*⟩

lemma *z-less-i*[*simp*]:$(ant\ x) < \infty$
⟨*proof*⟩

lemma *m-le-z*:$-\infty \leq (ant\ x)$
⟨*proof*⟩

lemma *m-less-z*[*simp*]:$-\infty < (ant\ x)$
⟨*proof*⟩

lemma *noninf-mem-Z*:$[\![x \in Z_\infty;\ x \neq \infty]\!] \Longrightarrow \exists\,(z::int).\ x = ant\ z$
⟨*proof*⟩

lemma *z-mem-Z*:$ant\ z \in Z_\infty$
⟨*proof*⟩

lemma *inf-ge-any*[*simp*]:$x \leq \infty$
⟨*proof*⟩

lemma *zero-lt-inf*:$0 < \infty$
⟨*proof*⟩

lemma *minf-le-any*[*simp*]:$-\infty \leq x$
⟨*proof*⟩

lemma *minf-less-0*:$-\infty < 0$
⟨*proof*⟩

lemma *ale-antisym*[*simp*]:$[\![(x::ant) \leq y;\ y \leq x\ ]\!] \Longrightarrow x = y$
⟨*proof*⟩

lemma *x-gt-inf*[*simp*]:$\infty \leq x \Longrightarrow x = \infty$
⟨*proof*⟩

lemma *Zinf-pOp-closed*:$[\![x \in Z_\infty;\ y \in Z_\infty]\!] \Longrightarrow x + y \in Z_\infty$
⟨*proof*⟩

lemma *Zminf-pOp-closed*:$[\![x \in Z_{-\infty};\ y \in Z_{-\infty}]\!] \Longrightarrow x + y \in Z_{-\infty}$
⟨*proof*⟩

lemma *amult-distrib1*:$(ant\ z) \neq 0 \Longrightarrow$
$(a + b) * (ant\ z) = a * (ant\ z) + b * (ant\ z)$
⟨*proof*⟩

lemma *amult-0-r*:$(ant\ z) * 0 = 0$
⟨*proof*⟩

lemma *amult-0-l*:$0 * (ant\ z) = 0$
⟨*proof*⟩

definition
asprod :: [*int*, *ant*] ⇒ *ant* (**infixl** ‹$*_a$› *200*) **where**
$m *_a x ==$
if $x = \infty$ *then* (*if* $0 < m$ *then* ∞ *else* (*if* $m < 0$ *then* $-\infty$ *else*
if $m = 0$ *then* 0 *else undefined*))
else (*if* $x = -\infty$ *then*
(*if* $0 < m$ *then* $-\infty$ *else* (*if* $m < 0$ *then* ∞ *else*
if $m = 0$ *then* 0 *else undefined*))
else $(ant\ m) * x$)

lemma *asprod-pos-inf*[*simp*]:$0 < m \Longrightarrow m *_a \infty = \infty$
⟨*proof*⟩

lemma *asprod-neg-inf*[*simp*]:$m < 0 \Longrightarrow m *_a \infty = -\infty$
⟨*proof*⟩

lemma *asprod-pos-minf*[*simp*]:$0 < m \Longrightarrow m *_a (-\infty) = (-\infty)$
⟨*proof*⟩

lemma *asprod-neg-minf*[*simp*]:$m < 0 \Longrightarrow m *_a (-\infty) = \infty$
⟨*proof*⟩

lemma *asprod-mult*: $m *_a (ant\ n) = ant(m * n)$
⟨*proof*⟩

lemma *asprod-1*:$1 *_a x = x$
⟨*proof*⟩

lemma *agsprod-assoc-a*:$m *_a (n *_a (ant\ x)) = (m * n) *_a (ant\ x)$
⟨*proof*⟩

lemma *agsprod-assoc*:$[\![m \neq 0;\ n \neq 0]\!] \Longrightarrow m *_a (n *_a x) = (m * n) *_a x$
⟨*proof*⟩

lemma *asprod-distrib1*:$m \neq 0 \Longrightarrow m *_a (x + y) = (m *_a x) + (m *_a y)$
⟨*proof*⟩

lemma *asprod-0-x*[*simp*]:$0 *_a x = 0$
⟨*proof*⟩

lemma *asprod-n-0*:$n *_a 0 = 0$
⟨*proof*⟩

lemma *asprod-distrib2*:$[\![0 < i; 0 < j]\!] \Longrightarrow (i + j) *_a x = (i *_a x) + (j *_a x)$
⟨*proof*⟩

lemma *asprod-minus*:$x \neq -\infty \wedge x \neq \infty \Longrightarrow - z *_a x = z *_a (- x)$
⟨*proof*⟩

lemma *asprod-div-eq*:$[\![n \neq 0; n *_a x = n *_a y]\!] \Longrightarrow x = y$
⟨*proof*⟩

lemma *asprod-0*:$[\![z \neq 0; z *_a x = 0]\!] \Longrightarrow x = 0$
⟨*proof*⟩

lemma *asp-z-Z*:$z *_a ant x \in Z_\infty$
⟨*proof*⟩

lemma *tna-ant*: $tna (ant z) = z$
⟨*proof*⟩

lemma *ant-tna*:$x \neq \infty \wedge x \neq -\infty \Longrightarrow ant (tna x) = x$
⟨*proof*⟩

lemma *ant-sol*:$[\![a \in Z_\infty; b \in Z_\infty; c \in Z_\infty; b \neq \infty; a = b + c]\!] \Longrightarrow a - b = c$
⟨*proof*⟩

1.7.1 Ordering of integers and ordering nats

1.7.2 The $\leq$ Ordering

lemma *zneq-aneq*:$(n \neq m) = ((ant n) \neq (ant m))$
⟨*proof*⟩

lemma *ale*:$(n \leq m) = ((ant n) \leq (ant m))$
⟨*proof*⟩

lemma *aless*:$(n < m) = ((ant n) < (ant m))$
⟨*proof*⟩

lemma *ale-refl*: $w \leq (w::ant)$
⟨*proof*⟩

lemma *aeq-ale*:$(a::ant) = b \Longrightarrow a \leq b$
⟨*proof*⟩

lemma *ale-trans*: $[\![(i::ant) \leq j; j \leq k]\!] \Longrightarrow i \leq k$
⟨*proof*⟩

lemma *aless-le-not-le*: $((w::ant) < z) = (w \leq z \wedge \neg\ z \leq w)$
⟨*proof*⟩

instance *ant* :: *order*
⟨*proof*⟩

lemma *ale-linear*: $(z::ant) \leq w \vee w \leq z$
⟨*proof*⟩

instance *ant* :: *linorder*
⟨*proof*⟩

lemmas *aless-linear* = *less-linear* [**where** $'a = ant$]

lemma *ant-eq-0-conv* [*simp*]: $(ant\ n = 0) = (n = 0)$
⟨*proof*⟩

lemma *aless-zless*: $(ant\ m < ant\ n) = (m<n)$
⟨*proof*⟩

lemma *a0-less-int-conv* [*simp*]: $(0 < ant\ n) = (0 < n)$
⟨*proof*⟩

lemma *a0-less-1*: $0 < (1::ant)$
⟨*proof*⟩

lemma *a0-neq-1* [*simp*]: $0 \neq (1::ant)$
⟨*proof*⟩

lemma *ale-zle* [*simp*]: $((ant\ i) \leq (ant\ j)) = (i \leq j)$
⟨*proof*⟩

lemma *ant-1* [*simp*]: $ant\ 1 = 1$
⟨*proof*⟩

lemma *zpos-apos*:$(0 \leq n) = (0 \leq (ant\ n))$
⟨*proof*⟩

lemma *zposs-aposss*:$(0 < n) = (0 < (ant\ n))$
⟨*proof*⟩

lemma *an-nat-pos*[*simp*]:$0 \leq an\ n$
⟨*proof*⟩

lemma *amult-one-l*: $1 * (x::ant) = x$
⟨*proof*⟩

lemma *amult-one-r*:$(x::ant) * 1 = x$
⟨*proof*⟩

lemma *amult-eq-eq-r*:$[\![z \neq 0;\ \ a * ant\ z = b * ant\ z]\!] \Longrightarrow a = b$
⟨*proof*⟩

lemma *amult-eq-eq-l*:$[\![z \neq 0;\ \ (ant\ z) * a = (ant\ z) * b]\!] \Longrightarrow a = b$
⟨*proof*⟩

lemma *amult-pos*:$[\![0 < b;\ 0 \leq x]\!] \Longrightarrow x \leq (b *_a x)$
⟨*proof*⟩

lemma *asprod-amult*:$0 < z \Longrightarrow z *_a x = (ant\ z) * x$
⟨*proof*⟩

lemma *amult-pos1*:$[\![0 < b;\ 0 \leq x]\!] \Longrightarrow x \leq ((ant\ b) * x)$
⟨*proof*⟩

lemma *amult-pos-mono-l*:$0 < w \Longrightarrow (((ant\ w) * x) \leq ((ant\ w) * y)) = \ (x \leq y)$
⟨*proof*⟩

lemma *amult-pos-mono-r*:$0 < w \Longrightarrow ((x * (ant\ w)) \leq (y * (ant\ w))) = \ (x \leq y)$
⟨*proof*⟩

lemma *apos-neq-minf*:$0 \leq a \Longrightarrow a \neq -\infty$
⟨*proof*⟩

lemma *asprod-pos-mono*:$0 < w \Longrightarrow ((w *_a x) \leq (w *_a y)) = \ (x \leq y)$
⟨*proof*⟩

lemma *a-inv*:$(a::ant) + b = 0 \Longrightarrow a = -b$
⟨*proof*⟩

lemma *asprod-pos-pos*:$0 \leq x \Longrightarrow 0 \leq int\ n *_a x$
⟨*proof*⟩

lemma *asprod-1-x*[*simp*]:$1 *_a x = x$
⟨*proof*⟩

lemma *asprod-n-1*[*simp*]:$n *_a 1 = ant\ n$
⟨*proof*⟩

1.7.3 Aug ordering

lemma *aless-imp-le*: $x < (y::ant) \Longrightarrow x \leq y$

⟨*proof*⟩

lemma *gt-a0-ge-1*:$(0::ant) < x \Longrightarrow 1 \leq x$
⟨*proof*⟩

lemma *gt-a0-ge-aN*:$[\![0 < x;\ N \neq 0]\!] \Longrightarrow (ant\ (int\ N)) \leq (int\ N) *_a x$
⟨*proof*⟩

lemma *aless-le-trans*:$[\![(x::ant) < y;\ y \leq z]\!] \Longrightarrow x < z$
⟨*proof*⟩

lemma *ale-less-trans*:$[\![(x::ant) \leq y;\ y < z]\!] \Longrightarrow x < z$
⟨*proof*⟩

lemma *aless-trans*:$[\![(x::ant) < y;\ y < z]\!] \Longrightarrow x < z$
⟨*proof*⟩

lemma *ale-neq-less*:$[\![\ (x::ant) \leq y;\ x \neq y]\!] \Longrightarrow x < y$
⟨*proof*⟩

lemma *aneg-le*:$(\neg\ (x::ant) \leq y) = (y\ <\ x)$
⟨*proof*⟩

lemma *aneg-less*:$(\neg\ x < (y::ant)) = (y \leq x)$
⟨*proof*⟩

lemma *aadd-le-mono*:$x \leq (y::ant) \Longrightarrow x + z \leq y + z$
⟨*proof*⟩

lemma *aadd-less-mono-z*:$(x::ant) < y \Longrightarrow (x + (ant\ z)) < (y + (ant\ z))$
⟨*proof*⟩

lemma *aless-le-suc*[*simp*]:$(a::ant) < b \Longrightarrow a + 1 \leq b$
⟨*proof*⟩

lemma *aposs-le-1*:$(0::ant) < x \Longrightarrow 1 \leq x$
⟨*proof*⟩

lemma *pos-in-aug-inf*:$(0::ant) \leq x \Longrightarrow x \in Z_\infty$
⟨*proof*⟩

lemma *aug-inf-noninf-is-z*:$[\![x \in Z_\infty;\ x \neq \infty]\!] \Longrightarrow \exists z.\ x = ant\ z$
⟨*proof*⟩

lemma *aadd-two-pos*:$[\![0 \leq (x::ant);\ 0 \leq y]\!] \Longrightarrow 0 \leq x + y$
⟨*proof*⟩

lemma *aadd-pos-poss*:$[\![(0::ant) \leq x;\ 0 < y]\!] \Longrightarrow 0 < (x + y)$
⟨*proof*⟩

lemma *aadd-poss-pos*:$[\![(0::ant) < x;\ 0 \leq y]\!] \Longrightarrow 0 < (x + y)$
⟨*proof*⟩

lemma *aadd-pos-le*:$0 \leq (a::ant) \Longrightarrow b \leq a + b$
⟨*proof*⟩

lemma *aadd-poss-less*:$[\![b \neq \infty;\ b \neq -\infty;\ 0 < a]\!] \Longrightarrow b < a + b$
⟨*proof*⟩

lemma *ale-neg*:$(0::ant) \leq x \Longrightarrow (-\ x) \leq 0$
⟨*proof*⟩

lemma *ale-diff-pos*:$(x::ant) \leq y \Longrightarrow 0 \leq (y - x)$
⟨*proof*⟩

lemma *aless-diff-poss*:$(x::ant) < y \Longrightarrow 0 < (y - x)$
⟨*proof*⟩

lemma *ale-minus*: $(x::ant) \leq y \Longrightarrow -\ y \leq -\ x$
⟨*proof*⟩

lemma *aless-minus*:$(x::ant) < y \Longrightarrow -\ y < -\ x$
⟨*proof*⟩

lemma *aadd-minus-le*:$(a::ant) \leq 0 \Longrightarrow a + b \leq b$
⟨*proof*⟩

lemma *aadd-minus-less*:$[\![b \neq -\infty \wedge b \neq \infty;\ (a::ant) < 0]\!] \Longrightarrow a + b < b$
⟨*proof*⟩

lemma *an-inj*:$an\ n = an\ m \Longrightarrow n = m$
⟨*proof*⟩

lemma *nat-eq-an-eq*:$n = m \Longrightarrow an\ n = an\ m$
⟨*proof*⟩

lemma *aneq-natneq*:$(an\ n \neq an\ m) = (n \neq m)$
⟨*proof*⟩

lemma *ale-natle*: $(an\ n \leq an\ m) = (n \leq m)$
⟨*proof*⟩

lemma *aless-natless*:$(an\ n < an\ m) = (n < m)$
⟨*proof*⟩

lemma *na-an*:$na\ (an\ n) = n$
⟨*proof*⟩

lemma *asprod-ge*:
$0 < b \Longrightarrow N \neq 0 \Longrightarrow an\ N \leq int\ N *_a b$
⟨*proof*⟩

lemma *an-npn*:$an\ (n + m) = an\ n + an\ m$
⟨*proof*⟩

lemma *an-ndn*:$n \leq m \Longrightarrow an\ (m - n) = an\ m - an\ n$
⟨*proof*⟩

1.8 Amin, amax

definition
$amin :: [ant, ant] \Rightarrow ant$ **where**
$amin\ x\ y = (if\ (x \leq y)\ then\ x\ else\ y)$

definition
$amax :: [ant, ant] \Rightarrow ant$ **where**
$amax\ x\ y = (if\ (x \leq y)\ then\ y\ else\ x)$

primrec $Amin :: [nat, nat \Rightarrow ant] \Rightarrow ant$
where
$Amin\text{-}0 :\ Amin\ 0\ f = (f\ 0)$
| $Amin\text{-}Suc : Amin\ (Suc\ n)\ f = amin\ (Amin\ n\ f)\ (f\ (Suc\ n))$

primrec $Amax :: [nat, nat \Rightarrow ant] \Rightarrow ant$
where
$Amax\text{-}0 : Amax\ 0\ f = f\ 0$
| $Amax\text{-}Suc : Amax\ (Suc\ n)\ f = amax\ (Amax\ n\ f)\ (f\ (Suc\ n))$

lemma *amin-ge*:$x \leq amin\ x\ y \vee y \leq amin\ x\ y$
⟨*proof*⟩

lemma *amin-le-l*:$amin\ x\ y \leq x$
⟨*proof*⟩

lemma *amin-le-r*:$amin\ x\ y \leq y$
⟨*proof*⟩

lemma *amax-le*:$amax\ x\ y \leq x \vee amax\ x\ y \leq y$
⟨*proof*⟩

lemma *amax-le-n*:$[\![x \leq n; y \leq n]\!] \Longrightarrow amax\ x\ y \leq n$
⟨*proof*⟩

lemma *amax-ge-l*:$x \leq amax\ x\ y$
⟨*proof*⟩

lemma *amax-ge-r*:$y \leq amax\ x\ y$

⟨*proof*⟩

lemma *amin-mem-i*:⟦$x \in Z_\infty$; $y \in Z_\infty$⟧ ⟹ *amin* x $y \in Z_\infty$
⟨*proof*⟩

lemma *amax-mem-m*:⟦$x \in Z_{-\infty}$; $y \in Z_{-\infty}$⟧ ⟹ *amax* x $y \in Z_{-\infty}$
⟨*proof*⟩

lemma *amin-commute*:*amin* x y = *amin* y x
⟨*proof*⟩

lemma *amin-mult-pos*:$0 < z$ ⟹ *amin* $(z *_a x)$ $(z *_a y)$ = $z *_a$ *amin* x y
⟨*proof*⟩

lemma *amin-amult-pos*:$0 < z$ ⟹
amin $((ant\ z) * x)$ $((ant\ z) * y)$ = $(ant\ z) *$ *amin* x y
⟨*proof*⟩

lemma *times-amin*:⟦$0 < a$; *amin* $(x * (ant\ a))$ $(y * (ant\ a)) \leq z * (ant\ a)$⟧ ⟹
amin x $y \leq z$
⟨*proof*⟩

lemma *Amin-memTr*:$f \in \{i.\ i \leq n\} \to Z_\infty \longrightarrow$ *Amin* n $f \in Z_\infty$
⟨*proof*⟩

lemma *Amin-mem*:$f \in \{i.\ i \leq n\} \to Z_\infty$ ⟹ *Amin* n $f \in Z_\infty$
⟨*proof*⟩

lemma *Amax-memTr*:$f \in \{i.\ i \leq n\} \to Z_{-\infty} \longrightarrow$ *Amax* n $f \in Z_{-\infty}$
⟨*proof*⟩

lemma *Amax-mem*:$f \in \{i.\ i \leq n\} \to Z_{-\infty}$ ⟹ *Amax* n $f \in Z_{-\infty}$
⟨*proof*⟩

lemma *Amin-mem-mem*:$\forall j \leq n.\ f\ j \in Z_\infty$ ⟹ *Amin* n $f \in Z_\infty$
⟨*proof*⟩

lemma *Amax-mem-mem*:$\forall j \leq n.\ f\ j \in Z_{-\infty}$ ⟹ *Amax* n $f \in Z_{-\infty}$
⟨*proof*⟩

lemma *Amin-leTr*:$f \in \{i.\ i \leq n\} \to Z_\infty \longrightarrow (\forall j \in \{i.\ i \leq n\}.$ *Amin* n $f \leq (f\ j))$
⟨*proof*⟩

lemma *Amin-le*:⟦$f \in \{j.\ j \leq n\} \to Z_\infty$; $j \in \{k.\ k \leq n\}$⟧ ⟹ *Amin* n $f \leq (f\ j)$
⟨*proof*⟩

lemma *Amax-geTr*:$f \in \{j.\ j \leq n\} \to Z_{-\infty} \longrightarrow (\forall j \in \{j.\ j \leq n\}.\ (f\ j) \leq$ *Amax* n $f)$
⟨*proof*⟩

lemma *Amax-ge*:$[\![f \in \{j.\ j \leq n\} \to Z_{-\infty};\ j \in \{j.\ j \leq n\}]\!] \Longrightarrow$
$(f\ j) \leq (Amax\ n\ f)$
⟨*proof*⟩

lemma *Amin-mem-le*:$[\![\forall j \leq n.\ (f\ j) \in\ Z_{\infty};\ j \in \{j.\ j \leq n\}]\!] \Longrightarrow$
$(Amin\ n\ f) \leq (f\ j)$
⟨*proof*⟩

lemma *Amax-mem-le*:$[\![\forall j \leq n.\ (f\ j) \in\ Z_{-\infty};\ j \in \{j.\ j \leq n\}]\!] \Longrightarrow$
$(f\ j) \leq (Amax\ n\ f)$
⟨*proof*⟩

lemma *amin-ge1*:$[\![(z::ant) \leq x;\ z \leq y\ ]\!] \Longrightarrow z \leq amin\ x\ y$
⟨*proof*⟩

lemma *amin-gt*:$[\![(z::ant) < x;\ z < y]\!] \Longrightarrow z < amin\ x\ y$
⟨*proof*⟩

lemma *Amin-ge1Tr*:$(\forall j{\leq}(Suc\ n).\ (f\ j) \in Z_{\infty} \wedge z \leq (f\ j)) \longrightarrow$
$z \leq (Amin\ (Suc\ n)\ f)$
⟨*proof*⟩

lemma *Amin-ge1*:$[\![\ \forall j \leq (Suc\ n).\ f\ j \in Z_{\infty};\ \forall j \leq (Suc\ n).\ z \leq (f\ j)]\!] \Longrightarrow$
$z \leq (Amin\ (Suc\ n)\ f)$
⟨*proof*⟩

lemma *amin-trans1*:$[\![x \in Z_{\infty};\ y \in Z_{\infty};\ z \in Z_{\infty};\ z \leq x\ ]\!] \Longrightarrow amin\ z\ y \leq amin\ x\ y$
⟨*proof*⟩

lemma *inf-in-aug-inf*:$\infty\ \in Z_{\infty}$
⟨*proof*⟩

1.8.1 Maximum element of a set of ants

primrec *aasc-seq* :: $[ant\ set,\ ant,\ nat] \Rightarrow ant$
where
aasc-seq-0 : $aasc\text{-}seq\ A\ a\ 0 = a$
| *aasc-seq-Suc* : $aasc\text{-}seq\ A\ a\ (Suc\ n) =$
$(SOME\ b.\ ((b \in A) \wedge (aasc\text{-}seq\ A\ a\ n) < b))$

lemma *aasc-seq-mem*:$[\![a \in A;\ \neg\ (\exists m.\ m{\in}A \wedge (\forall x{\in}A.\ x \leq m))]\!] \Longrightarrow$
$(aasc\text{-}seq\ A\ a\ n) \in A$
⟨*proof*⟩

lemma *aasc-seqn*:$[\![a \in A;\ \neg\ (\exists m.\ m{\in}A \wedge (\forall x{\in}A.\ x \leq m))]\!] \Longrightarrow$
$(aasc\text{-}seq\ A\ a\ n) < (aasc\text{-}seq\ A\ a\ (Suc\ n))$
⟨*proof*⟩

lemma *aasc-seqn1*:$[\![a \in A; \neg\ (\exists\, m.\ m{\in}A \wedge (\forall\, x{\in}A.\ x \leq m))]\!] \Longrightarrow$
$(aasc\text{-}seq\ A\ a\ n) + 1 \leq (aasc\text{-}seq\ A\ a\ (Suc\ n))$
⟨*proof*⟩

lemma *aubs-ex-n-maxTr*:$[\![a \in A; \neg\ (\exists\, m.\ m{\in}A \wedge (\forall\, x{\in}A.\ x \leq m))]\!] \Longrightarrow$
$(a + an\ n) \leq (aasc\text{-}seq\ A\ a\ n)$
⟨*proof*⟩

lemma *aubs-ex-AMax*:$[\![A \subseteq UBset\ (ant\ z);\ A \neq \{\}]\!] \Longrightarrow \exists!m.\ m{\in}A \wedge (\forall\, x{\in}A.\ x \leq m)$
⟨*proof*⟩

definition
AMax :: *ant set* ⇒ *ant* **where**
AMax A = (*THE m*. $m \in A \wedge (\forall\, x{\in}A.\ x \leq m)$)

definition
AMin::*ant set* ⇒ *ant* **where**
AMin A = (*THE m*. $m \in A \wedge (\forall\, x{\in}A.\ m \leq x)$)

definition
rev-o :: *ant* ⇒ *ant* **where**
rev-o x = − *x*

lemma *AMax*:$[\![A \subseteq UBset\ (ant\ z);\ A \neq \{\}]\!] \Longrightarrow$
$(AMax\ A) \in A \wedge (\forall\, x{\in}A.\ x \leq (AMax\ A))$
⟨*proof*⟩

lemma *AMax-mem*:$[\![A \subseteq UBset\ (ant\ z);\ A \neq \{\}]\!] \Longrightarrow (AMax\ A) \in A$
⟨*proof*⟩

lemma *rev-map-nonempty*:$A \neq \{\} \Longrightarrow rev\text{-}o\ `\ A \neq \{\}$
⟨*proof*⟩

lemma *rev-map*:*rev-o* ∈ $LBset\ (ant\ (-z)) \rightarrow UBset\ (ant\ z)$
⟨*proof*⟩

lemma *albs-ex-AMin*:$[\![A \subseteq LBset\ (ant\ z);\ A \neq \{\}]\!] \Longrightarrow \exists!m.\ m{\in}A \wedge (\forall\, x{\in}A.\ m \leq x)$
⟨*proof*⟩

lemma *AMin*:$[\![A \subseteq LBset\ (ant\ z);\ A \neq \{\}]\!] \Longrightarrow$
$(AMin\ A) \in A \wedge (\forall\, x{\in}A.\ (AMin\ A) \leq x)$
⟨*proof*⟩

lemma *AMin-mem*:$[\![A \subseteq LBset\ (ant\ z);\ A \neq \{\}]\!] \Longrightarrow (AMin\ A) \in A$
⟨*proof*⟩

primrec $ASum :: (nat \Rightarrow ant) \Rightarrow nat \Rightarrow ant$
where
$ASum\text{-}0$: $ASum\ f\ 0 = f\ 0$
| $ASum\text{-}Suc$: $ASum\ f\ (Suc\ n) = (ASum\ f\ n) + (f\ (Suc\ n))$

lemma *age-plus*:$[\![0 \leq (a::ant);\ 0 \leq b;\ a + b \leq c]\!] \Longrightarrow a \leq c$
⟨*proof*⟩

lemma *age-diff-le*:$[\![(a::ant) \leq c;\ 0 \leq b]\!] \Longrightarrow a - b \leq c$
⟨*proof*⟩

lemma *adiff-le-adiff*:$a \leq (a'::ant) \Longrightarrow a - b \leq a' - b$
⟨*proof*⟩

lemma *aplus-le-aminus*:$[\![\ a \in\ Z_{-\infty};\ b \in\ Z_{-\infty};\ c \in\ Z_{-\infty};\ -b \in\ Z_{-\infty}]\!] \Longrightarrow$
$((a + b) \leq (c::ant)) = (a \leq c - b)$
⟨*proof*⟩

1.9 Cardinality of sets

cardinality is defined for the finite sets only

lemma *card-eq*:$A = B \Longrightarrow card\ A = card\ B$
⟨*proof*⟩

lemma *card0*:$card\ \{\} = 0$
⟨*proof*⟩

lemma *card-nonzero*:$[\![finite\ A;\ card\ A \neq 0]\!] \Longrightarrow A \neq \{\}$
⟨*proof*⟩

lemma *finite1*:$finite\ \{a\}$
⟨*proof*⟩

lemma *card1*:$card\ \{a\} = 1$
⟨*proof*⟩

lemma *nonempty-card-pos*:$[\![finite\ A;\ A \neq \{\}]\!] \Longrightarrow 0 < card\ A$
⟨*proof*⟩

lemma *nonempty-card-pos1*:$[\![finite\ A;\ A \neq \{\}]\!] \Longrightarrow Suc\ 0 \leq card\ A$
⟨*proof*⟩

lemma *card1-tr0*:$[\![\ finite\ A;\ card\ A = Suc\ 0;\ a \in A\ ]\!] \Longrightarrow \{a\} = A$
⟨*proof*⟩

lemma *card1-tr1*:$(constmap\ \{0::nat\}\ \{x\}) \in \{0\} \rightarrow \{x\} \wedge$
$surj\text{-}to\ (constmap\ \{0::nat\}\ \{x\})\ \{0\}\ \{x\}$
⟨*proof*⟩

lemma *card1-Tr2*:⟦*finite A*; *card A = Suc 0*⟧ ⟹
$\exists f.\ f \in \{0::nat\} \to A \wedge$ *surj-to f* $\{0\}$ *A*
⟨*proof*⟩

lemma *card2*:⟦ *finite A*; $a \in A$; $b \in A$; $a \neq b$ ⟧ ⟹ *Suc* (*Suc 0*) ≤ *card A*
⟨*proof*⟩

lemma *card2-inc-two*:⟦$0 < (n::nat)$; $x \in \{j.\ j \leq n\}$⟧ ⟹
$\exists\, y \in \{j.\ j \leq n\}.\ x \neq y$
⟨*proof*⟩

lemma *Nset2-prep1*:⟦*finite A*; *card A = Suc* (*Suc n*) ⟧ ⟹ $\exists\, x.\ x \in A$
⟨*proof*⟩

lemma *ex-least-set*:⟦$A = \{H.$ *finite H* ∧ *P H*$\}$; $H \in A$⟧ ⟹
$\exists\, K \in A.$ (*LEAST j*. $j \in$ (*card* ‘ *A*)) = *card K*

⟨*proof*⟩

lemma *Nset2-prep2*:$x \in A \Longrightarrow A - \{x\} \cup \{x\} = A$
⟨*proof*⟩

lemma *Nset2-finiteTr*:$\forall A.$ (*finite A* ∧(*card A = Suc n*) ⟶
(∃*f*. $f \in \{i.\ i \leq n\} \to A \wedge$ *surj-to f* $\{i.\ i \leq n\}$ *A*))
⟨*proof*⟩

lemma *Nset2-finite*:⟦ *finite A*; *card A = Suc n*⟧ ⟹
$\exists f.\ f \in \{i.\ i \leq n\} \to A \wedge$ *surj-to f* $\{i.\ i \leq n\}$ *A*
⟨*proof*⟩

lemma *Nset2finite-inj-tr0*:$j \in \{i.\ i \leq (n::nat)\}$ ⟹
card $(\{i.\ i \leq n\} - \{j\}) = n$
⟨*proof*⟩

lemma *Nset2finite-inj-tr1*:⟦ $i \leq (n::nat)$; $j \leq n$; $f\, i = f\, j$; $i \neq j$ ⟧ ⟹
f ‘ $(\{i.\ i \leq n\} - \{j\}) = f$ ‘ $\{i.\ i \leq n\}$
⟨*proof*⟩

lemma *Nset2finite-inj*:⟦*finite A*; *card A = Suc n*; *surj-to f* $\{i.\ i \leq n\}$ *A* ⟧ ⟹
inj-on f $\{i.\ i \leq n\}$
⟨*proof*⟩

definition
zmax :: [*int*, *int*] ⇒ *int* **where**
zmax x y = (*if* $(x \leq y)$ *then y else x*)

primrec $Zmax :: [nat, nat \Rightarrow int] \Rightarrow int$
where
  $Zmax\text{-}0 : Zmax\ 0\ f = f\ 0$
| $Zmax\text{-}Suc : Zmax\ (Suc\ n)\ f = zmax\ (Zmax\ n\ f)\ (f\ (Suc\ n))$

lemma $Zmax\text{-}memTr: f \in \{i.\ i \leq (n::nat)\} \to (UNIV::int\ set) \longrightarrow Zmax\ n\ f \in f\ `\ \{i.\ i \leq n\}$
⟨*proof*⟩

lemma $zmax\text{-}ge\text{-}r: y \leq zmax\ x\ y$
⟨*proof*⟩

lemma $zmax\text{-}ge\text{-}l: x \leq zmax\ x\ y$
⟨*proof*⟩

lemma $Zmax\text{-}geTr: f \in \{j.\ j \leq (n::nat)\} \to (UNIV::int\ set) \longrightarrow (\forall j{\in}\{j.\ j \leq n\}.\ (f\ j) \leq Zmax\ n\ f)$
⟨*proof*⟩

lemma $Zmax\text{-}plus1: f \in \{j.\ j \leq (n::nat)\} \to (UNIV::int\ set) \Longrightarrow ((Zmax\ n\ f) + 1) \notin f\ `\ \{j.\ j \leq n\}$
⟨*proof*⟩

lemma $image\text{-}Nsetn\text{-}card\text{-}pos: 0 < card\ (f\ `\ \{i.\ i \leq (n::nat)\})$
⟨*proof*⟩

lemma *card-image-Nsetn-Suc*
$: [\![f \in \{j.\ j \leq Suc\ n\} \to B;\ f\ (Suc\ n) \notin f\ `\ \{j.\ j \leq n\} ]\!] \Longrightarrow card\ (f\ `\ \{j.\ j \leq Suc\ n\}) - Suc\ 0 = Suc\ (card\ (f\ `\ \{j.\ j \leq n\}) - Suc\ 0)$
⟨*proof*⟩

lemma *slide-surj*:
  ‹*surj-to* (*slide i*) $\{l.\ l \leq (j - i)\}$ (*nset i j*)› **if** ‹$i < j$› **for** $i\ j :: nat$
  ⟨*proof*⟩

lemma $slide\text{-}inj: i < j \Longrightarrow inj\text{-}on\ (slide\ i)\ \{k.\ k \leq (j - i)\}$
⟨*proof*⟩

lemma $card\text{-}nset: i < (j :: nat) \Longrightarrow card\ (nset\ i\ j) = Suc\ (j - i)$
⟨*proof*⟩

lemma $sliden\text{-}hom: i < j \Longrightarrow (sliden\ i) \in nset\ i\ j \to \{k.\ k \leq (j - i)\}$
⟨*proof*⟩

lemma $slide\text{-}sliden: (sliden\ i)\ (slide\ i\ k) = k$
⟨*proof*⟩

lemma *sliden-surj*:$i < j \Longrightarrow$ *surj-to* (*sliden i*) (*nset i j*) {*k*. $k \leq (j - i)$}
⟨*proof*⟩

lemma *sliden-inj*: $i < j \Longrightarrow$ *inj-on* (*sliden i*) (*nset i j*)
⟨*proof*⟩

definition
transpos :: [*nat*, *nat*] ⇒ (*nat* ⇒ *nat*) **where**
transpos i j = (λ*k*. *if k* = *i then j else if k* = *j then i else k*)

lemma *transpos-id*:⟦ $i \leq n$; $j \leq n$; $i \neq j$; $x \in \{k.\ k \leq n\} - \{i, j\}$ ⟧
$\Longrightarrow$ *transpos i j x* = *x*
⟨*proof*⟩

lemma *transpos-id-1*:⟦$i \leq n$; $j \leq n$; $i \neq j$; $x \leq n$; $x \neq i$; $x \neq j$⟧ $\Longrightarrow$
transpos i j x = *x*
⟨*proof*⟩

lemma *transpos-id-2*:$i \leq n \Longrightarrow$ *transpos i n* (*Suc n*) = *Suc n*
⟨*proof*⟩

lemma *transpos-ij-1*:⟦$i \leq n$; $j \leq n$; $i \neq j$ ⟧ $\Longrightarrow$
transpos i j i = *j*
⟨*proof*⟩

lemma *transpos-ij-2*:⟦$i \leq n$; $j \leq n$; $i \neq j$⟧ $\Longrightarrow$ *transpos i j j* = *i*
⟨*proof*⟩

lemma *transpos-hom*:⟦$i \leq n$; $j \leq n$; $i \neq j$⟧ $\Longrightarrow$
(*transpos i j*) ∈ {*i*. $i \leq n$} → {*i*. $i \leq n$}
⟨*proof*⟩

lemma *transpos-mem*:⟦$i \leq n$; $j \leq n$; $i \neq j$; $l \leq n$⟧ $\Longrightarrow$
(*transpos i j l*) $\leq n$
⟨*proof*⟩

lemma *transpos-inj*:⟦$i \leq n$; $j \leq n$; $i \neq j$⟧
$\Longrightarrow$ *inj-on* (*transpos i j*) {*i*. $i \leq n$}
⟨*proof*⟩

lemma *transpos-surjec*:⟦$i \leq n$; $j \leq n$; $i \neq j$⟧
$\Longrightarrow$ *surj-to* (*transpos i j*) {*i*. $i \leq n$} {*i*. $i \leq n$}
⟨*proof*⟩

lemma *comp-transpos*:⟦$i \leq n$; $j \leq n$; $i \neq j$⟧ $\Longrightarrow$
$\forall k \leq n$. (*compose* {*i*. $i \leq n$} (*transpos i j*) (*transpos i j*)) *k* = *k*
⟨*proof*⟩

lemma *comp-transpos-1*:$[\![i \leq n; j \leq n; i \neq j; k \leq n]\!] \Longrightarrow$
$(transpos\ i\ j)\ ((transpos\ i\ j)\ k) = k$
⟨*proof*⟩

lemma *cmp-transpos1*:$[\![i \leq n; j \leq n; i \neq j; k \leq n]\!] \Longrightarrow$
$(cmp\ (transpos\ i\ j)\ (transpos\ i\ j))\ k = k$
⟨*proof*⟩

lemma *cmp-transpos*:$[\![i \leq n; i \neq n; a \leq (Suc\ n)]\!] \Longrightarrow$
$(cmp\ (transpos\ i\ n)\ (cmp\ (transpos\ n\ (Suc\ n))\ (transpos\ i\ n)))\ a =$
$transpos\ i\ (Suc\ n)\ a$
⟨*proof*⟩

lemma *im-Nset-Suc*:$insert\ (f\ (Suc\ n))\ (f\ `\ \{i.\ i \leq n\}) = f\ `\ \{i.\ i \leq (Suc\ n)\}$
⟨*proof*⟩

lemma *Nset-injTr0*:$[\![f \in \{i.\ i \leq (Suc\ n)\} \to \{i.\ i \leq (Suc\ n)\};$
$inj\text{-}on\ f\ \{i.\ i \leq (Suc\ n)\};\ f\ (Suc\ n) = Suc\ n]\!] \Longrightarrow$
$f \in \{i.\ i \leq n\} \to \{i.\ i \leq n\} \wedge inj\text{-}on\ f\ \{i.\ i \leq n\}$
⟨*proof*⟩

lemma *inj-surj*:$[\![f \in \{i.\ i \leq (n::nat)\} \to \{i.\ i \leq n\};$
$inj\text{-}on\ f\ \{i.\ i \leq (n::nat)\}]\!] \Longrightarrow f\ `\ \{i.\ i \leq n\} = \{i.\ i \leq n\}$
⟨*proof*⟩

lemma *Nset-pre-mem*:$[\![f:\{i.\ i \leq (Suc\ n)\} \to \{i.\ i \leq (Suc\ n)\};$
$inj\text{-}on\ f\ \{i.\ i \leq (Suc\ n)\};\ f\ (Suc\ n) = Suc\ n;\ k \leq n]\!] \Longrightarrow f\ k \in \{i.\ i \leq n\}$
⟨*proof*⟩

lemma *Nset-injTr1*:$[\![\ \forall\ l \leq (Suc\ n).\ f\ l \leq (Suc\ n);\ inj\text{-}on\ f\ \{i.\ i \leq (Suc\ n)\};$
$f\ (Suc\ n) = Suc\ n\ ]\!] \Longrightarrow inj\text{-}on\ f\ \{i.\ i \leq n\}$
⟨*proof*⟩

lemma *Nset-injTr2*:$[\![\forall\ l \leq (Suc\ n).\ f\ l \leq (Suc\ n);\ inj\text{-}on\ f\ \{i.\ i \leq (Suc\ n)\};$
$f\ (Suc\ n) = Suc\ n]\!] \Longrightarrow \forall\ l \leq n.\ f\ l \leq n$
⟨*proof*⟩

lemma *TR-inj-inj*:$[\![\forall\ l \leq (Suc\ n).\ f\ l \leq (Suc\ n);\ inj\text{-}on\ f\ \{i.\ i \leq (Suc\ n)\};$
$i \leq (Suc\ n);\ j \leq (Suc\ n);\ i < j\ ]\!] \Longrightarrow$
$inj\text{-}on\ (compose\ \{i.\ i \leq (Suc\ n)\}\ (transpos\ i\ j)\ f)\ \{i.\ i \leq (Suc\ n)\}$
⟨*proof*⟩

lemma *enumeration*:$[\![f \in \{i.\ i \leq (n::nat)\} \to \{i.\ i \leq m\};\ inj\text{-}on\ f\ \{i.\ i \leq n\}]\!]$
$\Longrightarrow\ n \leq m$
⟨*proof*⟩

lemma *enumerate-1*:$[\![\forall\ j \leq (n::nat).\ f\ j \in A;\ \forall\ j \leq (m::nat).\ g\ j \in A;$
$inj\text{-}on\ f\ \{i.\ i \leq n\};\ inj\text{-}on\ g\ \{j.\ j \leq m\};\ f\ `\{j.\ j \leq n\} = A;$
$g\ `\ \{j.\ j \leq m\} = A\ ]\!] \Longrightarrow n = m$

⟨*proof*⟩

definition
ninv :: [*nat*, (*nat* ⇒ *nat*)] ⇒ (*nat* ⇒ *nat*) **where**
ninv n f = (λ*y*∈{*i*. *i* ≤ *n*}. (*SOME x*. (*x* ≤ *n* ∧ *y* = *f x*)))

lemma *ninv-hom*:⟦*f* ∈ {*i*. *i* ≤ *n*} → {*i*. *i* ≤ *n*}; *inj-on f* {*i*. *i* ≤ *n*}⟧ ⟹
ninv n f ∈ {*i*. *i* ≤ *n*} → {*i*. *i* ≤ *n*}
⟨*proof*⟩

lemma *ninv-r-inv*:⟦*f* ∈ {*i*. *i* ≤ (*n*::*nat*)} → {*i*. *i* ≤ *n*}; *inj-on f* {*i*. *i* ≤ *n*};
b ≤ *n*⟧ ⟹ *f* (*ninv n f b*) = *b*
⟨*proof*⟩

lemma *ninv-inj*:⟦*f* ∈ {*i*. *i* ≤ *n*} → {*i*. *i* ≤ *n*}; *inj-on f* {*i*. *i* ≤ *n*}⟧ ⟹
inj-on (*ninv n f*) {*i*. *i* ≤ *n*}
⟨*proof*⟩

1.9.1 Lemmas required in Algebra6.thy

lemma *ge2-zmult-pos*:
2 ≤ *m* ⟹ 0 < *z* ⟹ 1 < *int m* ∗ *z*
⟨*proof*⟩

lemma *zmult-pos-mono*:⟦ (0::*int*) < *w*; *w* ∗ *z* ≤ *w* ∗ *z*′⟧ ⟹ *z* ≤ *z*′
⟨*proof*⟩

lemma *zmult-pos-mono-r*:
⟦(0::*int*) < *w*; *z* ∗ *w* ≤ *z*′ ∗ *w*⟧ ⟹ *z* ≤ *z*′
⟨*proof*⟩

lemma *an-neq-inf*:*an n* ≠ ∞
⟨*proof*⟩

lemma *an-neq-minf*:*an n* ≠ −∞
⟨*proof*⟩

lemma *aeq-mult*:⟦*z* ≠ 0; *a* = *b*⟧ ⟹ *a* ∗ *ant z* = *b* ∗ *ant z*
⟨*proof*⟩

lemma *tna-0*[*simp*]:*tna* 0 = 0
⟨*proof*⟩

lemma *ale-nat-le*:(*an n* ≤ *an m*) = (*n* ≤ *m*)
⟨*proof*⟩

lemma *aless-nat-less*:(*an n* < *an m*) = (*n* < *m*)
⟨*proof*⟩

lemma *apos-natpos*:$[\![a \neq \infty;\ 0 \leq a]\!] \Longrightarrow 0 \leq na\ a$
⟨*proof*⟩

lemma *apos-tna-pos*:$[\![n \neq \infty;\ 0 \leq n]\!] \Longrightarrow 0 \leq tna\ n$
⟨*proof*⟩

lemma *apos-na-pos*:$[\![n \neq \infty;\ 0 \leq n]\!] \Longrightarrow 0 \leq na\ n$
⟨*proof*⟩

lemma *aposs-tna-poss*:$[\![n \neq \infty;\ 0 < n]\!] \Longrightarrow 0 < tna\ n$
⟨*proof*⟩

lemma *aposs-na-poss*:$[\![n \neq \infty;\ 0 < n]\!] \Longrightarrow 0 < na\ n$
⟨*proof*⟩

lemma *nat-0-le*: $0 \leq z \Longrightarrow int\ (nat\ z) = z$
⟨*proof*⟩

lemma *int-eq*:$m = n \Longrightarrow int\ m = int\ n$
⟨*proof*⟩

lemma *box-equation*:$[\![a = b;\ a = c]\!] \Longrightarrow b = c$
⟨*proof*⟩

lemma *aeq-nat-eq*:$[\![n \neq \infty;\ 0 \leq n;\ m \neq \infty;\ 0 \leq m]\!] \Longrightarrow$
$(n = m) = (na\ n = na\ m)$
⟨*proof*⟩

lemma *na-minf*:$na\ (-\infty) = 0$
⟨*proof*⟩

lemma *an-na*:$[\![a \neq \infty;\ 0 \leq a]\!] \Longrightarrow an\ (na\ a) = a$
⟨*proof*⟩

lemma *not-na-le-minf*:$\neg\ (an\ n \leq -\infty\ )$
⟨*proof*⟩

lemma *not-na-less-minf*:$\neg\ (an\ n < -\infty)$
⟨*proof*⟩

lemma *not-na-ge-inf*:$\neg\ \infty \leq (an\ n)$
⟨*proof*⟩

lemma *an-na-le*:$j \leq an\ n \Longrightarrow na\ j \leq n$
⟨*proof*⟩

lemma *aless-neq* :$(x::ant) < y \Longrightarrow x \neq y$
⟨*proof*⟩

Chapter 2

Ordered Set

2.1 Basic Concepts of Ordered Sets

record $'a$ *carrier* =
  carrier :: $'a$ *set*

record $'a$ *Order* = $'a$ *carrier* +
  rel :: ($'a \times 'a$) *set*

locale *Order* =
  fixes *D* (**structure**)
  assumes *closed*: *rel D* $\subseteq$ *carrier D* $\times$ *carrier D*
    and *refl*: $a \in$ *carrier D* $\Longrightarrow$ $(a, a) \in$ *rel D*
    and *antisym*: $[\![a \in$ *carrier D*; $b \in$ *carrier D*; $(a, b) \in$ *rel D*;
        $(b, a) \in$ *rel D*$]\!] \Longrightarrow a = b$
    and *trans*: $[\![a \in$ *carrier D*; $b \in$ *carrier D*; $c \in$ *carrier D*;
        $(a, b) \in$ *rel D*; $(b, c) \in$ *rel D*$]\!] \Longrightarrow (a, c) \in$ *rel D*

definition
  ole :: - $\Rightarrow 'a \Rightarrow 'a \Rightarrow$ *bool* (**infix** $\langle \preceq_1 \rangle$ *60*) **where**
  $a \preceq_D b \longleftrightarrow (a, b) \in$ *rel D*

definition
  oless :: - $\Rightarrow 'a \Rightarrow 'a \Rightarrow$ *bool* (**infix** $\langle \prec_1 \rangle$ *60*) **where**
  $a \prec_D b \equiv a \preceq_D b \wedge a \neq b$

lemma *Order-component*:(*E*::$'a$ *Order*) = $(\!|$ *carrier* = *carrier E*, *rel* = *rel E* $|\!)$
⟨*proof*⟩

lemma *Order-comp-eq*:$[\![$*carrier* (*E*::$'a$ *Order*) = *carrier* (*F*::$'a$ *Order*);
    rel E = *rel F*$]\!] \Longrightarrow E = F$
⟨*proof*⟩

lemma (**in** *Order*) *le-rel*:$[\![a \in \mathit{carrier}\ D;\ b \in \mathit{carrier}\ D]\!] \Longrightarrow$
$(a \preceq b) = ((a,\ b) \in \mathit{rel}\ D)$
⟨*proof*⟩

lemma (**in** *Order*) *less-imp-le*:
$[\![a \in \mathit{carrier}\ D;\ b \in \mathit{carrier}\ D;\ a \prec b\ ]\!] \Longrightarrow a \preceq b$
⟨*proof*⟩

lemma (**in** *Order*) *le-refl*:$a \in \mathit{carrier}\ D \Longrightarrow a \preceq a$
⟨*proof*⟩

lemma (**in** *Order*) *le-antisym*:$[\![a \in \mathit{carrier}\ D;\ b \in \mathit{carrier}\ D;$
$a \preceq b;\ b \preceq a\ ]\!] \Longrightarrow a = b$
⟨*proof*⟩

lemma (**in** *Order*) *le-trans*:$[\![a \in \mathit{carrier}\ D;\ b \in \mathit{carrier}\ D;\ c \in \mathit{carrier}\ D;$
$a \preceq b;\ b \preceq c\ ]\!] \Longrightarrow a \preceq c$
⟨*proof*⟩

lemma (**in** *Order*) *less-trans*:$[\![a \in \mathit{carrier}\ D;\ b \in \mathit{carrier}\ D;\ c \in \mathit{carrier}\ D;$
$a \prec b;\ b \prec c\ ]\!] \Longrightarrow a \prec c$
⟨*proof*⟩

lemma (**in** *Order*) *le-less-trans*:$[\![a \in \mathit{carrier}\ D;\ b \in \mathit{carrier}\ D;\ c \in \mathit{carrier}\ D;$
$a \preceq b;\ b \prec c\ ]\!] \Longrightarrow a \prec c$
⟨*proof*⟩

lemma (**in** *Order*) *less-le-trans*:$[\![a \in \mathit{carrier}\ D;\ b \in \mathit{carrier}\ D;\ c \in \mathit{carrier}\ D;$
$a \prec b;\ b \preceq c\ ]\!] \Longrightarrow a \prec c$
⟨*proof*⟩

lemma (**in** *Order*) *le-imp-less-or-eq*:
$[\![a \in \mathit{carrier}\ D;\ b \in \mathit{carrier}\ D]\!] \Longrightarrow (a \preceq b) = (a \prec b \vee a = b)$
⟨*proof*⟩

lemma (**in** *Order*) *less-neq*: $a \prec b \Longrightarrow a \neq b$
⟨*proof*⟩

lemma (**in** *Order*) *le-neq-less*: $[\![a \preceq b;\ a \neq b]\!] \Longrightarrow a \prec b$
⟨*proof*⟩

lemma (**in** *Order*) *less-irrefl*: $[\![a \in \mathit{carrier}\ D;\ a \prec a]\!] \Longrightarrow C$
⟨*proof*⟩

lemma (**in** *Order*) *less-irrefl'*: $a \in \mathit{carrier}\ D \Longrightarrow \neg\ a \prec a$
⟨*proof*⟩

lemma (**in** *Order*) *less-asym*:

$a \in carrier\ D \Longrightarrow b \in carrier\ D \Longrightarrow a \prec b \Longrightarrow b \prec a \Longrightarrow C$
⟨*proof*⟩

lemma (**in** *Order*) *less-asym'*:
$a \in carrier\ D \Longrightarrow b \in carrier\ D \Longrightarrow a \prec b \Longrightarrow \neg\ b \prec a$
⟨*proof*⟩

lemma (**in** *Order*) *gt-than-any-outside*:$[\![A \subseteq carrier\ D;\ b \in carrier\ D;$
$\forall x{\in}A.\ x \prec b]\!] \Longrightarrow b \notin A$
⟨*proof*⟩

definition
Iod :: - ⇒ *'a set* ⇒ - **where**
Iod D T =
D (|*carrier* := *T*, *rel* := {(*a*, *b*). (*a*, *b*) ∈ *rel D* ∧ *a* ∈ *T* ∧ *b* ∈ *T*}|)

definition
SIod :: *'a Order* ⇒ *'a set* ⇒ *'a Order* **where**
SIod D T = (|*carrier* = *T*, *rel* = {(*a*, *b*). (*a*, *b*) ∈ *rel D* ∧ *a* ∈ *T* ∧ *b* ∈ *T*}|)

lemma (**in** *Order*) *Iod-self*: *D* = *Iod D* (*carrier D*)
⟨*proof*⟩

lemma *SIod-self*:*Order D* ⟹ *D* = *SIod D* (*carrier D*)
⟨*proof*⟩

lemma (**in** *Order*) *Od-carrier*:*carrier* (*D*(|*carrier* := *S*, *rel* := *R*|)) = *S*
⟨*proof*⟩

lemma (**in** *Order*) *Od-rel*:*rel* (*D*(|*carrier* := *S*, *rel* := *R*|)) = *R*
⟨*proof*⟩

lemma (**in** *Order*) *Iod-carrier*:
T ⊆ *carrier D* ⟹ *carrier* (*Iod D T*) = *T*
⟨*proof*⟩

lemma *SIod-carrier*:[|*Order D*; *T* ⊆ *carrier D*|] ⟹ *carrier* (*SIod D T*) = *T*
⟨*proof*⟩

lemma (**in** *Order*) *Od-compare*:(*S* = *S'* ∧ *R* = *R'*) = (*D*(|*carrier* := *S*, *rel* := *R*|)
= *D*(|*carrier* := *S'*, *rel* := *R'*|))
⟨*proof*⟩

lemma (**in** *Order*) *Iod-le*:
[|*T* ⊆ *carrier D*; *a* ∈ *T*; *b* ∈ *T*|] ⟹ (*a* ⪯$_{Iod\ D\ T}$ *b*) = (*a* ⪯ *b*)
⟨*proof*⟩

lemma *SIod-le*:[|*T* ⊆ *carrier D*; *a* ∈ *T*; *b* ∈ *T*|] ⟹
(*a* ⪯$_{SIod\ D\ T}$ *b*) = (*a* ⪯$_D$ *b*)

⟨*proof*⟩

lemma (**in** *Order*) *Iod-less*:
$[\![T \subseteq \mathit{carrier}\ D;\ a \in T;\ b \in T]\!] \Longrightarrow (a \prec_{\mathit{Iod}\ D\ T}\ b) = (a \prec b)$
⟨*proof*⟩

lemma *SIod-less*:$[\![T \subseteq \mathit{carrier}\ D;\ a \in T;\ b \in T]\!] \Longrightarrow$
$(a \prec_{\mathit{SIod}\ D\ T}\ b) = (a \prec_D\ b)$
⟨*proof*⟩

lemma (**in** *Order*) *Iod-Order*:
$T \subseteq \mathit{carrier}\ D \Longrightarrow \mathit{Order}\ (\mathit{Iod}\ D\ T)$
⟨*proof*⟩

lemma *SIod-Order*:$[\![\ \mathit{Order}\ D;\ T \subseteq \mathit{carrier}\ D]\!] \Longrightarrow \mathit{Order}\ (\mathit{SIod}\ D\ T)$
⟨*proof*⟩

lemma (**in** *Order*) *emptyset-Iod*:$\mathit{Order}\ (\mathit{Iod}\ D\ \{\})$
⟨*proof*⟩

lemma (**in** *Order*) *Iod-sub-sub*:
$[\![S \subseteq T;\ T \subseteq \mathit{carrier}\ D]\!] \Longrightarrow \mathit{Iod}\ (\mathit{Iod}\ D\ T)\ S = \mathit{Iod}\ D\ S$
⟨*proof*⟩

lemma *SIod-sub-sub*:
$[\![S \subseteq T;\ T \subseteq \mathit{carrier}\ D]\!] \Longrightarrow \mathit{SIod}\ (\mathit{SIod}\ D\ T)\ S = \mathit{SIod}\ D\ S$
⟨*proof*⟩

lemma *rel-SIod*:$[\![\mathit{Order}\ D;\ \mathit{Order}\ E;\ \mathit{carrier}\ E \subseteq \mathit{carrier}\ D;$
$\forall\, a{\in}\mathit{carrier}\ E.\ \forall\, b{\in}\mathit{carrier}\ E.\ (a \preceq_E\ b) = (a \preceq_D\ b)]\!] \Longrightarrow$
$\mathit{rel}\ E = \mathit{rel}\ (\mathit{SIod}\ D\ (\mathit{carrier}\ E))$
⟨*proof*⟩

lemma *SIod-self-le*:$[\![\mathit{Order}\ D;\ \mathit{Order}\ E;$
$\mathit{carrier}\ E \subseteq \mathit{carrier}\ D;$
$\forall\, a{\in}\mathit{carrier}\ E.\ \forall\, b{\in}\mathit{carrier}\ E.\ (a \preceq_E\ b) = (a \preceq_D\ b)\ ]\!] \Longrightarrow$
$E = \mathit{SIod}\ D\ (\mathit{carrier}\ E)$
⟨*proof*⟩

2.1.1 Total ordering

locale *Torder* = *Order* +
assumes *le-linear*: $[\![a \in \mathit{carrier}\ D;\ b \in \mathit{carrier}\ D]\!] \Longrightarrow$
$a \preceq b \vee b \preceq a$

lemma (**in** *Order*) *Iod-empty-Torder*:$\mathit{Torder}\ (\mathit{Iod}\ D\ \{\})$
⟨*proof*⟩

lemma (**in** *Torder*) *le-cases*:
$[\![a \in \mathit{carrier}\ D;\ b \in \mathit{carrier}\ D;\ (a \preceq b \Longrightarrow C);\ (b \preceq a \Longrightarrow C)]\!] \Longrightarrow C$
⟨*proof*⟩

lemma (**in** *Torder*) *Order*:*Order D*
⟨*proof*⟩

lemma (**in** *Torder*) *less-linear*:
$a \in \mathit{carrier}\ D \Longrightarrow b \in \mathit{carrier}\ D \Longrightarrow a \prec b \vee a = b \vee b \prec a$
⟨*proof*⟩

lemma (**in** *Torder*) *not-le-less*:
$a \in \mathit{carrier}\ D \Longrightarrow b \in \mathit{carrier}\ D \Longrightarrow$
$(\neg\ a \preceq b) = (b \prec a)$
⟨*proof*⟩

lemma (**in** *Torder*) *not-less-le*:
$a \in \mathit{carrier}\ D \Longrightarrow b \in \mathit{carrier}\ D \Longrightarrow$
$(\neg\ a \prec b) = (b \preceq a)$
⟨*proof*⟩

lemma (**in** *Order*) *Iod-not-le-less*:$[\![T \subseteq \mathit{carrier}\ D;\ a \in T;\ b \in T;$
$\mathit{Torder}\ (\mathit{Iod}\ D\ T)]\!] \Longrightarrow (\neg\ a \preceq_{(\mathit{Iod}\ D\ T)}\ b) = b \prec_{(\mathit{Iod}\ D\ T)}\ a$
⟨*proof*⟩

lemma (**in** *Order*) *Iod-not-less-le*:$[\![T \subseteq \mathit{carrier}\ D;\ a \in T;\ b \in T;$
$\mathit{Torder}\ (\mathit{Iod}\ D\ T)]\!] \Longrightarrow (\neg\ a \prec_{(\mathit{Iod}\ D\ T)}\ b) = b \preceq_{(\mathit{Iod}\ D\ T)}\ a$
⟨*proof*⟩

2.1.2 Two ordered sets

definition
Order-Pow :: *'a set* ⇒ *'a set Order* (‹(*po* -)› [*999*] *1000*) **where**
po A =
(|*carrier* = *Pow A*,
rel = {(*X*, *Y*). *X* ∈ *Pow A* ∧ *Y* ∈ *Pow A* ∧ *X* ⊆ *Y*}|)

interpretation *order-Pow*: *Order po A*
⟨*proof*⟩

definition
Order-fs :: *'a set* ⇒ *'b set* ⇒ (*'a set* ∗ (*'a* ⇒ *'b*)) *Order* **where**
Order-fs A B =
(|*carrier* = {*Z*. ∃ *A1 f*. *A1* ∈ *Pow A* ∧ *f* ∈ *A1* → *B* ∧
f ∈ *extensional A1* ∧ *Z* = (*A1*, *f*)},
rel = {*Y*. *Y* ∈ ({*Z*. ∃ *A1 f*. *A1* ∈ *Pow A* ∧ *f* ∈ *A1* → *B* ∧ *f* ∈ *extensional A1*
∧ *Z* = (*A1*, *f*)}) × ({*Z*. ∃ *A1 f*. *A1* ∈ *Pow A* ∧ *f* ∈ *A1* → *B* ∧ *f* ∈ *extensional*
A1
∧ *Z* = (*A1*, *f*)}) ∧ *fst* (*fst Y*) ⊆ *fst* (*snd Y*) ∧

$(\forall\, a{\in}\ (fst\ (fst\ Y)).\ (snd\ (fst\ Y))\ a = (snd\ (snd\ Y))\ a)\}\rrbracket$

lemma *Order-fs*:*Order* (*Order-fs A B*)
⟨*proof*⟩

2.1.3 Homomorphism of ordered sets

definition
ord-inj :: [('*a*, '*m0*) *Order-scheme*, ('*b*, '*m1*) *Order-scheme*,
'*a* ⇒ '*b*] ⇒ *bool* **where**
ord-inj D E f ⟷ *f* ∈ *extensional* (*carrier D*) ∧
f ∈ (*carrier D*) → (*carrier E*) ∧
(*inj-on f* (*carrier D*)) ∧
(∀ *a*∈*carrier D*. ∀ *b*∈*carrier D*. (*a* ≺$_D$ *b*) = ((*f a*) ≺$_E$ (*f b*)))

definition
ord-isom :: [('*a*, '*m0*) *Order-scheme*, ('*b*, '*m1*) *Order-scheme*,
'*a* ⇒ '*b*] ⇒ *bool* **where**
ord-isom D E f ⟷ *ord-inj D E f* ∧
(*surj-to f* (*carrier D*) (*carrier E*))

lemma (**in** *Order*) *ord-inj-func*:⟦*Order E*; *ord-inj D E f*⟧ ⟹
f ∈ *carrier D* → *carrier E*
⟨*proof*⟩

lemma (**in** *Order*) *ord-isom-func*:⟦*Order E*; *ord-isom D E f*⟧ ⟹
f ∈ *carrier D* → *carrier E*
⟨*proof*⟩

lemma (**in** *Order*) *ord-inj-restrict-isom*:⟦*Order E*; *ord-inj D E f*; *T* ⊆ *carrier D*⟧
⟹ *ord-isom* (*Iod D T*) (*Iod E* (*f* ‘ *T*)) (*restrict f T*)
⟨*proof*⟩

lemma *ord-inj-Srestrict-isom*:⟦*Order D*; *Order E*; *ord-inj D E f*; *T* ⊆ *carrier D*⟧
⟹ *ord-isom* (*SIod D T*) (*SIod E* (*f* ‘ *T*)) (*restrict f T*)
⟨*proof*⟩

lemma (**in** *Order*) *id-ord-isom*:*ord-isom D D* (*idmap* (*carrier D*))
⟨*proof*⟩

lemma (**in** *Order*) *ord-isom-bij-to*:⟦*Order E*; *ord-isom D E f*⟧ ⟹
bij-to f (*carrier D*) (*carrier E*)
⟨*proof*⟩

lemma (**in** *Order*) *ord-inj-mem*:⟦*Order E*; *ord-inj D E f*; *a* ∈ *carrier D*⟧ ⟹
(*f a*) ∈ *carrier E*
⟨*proof*⟩

lemma (**in** *Order*) *ord-isom-mem*:⟦*Order E*; *ord-isom D E f*; *a* ∈ *carrier D*⟧ ⟹

$(f\ a) \in carrier\ E$

⟨*proof*⟩

lemma (**in** *Order*) *ord-isom-surj*:⟦*Order E*; *ord-isom D E f*; $b \in carrier\ E$⟧ ⟹
$\exists\, a{\in}carrier\ D.\ b = f\ a$

⟨*proof*⟩

lemma (**in** *Order*) *ord-isom-surj-forall*:⟦*Order E*; *ord-isom D E f*⟧ ⟹
$\forall\, b \in carrier\ E.\ \exists\, a{\in}carrier\ D.\ b = f\ a$

⟨*proof*⟩

lemma (**in** *Order*) *ord-isom-onto*:⟦*Order E*; *ord-isom D E f*⟧ ⟹
$f\ `\ (carrier\ D) = carrier\ E$

⟨*proof*⟩

lemma (**in** *Order*) *ord-isom-inj-on*:⟦*Order E*; *ord-isom D E f*⟧ ⟹
inj-on f (*carrier D*)

⟨*proof*⟩

lemma (**in** *Order*) *ord-isom-inj*:⟦*Order E*; *ord-isom D E f*;
$a \in carrier\ D$; $b \in carrier\ D$⟧ ⟹ $(a = b) = ((f\ a) = (f\ b))$

⟨*proof*⟩

lemma (**in** *Order*) *ord-isom-surj-to*:⟦*Order E*; *ord-isom D E f*⟧ ⟹
surj-to f (*carrier D*) (*carrier E*)

⟨*proof*⟩

lemma (**in** *Order*) *ord-inj-less*:⟦*Order E*; *ord-inj D E f*; $a \in carrier\ D$;
$b \in carrier\ D$⟧ ⟹ $(a \prec_D b) = ((f\ a) \prec_E (f\ b))$

⟨*proof*⟩

lemma (**in** *Order*) *ord-isom-less*:⟦*Order E*; *ord-isom D E f*;
$a \in carrier\ D$; $b \in carrier\ D$⟧ ⟹ $(a \prec_D b) = ((f\ a) \prec_E (f\ b))$

⟨*proof*⟩

lemma (**in** *Order*) *ord-isom-less-forall*:⟦*Order E*; *ord-isom D E f*⟧ ⟹
$\forall\, a \in carrier\ D.\ \forall\ b \in carrier\ D.\ (a \prec_D b) = ((f\ a) \prec_E (f\ b))$

⟨*proof*⟩

lemma (**in** *Order*) *ord-isom-le*:⟦*Order E*; *ord-isom D E f*;
$a \in carrier\ D$; $b \in carrier\ D$⟧ ⟹ $(a \preceq_D b) = ((f\ a) \preceq_E (f\ b))$

⟨*proof*⟩

lemma (**in** *Order*) *ord-isom-le-forall*:⟦*Order E*; *ord-isom D E f*⟧ ⟹
$\forall\, a \in carrier\ D.\ \forall\ b \in carrier\ D.\ (a \preceq b) = ((f\ a) \preceq_E (f\ b))$

⟨*proof*⟩

lemma (**in** *Order*) *ord-isom-convert*:⟦*Order E*; *ord-isom D E f*;
$x \in carrier\ D$; $a \in carrier\ D$⟧ ⟹ $(\forall\, y{\in}carrier\ D.\ (x \prec y \longrightarrow \neg\ y \prec a)) =$

$(\forall\, z{\in}carrier\ E.\ ((f\ x) \prec_E z \longrightarrow \neg\ z \prec_E (f\ a)))$
⟨*proof*⟩

lemma (**in** *Order*) *ord-isom-sym*:⟦*Order E*; *ord-isom D E f*⟧ ⟹
ord-isom E D (*invfun* (*carrier D*) (*carrier E*) *f*)
⟨*proof*⟩

lemma (**in** *Order*) *ord-isom-trans*:⟦*Order E*; *Order F*; *ord-isom D E f*;
ord-isom E F g ⟧ ⟹ *ord-isom D F* (*compose* (*carrier D*) *g f*)
⟨*proof*⟩

definition
ord-equiv :: [-, ('*b*, '*m1*) *Order-scheme*] ⇒ *bool* **where**
ord-equiv D E ⟷ (∃ *f*. *ord-isom D E f*)

lemma (**in** *Order*) *ord-equiv*:⟦*Order E*; *ord-isom D E f*⟧ ⟹ *ord-equiv D E*
⟨*proof*⟩

lemma (**in** *Order*) *ord-equiv-isom*:⟦*Order E*; *ord-equiv D E*⟧ ⟹
∃ *f*. *ord-isom D E f*
⟨*proof*⟩

lemma (**in** *Order*) *ord-equiv-reflex*:*ord-equiv D D*
⟨*proof*⟩

lemma (**in** *Order*) *eq-ord-equiv*:⟦*Order E*; *D* = *E*⟧ ⟹ *ord-equiv D E*
⟨*proof*⟩

lemma (**in** *Order*) *ord-equiv-sym*:⟦*Order E*; *ord-equiv D E* ⟧ ⟹ *ord-equiv E D*
⟨*proof*⟩

lemma (**in** *Order*) *ord-equiv-trans*:⟦*Order E*; *Order F*; *ord-equiv D E*;
ord-equiv E F⟧ ⟹ *ord-equiv D F*
⟨*proof*⟩

lemma (**in** *Order*) *ord-equiv-box*:⟦*Order E*; *Order F*; *ord-equiv D E*;
ord-equiv D F⟧ ⟹ *ord-equiv E F*
⟨*proof*⟩

lemma *SIod-isom-Iod*:⟦*Order D*; *T* ⊆ *carrier D* ⟧ ⟹
ord-isom (*SIod D T*) (*Iod D T*) (λ*x*∈*T*. *x*)
⟨*proof*⟩

definition
minimum-elem :: [- , '*a set*, '*a*] ⇒ *bool* **where**
minimum-elem = (λ*D X a*. *a* ∈ *X* ∧ (∀ *x*∈*X*. *a* ⪯$_D$ *x*))

locale *Worder* = *Torder* +
assumes *ex-minimum*: ∀ *X*. *X* ⊆ (*carrier D*) ∧ *X* ≠ {} ⟶

$(\exists\, x.\ minimum\text{-}elem\ D\ X\ x)$

lemma (**in** *Worder*) *Order*:*Order D*
⟨*proof*⟩

lemma (**in** *Worder*) *Torder*:*Torder D*
⟨*proof*⟩

lemma (**in** *Worder*) *Worder*:*Worder D*
⟨*proof*⟩

lemma (**in** *Worder*) *equiv-isom*:⟦*Worder E*; *ord-equiv D E*⟧ ⟹
$\exists\, f.$ *ord-isom D E f*
⟨*proof*⟩

lemma (**in** *Order*) *minimum-elem-mem*:⟦$X \subseteq$ *carrier D*; *minimum-elem D X a*⟧
⟹ $a \in X$
⟨*proof*⟩

lemma (**in** *Order*) *minimum-elem-unique*:⟦$X \subseteq$ *carrier D*; *minimum-elem D X a1*;
minimum-elem D X a2⟧ ⟹ *a1* = *a2*
⟨*proof*⟩

lemma (**in** *Order*) *compare-minimum-elements*:⟦$S \subseteq$ *carrier D*; $T \subseteq$ *carrier D*;
$S \subseteq T$; *minimum-elem D S s*; *minimum-elem D T t* ⟧ ⟹ $t \preceq s$
⟨*proof*⟩

lemma (**in** *Order*) *minimum-elem-sub*:⟦$T \subseteq$ *carrier D*; $X \subseteq T$⟧
⟹ *minimum-elem D X a* = *minimum-elem* (*Iod D T*) *X a*
⟨*proof*⟩

lemma *minimum-elem-Ssub*:⟦*Order D*; $T \subseteq$ *carrier D*; $X \subseteq T$⟧
⟹ *minimum-elem D X a* = *minimum-elem* (*SIod D T*) *X a*
⟨*proof*⟩

lemma (**in** *Order*) *augmented-set-minimum*:⟦$X \subseteq$ *carrier D*; $a \in$ *carrier D*;
$Y - \{a\} \subseteq X$; $y - \{a\} \neq \{\}$; *minimum-elem* (*Iod D X*) $(Y - \{a\})$ *x*;
$\forall\, x{\in}X.\ x \preceq a$⟧ ⟹ *minimum-elem* (*Iod D* (*insert a X*)) *Y x*
⟨*proof*⟩

lemma *augmented-Sset-minimum*:⟦*Order D*; $X \subseteq$ *carrier D*; $a \in$ *carrier D*;
$Y - \{a\} \subseteq X$; $y - \{a\} \neq \{\}$; *minimum-elem* (*SIod D X*) $(Y - \{a\})$ *x*;
$\forall\, x{\in}X.\ x \preceq_D a$⟧ ⟹ *minimum-elem* (*SIod D* (*insert a X*)) *Y x*
⟨*proof*⟩

lemma (**in** *Order*) *ord-isom-minimum*:⟦*Order E*; *ord-isom D E f*;
$S \subseteq$ *carrier D*; $a \in$ *carrier D*; *minimum-elem D S a*⟧ ⟹
minimum-elem E (*f*‘*S*) (*f a*)

⟨*proof*⟩

lemma (**in** *Worder*) *pre-minimum*:⟦$T \subseteq$ *carrier* D; *minimum-elem* D T t; $s \in$ *carrier* D; $s \prec_D t$ ⟧ $\Longrightarrow \neg\ s \in T$
⟨*proof*⟩

lemma *bex-nonempty-subset*:$\exists\, a.\ a \in A \wedge P\ a \Longrightarrow$
$\{x.\ x \in A \wedge P\ x\} \subseteq A \wedge \{x.\ x \in A \wedge P\ x\} \neq \{\}$
⟨*proof*⟩

lemma (**in** *Worder*) *to-subset*:⟦$T \subseteq$ *carrier* D; *ord-isom* D (*Iod* D T) f⟧ $\Longrightarrow$
$\forall\, a.\ a \in$ *carrier* $D \longrightarrow a \preceq (f\ a)$
⟨*proof*⟩

lemma *to-subsetS*:⟦*Worder* D; $T \subseteq$ *carrier* D; *ord-isom* D (*SIod* D T) f⟧ $\Longrightarrow$
$\forall\, a.\ a \in$ *carrier* $D \longrightarrow a \preceq_D (f\ a)$
⟨*proof*⟩

lemma (**in** *Worder*) *isom-Worder*:⟦*Order* T; *ord-isom* D T f⟧ $\Longrightarrow$ *Worder* T
⟨*proof*⟩

lemma (**in** *Worder*) *equiv-Worder*:⟦*Order* T; *ord-equiv* D T⟧ $\Longrightarrow$ *Worder* T
⟨*proof*⟩

lemma (**in** *Worder*) *equiv-Worder1*:⟦*Order* T; *ord-equiv* T D⟧ $\Longrightarrow$ *Worder* T
⟨*proof*⟩

lemma (**in** *Worder*) *ord-isom-self-id*:*ord-isom* D D $f \Longrightarrow f =$ *idmap* (*carrier* D)
⟨*proof*⟩

lemma (**in** *Worder*) *isom-unique*:⟦*Worder* E; *ord-isom* D E f; *ord-isom* D E g⟧
$\Longrightarrow f = g$
⟨*proof*⟩

definition
segment :: - $\Rightarrow$ $'a \Rightarrow$ $'a$ *set* **where**
segment D a = (*if* $a \notin$ *carrier* D *then carrier* D *else*
$\{x.\ \ x \prec_D a \wedge x \in$ *carrier* $D\}$)

definition
Ssegment :: $'a$ *Order* $\Rightarrow$ $'a \Rightarrow$ $'a$ *set* **where**
Ssegment D a = (*if* $a \notin$ *carrier* D *then carrier* D *else*
$\{x.\ \ x \prec_D a \wedge x \in$ *carrier* $D\}$)

lemma (**in** *Order*) *segment-sub*:*segment* D $a \subseteq$ *carrier* D
⟨*proof*⟩

lemma *Ssegment-sub*:*Ssegment* D $a \subseteq$ *carrier* D
⟨*proof*⟩

lemma (**in** *Order*) *segment-free*:$a \notin$ *carrier* $D \Longrightarrow$
segment D a = *carrier* D
⟨*proof*⟩

lemma *Ssegment-free*:$a \notin$ *carrier* $D \Longrightarrow$
Ssegment D a = *carrier* D
⟨*proof*⟩

lemma (**in** *Order*) *segment-sub-sub*:⟦$S \subseteq$ *carrier* D; $d \in S$⟧ $\Longrightarrow$
segment (*Iod* D S) $d \subseteq$ *segment* D d
⟨*proof*⟩

lemma *Ssegment-sub-sub*:⟦*Order* D; $S \subseteq$ *carrier* D; $d \in S$⟧ $\Longrightarrow$
Ssegment (*SIod* D S) $d \subseteq$ *Ssegment* D d
⟨*proof*⟩

lemma (**in** *Order*) *a-notin-segment*:$a \notin$ *segment* D a
⟨*proof*⟩

lemma *a-notin-Ssegment*:$a \notin$ *Ssegment* D a
⟨*proof*⟩

lemma (**in** *Order*) *Iod-carr-segment*:
carrier (*Iod* D (*segment* D a)) = *segment* D a
⟨*proof*⟩

lemma *SIod-carr-Ssegment*:*Order* $D \Longrightarrow$
carrier (*SIod* D (*Ssegment* D a)) = *Ssegment* D a
⟨*proof*⟩

lemma (**in** *Order*) *segment-inc*:⟦$a \in$ *carrier* D; $b \in$ *carrier* D⟧ $\Longrightarrow$
($a \prec b$) = ($a \in$ *segment* D b)
⟨*proof*⟩

lemma *Ssegment-inc*:⟦*Order* D; $a \in$ *carrier* D; $b \in$ *carrier* D⟧ $\Longrightarrow$
($a \prec_D b$) = ($a \in$ *Ssegment* D b)
⟨*proof*⟩

lemma (**in** *Order*) *segment-inc1*:$b \in$ *carrier* $D \Longrightarrow$
($a \prec b \wedge a \in$ *carrier* D) = ($a \in$ *segment* D b)
⟨*proof*⟩

lemma *Ssegment-inc1*:⟦*Order* D; $b \in$ *carrier* D⟧ $\Longrightarrow$
($a \prec_D b \wedge a \in$ *carrier* D) = ($a \in$ *Ssegment* D b)
⟨*proof*⟩

lemma (**in** *Order*) *segment-inc-if*:⟦$b \in$ *carrier* D;$a \in$ *segment* D b⟧ $\Longrightarrow$
$a \prec b$

⟨*proof*⟩

lemma *Ssegment-inc-if*:⟦*Order D*; $b \in$ *carrier D*; $a \in$ *Ssegment D b*⟧ ⟹
$a \prec_D b$
⟨*proof*⟩

lemma (**in** *Order*) *segment-inc-less*:⟦$W \subseteq$ *carrier D*; $a \in$ *carrier D*;
$y \in W$; $x \in$ *segment* (*Iod D W*) *a*; $y \prec x$⟧ ⟹ $y \in$ *segment* (*Iod D W*) *a*
⟨*proof*⟩

lemma (**in** *Order*) *segment-order-less*:∀ *b*∈*carrier D*. ∀ *x*∈ *segment D b*. ∀ *y*∈ *segment D b*. $(x \prec y) = (x \prec_{(Iod\ D\ (segment\ D\ b))}\ y)$
⟨*proof*⟩

lemma *Ssegment-order-less*:*Order D* ⟹
∀ *b*∈*carrier D*. ∀ *x*∈ *Ssegment D b*. ∀ *y*∈ *Ssegment D b*.
$(x \prec_D y) = (x \prec_{(SIod\ D\ (Ssegment\ D\ b))}\ y)$
⟨*proof*⟩

lemma (**in** *Order*) *segment-order-le*:∀ *b*∈*carrier D*. ∀ *x*∈ *segment D b*.
∀ *y*∈ *segment D b*. $(x \preceq y) = (x \preceq_{(Iod\ D\ (segment\ D\ b))}\ y)$
⟨*proof*⟩

lemma *Ssegment-order-le*:∀ *b*∈*carrier D*. ∀ *x*∈ *Ssegment D b*.
∀ *y*∈ *Ssegment D b*. $(x \preceq_D y) = (x \preceq_{(SIod\ D\ (Ssegment\ D\ b))}\ y)$
⟨*proof*⟩

lemma (**in** *Torder*) *Iod-Torder*:$X \subseteq$ *carrier D* ⟹ *Torder* (*Iod D X*)
⟨*proof*⟩

lemma *SIod-Torder*:⟦*Torder D*; $X \subseteq$ *carrier D*⟧ ⟹ *Torder* (*SIod D X*)
⟨*proof*⟩

lemma (**in** *Order*) *segment-not-inc*:⟦$a \in$ *carrier D*; $b \in$ *carrier D*;
$a \prec b$⟧ ⟹ $b \notin$ *segment D a*
⟨*proof*⟩

lemma *Ssegment-not-inc*:⟦*Order D*; $a \in$ *carrier D*; $b \in$ *carrier D*; $a \prec_D b$⟧ ⟹
$b \notin$ *Ssegment D a*
⟨*proof*⟩

lemma (**in** *Torder*) *segment-not-inc-iff*:⟦$a \in$ *carrier D*; $b \in$ *carrier D*⟧ ⟹
$(a \preceq b) = (b \notin$ *segment D a*)
⟨*proof*⟩

lemma *Ssegment-not-inc-iff*:⟦*Torder D*; $a \in$ *carrier D*; $b \in$ *carrier D*⟧ ⟹
$(a \preceq_D b) = (b \notin$ *Ssegment D a*)
⟨*proof*⟩

lemma (**in** *Torder*) *minimum-segment-of-sub*:⟦$X \subseteq$ *carrier D*;
minimum-elem D (*segment* (*Iod D X*) *d*) *m* ⟧ ⟹ *minimum-elem D X m*
⟨*proof*⟩

lemma (**in** *Torder*) *segment-out*:⟦$a \in$ *carrier D*; $b \in$ *carrier D*;
$a \prec b$⟧ ⟹ *segment* (*Iod D* (*segment D a*)) *b* = *segment D a*
⟨*proof*⟩

lemma (**in** *Torder*) *segment-minimum-minimum*:⟦$X \subseteq$ *carrier D*; $d \in X$;
minimum-elem (*Iod D* (*segment D d*)) ($X \cap$ (*segment D d*)) *m*⟧ ⟹
minimum-elem D X m
⟨*proof*⟩

lemma (**in** *Torder*) *segment-mono*:⟦$a \in$ *carrier D*; $b \in$ *carrier D*⟧ ⟹
($a \prec b$) = (*segment D a* $\subset$ *segment D b*)
⟨*proof*⟩

lemma *Ssegment-mono*:⟦*Torder D*; $a \in$ *carrier D*; $b \in$ *carrier D*⟧ ⟹
($a \prec_D b$) = (*Ssegment D a* $\subset$ *Ssegment D b*)
⟨*proof*⟩

lemma (**in** *Torder*) *segment-le-mono*:⟦$a \in$ *carrier D*; $b \in$ *carrier D*⟧ ⟹
($a \preceq b$) = (*segment D a* $\subseteq$ *segment D b*)
⟨*proof*⟩

lemma *Ssegment-le-mono*:⟦*Torder D*; $a \in$ *carrier D*; $b \in$ *carrier D*⟧ ⟹
($a \preceq_D b$) = (*Ssegment D a* $\subseteq$ *Ssegment D b*)
⟨*proof*⟩

lemma (**in** *Torder*) *segment-inj*:⟦$a \in$ *carrier D*; $b \in$ *carrier D*⟧ ⟹
($a = b$) = (*segment D a* = *segment D b*)
⟨*proof*⟩

lemma *Ssegment-inj*:⟦*Torder D*; $a \in$ *carrier D*; $b \in$ *carrier D*⟧ ⟹
($a = b$) = (*Ssegment D a* = *Ssegment D b*)
⟨*proof*⟩

lemma (**in** *Torder*) *segment-inj-neq*:⟦$a \in$ *carrier D*; $b \in$ *carrier D*⟧ ⟹
($a \neq b$) = (*segment D a* $\neq$ *segment D b*)
⟨*proof*⟩

lemma *Ssegment-inj-neq*:⟦*Torder D*; $a \in$ *carrier D*; $b \in$ *carrier D*⟧ ⟹
($a \neq b$) = (*Ssegment D a* $\neq$ *Ssegment D b*)
⟨*proof*⟩

lemma (**in** *Order*) *segment-inc-psub*:⟦$x \in$ *segment D a*⟧ ⟹
segment D x $\subset$ *segment D a*
⟨*proof*⟩

lemma *Ssegment-inc-psub*:⟦*Order D*; *x* ∈ *Ssegment D a*⟧ ⟹
Ssegment D x ⊂ *Ssegment D a*
⟨*proof*⟩

lemma (**in** *Order*) *segment-segment*:⟦*b* ∈ *carrier D*; *a* ∈ *segment D b*⟧ ⟹
segment (*Iod D* (*segment D b*)) *a* = *segment D a*
⟨*proof*⟩

lemma *Ssegment-Ssegment*:⟦*Order D*; *b* ∈ *carrier D*; *a* ∈ *Ssegment D b*⟧ ⟹
Ssegment (*SIod D* (*Ssegment D b*)) *a* = *Ssegment D a*
⟨*proof*⟩

lemma (**in** *Order*) *Iod-segment-segment*:*a* ∈ *carrier* (*Iod D* (*segment D b*)) ⟹
Iod (*Iod D* (*segment D b*)) (*segment* (*Iod D* (*segment D b*)) *a*) =
Iod D (*segment D a*)
⟨*proof*⟩

lemma *SIod-Ssegment-Ssegment*:⟦*Order D*; *a* ∈ *carrier* (*SIod D* (*Ssegment D b*))⟧
⟹
SIod (*SIod D* (*Ssegment D b*)) (*Ssegment* (*SIod D* (*Ssegment D b*)) *a*) =
SIod D (*Ssegment D a*)
⟨*proof*⟩

lemma (**in** *Order*) *ord-isom-segment-mem*:⟦*Order E*;
ord-isom D E f; *a* ∈ *carrier D*; *x* ∈ *segment D a* ⟧ ⟹
(*f x*) ∈ *segment E* (*f a*)

⟨*proof*⟩

lemma *ord-isom-Ssegment-mem*:⟦*Order D*; *Order E*;
ord-isom D E f; *a* ∈ *carrier D*; *x* ∈ *Ssegment D a*⟧ ⟹
(*f x*) ∈ *Ssegment E* (*f a*)
⟨*proof*⟩

lemma (**in** *Order*) *ord-isom-segment-segment*:⟦*Order E*;
ord-isom D E f; *a* ∈ *carrier D* ⟧ ⟹
ord-isom (*Iod D* (*segment D a*)) (*Iod E* (*segment E* (*f a*)))
(λ*x*∈*carrier* (*Iod D* (*segment D a*)). *f x*)
⟨*proof*⟩

lemma *ord-isom-Ssegment-Ssegment*:⟦*Order D*; *Order E*;
ord-isom D E f; *a* ∈ *carrier D* ⟧ ⟹
ord-isom (*SIod D* (*Ssegment D a*)) (*SIod E* (*Ssegment E* (*f a*)))
(λ*x*∈*carrier* (*SIod D* (*Ssegment D a*)). *f x*)
⟨*proof*⟩

lemma (**in** *Order*) *ord-equiv-segment-segment*:
⟦*Order E*; *ord-equiv D E*; *a* ∈ *carrier D*⟧

$\Longrightarrow \exists\, t{\in}carrier\ E.\ ord\text{-}equiv\ (Iod\ D\ (segment\ D\ a))\ (Iod\ E\ (segment\ E\ t))$

⟨*proof*⟩

lemma *ord-equiv-Ssegment-Ssegment*:
$[\![Order\ D;\ Order\ E;\ ord\text{-}equiv\ D\ E;\ a \in carrier\ D]\!]$
$\Longrightarrow \exists\, t{\in}carrier\ E.\ ord\text{-}equiv\ (SIod\ D\ (Ssegment\ D\ a))\ (SIod\ E\ (Ssegment\ E\ t))$
⟨*proof*⟩

lemma (**in** *Order*) *ord-isom-restricted*:
$[\![Order\ E;\ ord\text{-}isom\ D\ E\ f;\ D1 \subseteq carrier\ D]\!] \Longrightarrow$
$ord\text{-}isom\ (Iod\ D\ D1)\ (Iod\ E\ (f\ `\ D1))\ (\lambda x{\in}D1.\ f\ x)$
⟨*proof*⟩

lemma *ord-isom-restrictedS*:
$[\![Order\ D;\ Order\ E;\ ord\text{-}isom\ D\ E\ f;\ D1 \subseteq carrier\ D]\!] \Longrightarrow$
$ord\text{-}isom\ (SIod\ D\ D1)\ (SIod\ E\ (f\ `\ D1))\ (\lambda x{\in}D1.\ f\ x)$
⟨*proof*⟩

lemma (**in** *Order*) *ord-equiv-induced*:
$[\![Order\ E;\ ord\text{-}isom\ D\ E\ f;\ D1 \subseteq carrier\ D\ ]\!] \Longrightarrow$
$ord\text{-}equiv\ (Iod\ D\ D1)\ (Iod\ E\ (f\ `\ D1))$
⟨*proof*⟩

lemma *ord-equiv-inducedS*:
$[\![Order\ D;\ Order\ E;\ ord\text{-}isom\ D\ E\ f;\ D1 \subseteq carrier\ D\ ]\!] \Longrightarrow$
$ord\text{-}equiv\ (SIod\ D\ D1)\ (SIod\ E\ (f\ `\ D1))$
⟨*proof*⟩

lemma (**in** *Order*) *equiv-induced-by-inj*:$[\![Order\ E;\ ord\text{-}inj\ D\ E\ f;$
$D1 \subseteq carrier\ D]\!] \Longrightarrow\ ord\text{-}equiv\ (Iod\ D\ D1)\ (Iod\ E\ (f\ `\ D1))$
⟨*proof*⟩

lemma *equiv-induced-by-injS*:$[\![Order\ D;\ Order\ E;\ ord\text{-}inj\ D\ E\ f;$
$D1 \subseteq carrier\ D]\!] \Longrightarrow\ ord\text{-}equiv\ (SIod\ D\ D1)\ (SIod\ E\ (f\ `\ D1))$
⟨*proof*⟩

lemma (**in** *Torder*) *le-segment-segment*:$[\![a \in carrier\ D;\ b \in carrier\ D]\!] \Longrightarrow$
$(a \preceq b) = (segment\ (Iod\ D\ (segment\ D\ b))\ a = segment\ D\ a)$
⟨*proof*⟩

lemma *le-Ssegment-Ssegment*:$[\![Torder\ D;\ a \in carrier\ D;\ b \in carrier\ D]\!] \Longrightarrow$
$(a \preceq_D b) = (Ssegment\ (SIod\ D\ (Ssegment\ D\ b))\ a = Ssegment\ D\ a)$
⟨*proof*⟩

lemma (**in** *Torder*) *inc-segment-segment*:$[\![b \in carrier\ D;$
$a \in segment\ D\ b]\!] \Longrightarrow segment\ (Iod\ D\ (segment\ D\ b))\ a = segment\ D\ a$

⟨*proof*⟩

lemma (**in** *Torder*) *segment-segment*:$[\![a \in carrier\ D;\ b \in carrier\ D]\!] \Longrightarrow$
$(segment\ (Iod\ D\ (segment\ D\ b))\ a = segment\ D\ a) =$
$((segment\ D\ a) \subseteq\ (segment\ D\ b))$
⟨*proof*⟩

lemma (**in** *Torder*) *less-in-Iod*:$[\![a \in carrier\ D;\ b \in carrier\ D;\ a \prec b]\!]$
$\Longrightarrow (a \prec b) = (a \in carrier\ (Iod\ D\ (segment\ D\ b)))$
⟨*proof*⟩

definition
$SS :: - \Rightarrow 'a\ set\ Order$ **where**
$SS\ D = (\!|carrier = \{X.\ \exists\, a{\in}carrier\ D.\ X = segment\ D\ a\},\ rel =$
$\{XX.\ XX \in \{X.\ \exists\, a{\in}carrier\ D.\ X = segment\ D\ a\} \times$
$\{X.\ \exists\, a{\in}carrier\ D.\ X = segment\ D\ a\} \wedge ((fst\ XX) \subseteq (snd\ XX))\}\ |\!)$

definition
$segmap$::- $\Rightarrow 'a \Rightarrow 'a\ set$ **where**
$segmap\ D = (\lambda x{\in}(carrier\ D).\ segment\ D\ x)$

lemma *segmap-func*:$segmap\ D \in carrier\ D \rightarrow carrier\ (SS\ D)$
⟨*proof*⟩

lemma (**in** *Worder*) *ord-isom-segmap*: $ord\text{-}isom\ D\ (SS\ D)\ (segmap\ D)$
⟨*proof*⟩

lemma (**in** *Worder*) *nonequiv-segment*:$a \in carrier\ D \Longrightarrow$
$\neg\ ord\text{-}equiv\ D\ (Iod\ D\ (segment\ D\ a))$
⟨*proof*⟩

lemma *nonequiv-Ssegment*:$[\![Worder\ D;\ a \in carrier\ D]\!] \Longrightarrow$
$\neg\ ord\text{-}equiv\ D\ (SIod\ D\ (Ssegment\ D\ a))$
⟨*proof*⟩

lemma (**in** *Worder*) *subset-Worder*: $T \subseteq carrier\ D \Longrightarrow$
$Worder\ (Iod\ D\ T)$
⟨*proof*⟩

lemma *SIod-Worder*:$[\![Worder\ D;\ T \subseteq carrier\ D]\!] \Longrightarrow Worder\ (SIod\ D\ T)$
⟨*proof*⟩

lemma (**in** *Worder*) *segment-Worder*:$Worder\ (Iod\ D\ (segment\ D\ a))$
⟨*proof*⟩

lemma *Ssegment-Worder*:$Worder\ D \Longrightarrow Worder\ (SIod\ D\ (Ssegment\ D\ a))$
⟨*proof*⟩

lemma (**in** *Worder*) *segment-unique1*:⟦$a \in$ *carrier* D; $b \in$ *carrier* D; $a \prec b$⟧ ⟹
¬ *ord-equiv* (*Iod* D (*segment* D b)) (*Iod* D (*segment* D a))
⟨*proof*⟩

lemma *Ssegment-unique1*:⟦*Worder* D; $a \in$ *carrier* D; $b \in$ *carrier* D; $a \prec_D b$⟧ ⟹
¬ *ord-equiv* (*SIod* D (*Ssegment* D b)) (*SIod* D (*Ssegment* D a))
⟨*proof*⟩

lemma (**in** *Worder*) *segment-unique*:⟦$a \in$ *carrier* D; $b \in$ *carrier* D;
ord-equiv (*Iod* D (*segment* D a)) (*Iod* D (*segment* D b)) ⟧ ⟹ $a = b$
⟨*proof*⟩

lemma *Ssegment-unique*:⟦*Worder* D; $a \in$ *carrier* D; $b \in$ *carrier* D;
ord-equiv (*SIod* D (*Ssegment* D a)) (*SIod* D (*Ssegment* D b)) ⟧ ⟹ $a = b$
⟨*proof*⟩

lemma (**in** *Worder*) *subset-segment*:⟦$T \subseteq$ *carrier* D;
$\forall\, b{\in}T.\ \forall x.\ x \prec b \wedge x \in$ *carrier* $D \longrightarrow x \in T$;
minimum-elem D (*carrier* D − T) a⟧ ⟹ T = *segment* D a
⟨*proof*⟩

lemma *subset-Ssegment*:⟦*Worder* D; $T \subseteq$ *carrier* D;
$\forall\, b{\in}T.\ \forall x.\ x \prec_D b \wedge x \in$ *carrier* $D \longrightarrow x \in T$;
minimum-elem D (*carrier* D − T) a⟧ ⟹ T = *Ssegment* D a
⟨*proof*⟩

lemma (**in** *Worder*) *segmentTr*:⟦$T \subseteq$ *carrier* D;
$\forall\, b \in T.\ (\forall x.\ \ (x \prec b \wedge x \in (\text{carrier } D) \longrightarrow x \in T))$⟧ ⟹
$(T =$ *carrier* $D) \vee (\exists\, a.\ a \in (\text{carrier } D) \wedge T =$ *segment* D $a)$
⟨*proof*⟩

lemma *SsegmentTr*:⟦*Worder* D; $T \subseteq$ *carrier* D;
$\forall\, b \in T.\ (\forall x.\ \ (x \prec_D b \wedge x \in (\text{carrier } D) \longrightarrow x \in T))$⟧ ⟹
$(T =$ *carrier* $D) \vee (\exists\, a.\ a \in (\text{carrier } D) \wedge T =$ *Ssegment* D $a)$
⟨*proof*⟩

lemma (**in** *Worder*) *ord-isom-segment-segment*:⟦*Worder* E;
ord-isom D E f; $a \in$ *carrier* D ⟧ ⟹
ord-isom (*Iod* D (*segment* D a)) (*Iod* E (*segment* E (f a)))
($\lambda x{\in}$*carrier* (*Iod* D (*segment* D a)). f x)
⟨*proof*⟩

definition
Tw :: [- , ('*b*, '*m1*) *Order-scheme*] ⇒ '*a* ⇒ '*b* (‹(*2Tw*-,-)› [*60*,*61*]*60*) **where**
$Tw_{D,T}$ = ($\lambda a{\in}$ *carrier* D. *SOME* x. $x{\in}$*carrier* T ∧
ord-equiv (*Iod* D (*segment* D a)) (*Iod* T (*segment* T x)))

lemma (**in** *Worder*) *Tw-func*:⟦*Worder* T;

$\forall\, a{\in}carrier\ D.\ \exists\, b{\in}carrier\ T.\ ord\text{-}equiv\ (Iod\ D\ (segment\ D\ a))$
$(Iod\ T\ (segment\ T\ b))\rrbracket \Longrightarrow Tw_{D,T} \in carrier\ D \to carrier\ T$
⟨*proof*⟩

lemma (**in** *Worder*) *Tw-mem*:⟦*Worder E*; $x \in carrier\ D$;
$\forall\, a{\in}carrier\ D.\ \exists\, b{\in}carrier\ E.\ ord\text{-}equiv\ (Iod\ D\ (segment\ D\ a))$
$(Iod\ E\ (segment\ E\ b))\rrbracket \Longrightarrow (Tw_{D,E})\ x \in carrier\ E$
⟨*proof*⟩

lemma (**in** *Worder*) *Tw-equiv*:⟦*Worder T*;
$\forall\, a{\in}carrier\ D.\ \exists\, b{\in}carrier\ T.\ ord\text{-}equiv\ (Iod\ D\ (segment\ D\ a))$
$(Iod\ T\ (segment\ T\ b))$; $x \in carrier\ D\ \rrbracket \Longrightarrow$
$ord\text{-}equiv\ (Iod\ D\ (segment\ D\ x))\ (Iod\ T\ (segment\ T\ ((Tw_{D,T})\ x)))$
⟨*proof*⟩

lemma (**in** *Worder*) *Tw-inj*:⟦*Worder E*;
$\forall\, a{\in}carrier\ D.\ \exists\, b{\in}carrier\ E.\ \ ord\text{-}equiv\ (Iod\ D\ (segment\ D\ a))$
$(Iod\ E\ (segment\ E\ b))\rrbracket \Longrightarrow inj\text{-}on\ (Tw_{D,E})\ (carrier\ D)$

⟨*proof*⟩

lemma (**in** *Worder*) *Tw-eq-ord-isom*:⟦*Worder E*;
$\forall\, a{\in}carrier\ D.\ \exists\, b{\in}carrier\ E.$
$ord\text{-}equiv\ (Iod\ D\ (segment\ D\ a))\ (Iod\ E\ (segment\ E\ b))$; $a \in carrier\ D$;
$ord\text{-}isom\ (Iod\ D\ (segment\ D\ a))\ (Iod\ E\ (segment\ E\ (Tw\ D\ E\ a)))\ f$;
$x \in segment\ D\ a\ \rrbracket \Longrightarrow f\ x = Tw\ D\ E\ x$
⟨*proof*⟩

lemma (**in** *Worder*) *Tw-ord-injTr*:⟦*Worder E*;
$\forall\, a{\in}carrier\ D.\ \exists\, b{\in}carrier\ E.$
$ord\text{-}equiv\ (Iod\ D\ (segment\ D\ a))\ (Iod\ E\ (segment\ E\ b))$;
$a \in carrier\ D$; $b \in carrier\ D$; $\ a \prec b\rrbracket \Longrightarrow$
$Tw\ D\ E\ a \prec_E (Tw\ D\ E\ b)$
⟨*proof*⟩

lemma (**in** *Worder*) *Tw-ord-inj*:⟦*Worder E*;
$\forall\, a{\in}carrier\ D.\ \exists\, b{\in}carrier\ E.\ ord\text{-}equiv\ (Iod\ D\ (segment\ D\ a))$
$(Iod\ E\ (segment\ E\ b))\rrbracket \Longrightarrow ord\text{-}inj\ D\ E\ (Tw\ D\ E)$
⟨*proof*⟩

lemma (**in** *Worder*) *ord-isom-restricted-by-Tw*:⟦*Worder E*;
$\forall\, a{\in}carrier\ D.\ \exists\, b{\in}carrier\ E.$
$ord\text{-}equiv\ (Iod\ D\ (segment\ D\ a))\ (Iod\ E\ (segment\ E\ b))$;
$D1 \subseteq carrier\ D\rrbracket \Longrightarrow$
$ord\text{-}isom\ (Iod\ D\ D1)\ (Iod\ E\ ((Tw\ D\ E)\ `\ D1))$
$(restrict\ (Tw\ D\ E)\ D1)$
⟨*proof*⟩

lemma (**in** *Worder*) *Tw-segment-segment*:⟦*Worder E*;

$\forall\, a{\in}carrier\ D.\exists\, b{\in}carrier\ E.$
$ord\text{-}equiv\ (Iod\ D\ (segment\ D\ a))\ (Iod\ E\ (segment\ E\ b));\ a \in carrier\ D\rrbracket$
$\Longrightarrow Tw\ D\ E\ `\ (segment\ D\ a) = segment\ E\ (Tw\ D\ E\ a)$
⟨*proof*⟩

lemma (**in** *Worder*) *ord-isom-Tw-segment*:⟦*Worder E*;
$\forall\, a{\in}carrier\ D.\ \exists\, b{\in}carrier\ E.$
$ord\text{-}equiv\ (Iod\ D\ (segment\ D\ a))\ (Iod\ E\ (segment\ E\ b));\ a{\in}carrier\ D\rrbracket \Longrightarrow$
$ord\text{-}isom\ (Iod\ D\ (segment\ D\ a))\ (Iod\ E\ (segment\ E\ (Tw\ D\ E\ a)))$
$(restrict\ (Tw\ D\ E)\ (segment\ D\ a))$
⟨*proof*⟩

lemma (**in** *Worder*) *well-ord-compare1*:⟦*Worder E*;
$\forall\, a{\in}carrier\ D.\ \exists\, b{\in}carrier\ E.$
$ord\text{-}equiv\ (Iod\ D\ (segment\ D\ a))\ (Iod\ E\ (segment\ E\ b))\rrbracket \Longrightarrow$
$(ord\text{-}equiv\ D\ E) \vee (\exists\, c{\in}carrier\ E.\ ord\text{-}equiv\ D\ (Iod\ E\ (segment\ E\ c)))$
⟨*proof*⟩

lemma *bex-nonempty-set*:$\exists\, x \in A.\ P\ x \Longrightarrow \{x.\ x \in A \wedge P\ x\ \} \neq \{\}$
⟨*proof*⟩

lemma *nonempty-set-sub*:$\{x.\ x \in A \wedge P\ x\ \} \neq \{\} \Longrightarrow$
$\{x.\ x \in A \wedge P\ x\} \subseteq A$
⟨*proof*⟩

lemma (**in** *Torder*) *less-minimum*:⟦*minimum-elem D* $\{x.\ x \in carrier\ D \wedge P\ x\}$ *d*⟧
$\Longrightarrow \forall\, a.\ (((a \prec d) \wedge a \in carrier\ D) \longrightarrow\ \neg\ (P\ a))$
⟨*proof*⟩

lemma (**in** *Torder*) *segment-minimum-empty*:⟦$X \subseteq carrier\ D$; $d \in X$⟧ $\Longrightarrow$
$(minimum\text{-}elem\ D\ X\ d) = (segment\ (Iod\ D\ X)\ d = \{\})$
⟨*proof*⟩

end

theory *Algebra2*
imports *Algebra1*
begin

lemma (**in** *Order*) *less-and-segment*:$b \in carrier\ D \Longrightarrow$
$(\forall\, a.((a \prec b \wedge a \in carrier\ D) \longrightarrow (Q\ a))) =$
$(\forall\, a{\in}carrier\ (Iod\ D\ (segment\ D\ b)).(Q\ a))$
⟨*proof*⟩

lemma (**in** *Worder*) *Word-compare2*:⟦*Worder E*;
$\neg\ (\forall\, a{\in}carrier\ D.\ \exists\, b{\in}carrier\ E.\ ord\text{-}equiv\ (Iod\ D\ (segment\ D\ a))$
$(Iod\ E\ (segment\ E\ b)))\rrbracket \Longrightarrow$
$\exists\, c{\in}carrier\ D.\ ord\text{-}equiv\ (Iod\ D\ (segment\ D\ c))\ \ E$

⟨*proof*⟩

lemma (**in** *Worder*) *Worder-equiv*:⟦*Worder E*;
∀ *a*∈*carrier D*. ∃ *b*∈*carrier E*. *ord-equiv* (*Iod D* (*segment D a*))
(*Iod E* (*segment E b*));
∀ *c*∈*carrier E*. ∃ *d*∈*carrier D*. *ord-equiv* (*Iod E* (*segment E c*))
(*Iod D* (*segment D d*))⟧ ⟹ *ord-equiv D E*
⟨*proof*⟩

lemma (**in** *Worder*) *Worder-equiv1*:⟦*Worder E*; ¬ *ord-equiv D E*⟧ ⟹
¬ ((∀ *a*∈*carrier D*. ∃ *b*∈*carrier E*.
ord-equiv (*Iod D* (*segment D a*)) (*Iod E* (*segment E b*))) ∧
(∀ *c*∈*carrier E*. ∃ *d*∈*carrier D*.
ord-equiv (*Iod E* (*segment E c*)) (*Iod D* (*segment D d*))))
⟨*proof*⟩

lemma (**in** *Worder*) *Word-compare*:*Worder E* ⟹
(∃ *a*∈*carrier D*. *ord-equiv* (*Iod D* (*segment D a*)) *E*) ∨ *ord-equiv D E* ∨
(∃ *b*∈*carrier E*. *ord-equiv D* (*Iod E* (*segment E b*)))
⟨*proof*⟩

lemma (**in** *Worder*) *Word-compareTr1*:⟦*Worder E*;
∃ *a*∈*carrier D*. *ord-equiv* (*Iod D* (*segment D a*)) *E*; *ord-equiv D E* ⟧ ⟹
False
⟨*proof*⟩

lemma (**in** *Worder*) *Word-compareTr2*:⟦*Worder E*; *ord-equiv D E*;
∃ *b*∈*carrier E*. *ord-equiv D* (*Iod E* (*segment E b*))⟧ ⟹ *False*
⟨*proof*⟩

lemma (**in** *Worder*) *Word-compareTr3*:⟦*Worder E*;
∃ *b*∈*carrier E*. *ord-equiv D* (*Iod E* (*segment E b*));
∃ *a*∈*carrier D*. *ord-equiv* (*Iod D* (*segment D a*)) *E*⟧ ⟹ *False*
⟨*proof*⟩

lemma (**in** *Worder*) *subset-equiv-segment*:*S* ⊆ *carrier D* ⟹
ord-equiv D (*Iod D S*) ∨
(∃ *a*∈*carrier D*. *ord-equiv* (*Iod D S*) (*Iod D* (*segment D a*)))
⟨*proof*⟩

definition
ordinal-number :: *'a Order* ⇒ *'a Order set* **where**
ordinal-number S = {*X*. *Worder X* ∧ *ord-equiv X S*}

definition
ODnums :: *'a Order set set* **where**
ODnums = {*X*. ∃ *S*. *Worder S* ∧ *X* = *ordinal-number S*}

definition

ODord :: [*'a Order set*, *'a Order set*] ⇒ *bool* (**infix** ‹⊏› *60*) **where**
X ⊏ *Y* ⟷ (∃ *x* ∈ *X*. ∃ *y* ∈ *Y*. (∃ *c*∈*carrier y*. *ord-equiv x* (*Iod y* (*segment y c*))))

definition
ODord-le :: [*'a Order set*, *'a Order set*] ⇒ *bool* (**infix** ‹⊑› *60*) **where**
X ⊑ *Y* ⟷ *X* = *Y* ∨ *ODord X Y*

definition
ODrel :: (((*'a Order*) *set*) ∗ ((*'a Order*) *set*)) *set* **where**
ODrel = {*Z*. *Z* ∈ *ODnums* × *ODnums* ∧ *ODord-le* (*fst Z*) (*snd Z*)}

definition
ODnods :: (*'a Order set*) *Order* **where**
ODnods = (|*carrier* = *ODnums*, *rel* = *ODrel* |)

lemma *Worder-ord-equivTr*:⟦*Worder S*; *Worder T*⟧ ⟹
ord-equiv S T = (∃ *f*. *ord-isom S T f*)
⟨*proof*⟩

lemma *Worder-ord-isom-mem*:⟦*Worder S*; *Worder T*; *ord-isom S T f*; *a* ∈ *carrier S*⟧
⟹ *f a* ∈ *carrier T*
⟨*proof*⟩

lemma *Worder-refl*:*Worder S* ⟹ *ord-equiv S S*
⟨*proof*⟩

lemma *Worder-sym*:⟦*Worder S*; *Worder T*; *ord-equiv S T* ⟧ ⟹ *ord-equiv T S*
⟨*proof*⟩

lemma *Worder-trans*:⟦*Worder S*; *Worder T*; *Worder U*; *ord-equiv S T*; *ord-equiv T U*⟧ ⟹ *ord-equiv S U*
⟨*proof*⟩

lemma *ordinal-inc-self*:*Worder S* ⟹ *S* ∈ *ordinal-number S*
⟨*proof*⟩

lemma *ordinal-number-eq*:⟦*Worder D*; *Worder E*⟧ ⟹
(*ord-equiv D E*) = (*ordinal-number D* = *ordinal-number E*)
⟨*proof*⟩

lemma *mem-ordinal-number-equiv*:⟦*Worder D*;
X ∈ *ordinal-number D*⟧ ⟹ *ord-equiv X D*
⟨*proof*⟩

lemma *mem-ordinal-number-Worder*:⟦*Worder D*;
X ∈ *ordinal-number D*⟧ ⟹ *Worder X*
⟨*proof*⟩

lemma *mem-ordinal-number-Worder1*:$[\![x \in ODnums;\ X \in x]\!] \Longrightarrow Worder\ X$
⟨*proof*⟩

lemma *mem-ODnums-nonempty*:$X \in ODnums \Longrightarrow \exists\, x.\ x \in X$
⟨*proof*⟩

lemma *carr-ODnods*:$carrier\ ODnods = ODnums$
⟨*proof*⟩

lemma *ordinal-number-mem-carrier-ODnods*:
$Worder\ D \Longrightarrow ordinal\text{-}number\ D \in carrier\ ODnods$
⟨*proof*⟩

lemma *ordinal-number-mem-ODnums*:
$Worder\ D \Longrightarrow ordinal\text{-}number\ D \in ODnums$
⟨*proof*⟩

lemma *ODordTr1*: $[\![Worder\ D;\ Worder\ E]\!] \Longrightarrow$
$(ODord\ (ordinal\text{-}number\ D)\ (ordinal\text{-}number\ E)) =$
$(\exists\, b{\in}\ carrier\ E.\ ord\text{-}equiv\ D\ (Iod\ E\ (segment\ E\ b)))$
⟨*proof*⟩

lemma *ODord*: $[\![Worder\ D;\ d \in carrier\ D]\!] \Longrightarrow$
$ODord\ (ordinal\text{-}number\ (Iod\ D\ (segment\ D\ d)))\ (ordinal\text{-}number\ D)$
⟨*proof*⟩

lemma *ord-less-ODord*:$[\![Worder\ D;\ c \in carrier\ D;\ d \in carrier\ D;$
$a = ordinal\text{-}number\ (Iod\ D\ (segment\ D\ c));$
$b = ordinal\text{-}number\ (Iod\ D\ (segment\ D\ d))]\!] \Longrightarrow$
$c \prec_D d = \ a \sqsubset b$
⟨*proof*⟩

lemma *ODord-le-ref*:$[\![X \in ODnums;\ Y \in ODnums;\ ODord\text{-}le\ X\ Y;\ Y \sqsubseteq X\ ]\!] \Longrightarrow$
$X = Y$
⟨*proof*⟩

lemma *ODord-le-trans*:$[\![X \in ODnums;\ Y \in ODnums;\ Z \in ODnums;\ X \sqsubseteq Y;\ Y \sqsubseteq Z\ ]\!]$
$\Longrightarrow\ X \sqsubseteq Z$
⟨*proof*⟩

lemma *ordinal-numberTr1*: $X \in carrier\ ODnods \Longrightarrow \exists\, D.\ Worder\ D \wedge D \in X$
⟨*proof*⟩

lemma *ordinal-numberTr1-1*: $X \in ODnums \Longrightarrow \exists\, D.\ Worder\ D \wedge D \in X$
⟨*proof*⟩

lemma *ordinal-numberTr1-2*:$[\![x \in ODnums;\ S \in x;\ T \in x]\!] \Longrightarrow$

ord-equiv S T

⟨*proof*⟩

lemma *ordinal-numberTr2*:⟦*Worder D*; *x* = *ordinal-number D*⟧ ⟹
D ∈ *x*

⟨*proof*⟩

lemma *ordinal-numberTr3*:⟦*Worder D*; *Worder F*; *ord-equiv D F*;
x = *ordinal-number D*⟧ ⟹ *x* = *ordinal-number F*

⟨*proof*⟩

lemma *ordinal-numberTr4*:⟦*Worder D*; *X* ∈ *carrier ODnods*; *D* ∈ *X* ⟧ ⟹
X = *ordinal-number D*

⟨*proof*⟩

lemma *ordinal-numberTr5*:⟦*x* ∈ *ODnums*; *D* ∈ *x*⟧ ⟹ *x* = *ordinal-number D*

⟨*proof*⟩

lemma *ordinal-number-ord*:⟦ *X* ∈ *carrier ODnods*; *Y* ∈ *carrier ODnods*⟧ ⟹
ODord X Y ∨ *X* = *Y* ∨ *ODord Y X*

⟨*proof*⟩

lemma *ODnum-subTr*:⟦*Worder D*; *x* = *ordinal-number D*; *y* ∈*ODnums*; *y* ⊏ *x*; *Y*
∈ *y*⟧
⟹ ∃ *c*∈*carrier D*. *ord-equiv Y* (*Iod D* (*segment D c*))

⟨*proof*⟩

lemma *ODnum-segmentTr*:⟦*Worder D*; *x* = *ordinal-number D*; *y* ∈*ODnums*; *y* ⊏
x⟧ ⟹
∃ *c*. *c*∈*carrier D* ∧ (∀ *Y*∈*y*. *ord-equiv Y* (*Iod D* (*segment D c*)))

⟨*proof*⟩

lemma *ODnum-segmentTr1*:⟦*Worder D*; *x* = *ordinal-number D*; *y* ∈ *ODnums*; *y*
⊏ *x*⟧
⟹ ∃ *c*. *c* ∈ *carrier D* ∧ (*y* = *ordinal-number* (*Iod D* (*segment D c*)))

⟨*proof*⟩

lemma *ODnods-less*:⟦*x* ∈ *carrier ODnods*; *y* ∈ *carrier ODnods*⟧ ⟹
x ≺$_{ODnods}$ *y* = *x* ⊏ *y*

⟨*proof*⟩

lemma *ODord-less-not-eq*:⟦*x* ∈ *carrier ODnods*; *y* ∈ *carrier ODnods*; *x* ⊏ *y*⟧ ⟹
x ≠ *y*

⟨*proof*⟩

lemma *not-ODord*:⟦*a* ∈ *ODnums*; *b* ∈ *ODnums*; *a* ⊏ *b*⟧ ⟹ ¬ (*b* ⊑ *a*)

⟨*proof*⟩

lemma *Order-ODnods*:*Order ODnods*

⟨*proof*⟩

lemma *Torder-ODnods*:*Torder ODnods*
⟨*proof*⟩

definition
ODNmap :: *'a Order* ⇒ (*'a Order*) *set* ⇒ *'a* **where**
ODNmap D y = (*SOME z*. (*z* ∈ *carrier D* ∧
(∀ *Y*∈*y*. *ord-equiv Y* (*Iod D* (*segment D z*)))))

lemma *ODNmap-mem*:⟦*Worder D*; *x* = *ordinal-number D*; *y* ∈ *ODnums*; *ODord y x*⟧ ⟹
ODNmap D y ∈ *carrier D* ∧
(∀ *Y*∈*y*. *ord-equiv Y* (*Iod D* (*segment D* (*ODNmap D y*))))
⟨*proof*⟩

lemma *ODNmapTr1*:⟦*Worder D*; *x* = *ordinal-number D*; *y* ∈ *ODnums*; *ODord y x*⟧ ⟹
y = *ordinal-number* (*Iod D* (*segment D* (*ODNmap D y*)))
⟨*proof*⟩

lemma *ODNmap-self*:⟦*Worder D*; *c* ∈ *carrier D*;
a = *ordinal-number* (*Iod D* (*segment D c*))⟧ ⟹ *ODNmap D a* = *c*
⟨*proof*⟩

lemma *ODord-ODNmap-less*:⟦*Worder D*; *c* ∈ *carrier D*;
a = *ordinal-number* (*Iod D* (*segment D c*)); *d* ∈ *carrier D*;
b = *ordinal-number* (*Iod D* (*segment D d*)); *a* ⊏ *b*⟧ ⟹
ODNmap D a ≺$_D$ (*ODNmap D b*)
⟨*proof*⟩

lemma *ODNmap-mem1*: ⟦*Worder D*; *y* ∈ *segment ODnods* (*ordinal-number D*)⟧
⟹ *ODNmap D y* ∈ *carrier D*
⟨*proof*⟩

lemma *ODnods-segment-inc-ODord*:⟦*Worder D*;
y ∈ *segment ODnods* (*ordinal-number D*)⟧ ⟹ *ODord y* (*ordinal-number D*)
⟨*proof*⟩

lemma *restict-ODNmap-func*:⟦*Worder D*; *x* = *ordinal-number D*⟧ ⟹
restrict (*ODNmap D*) (*segment ODnods* (*ordinal-number D*))
∈ *segment ODnods* (*ordinal-number D*) → *carrier D*
⟨*proof*⟩

lemma *ODNmap-ord-isom*:⟦*Worder D*; *x* = *ordinal-number D*⟧ ⟹
ord-isom (*Iod ODnods* (*segment ODnods x*)) *D*
(λ*x*∈(*carrier* (*Iod ODnods* (*segment ODnods x*))). (*ODNmap D x*))
⟨*proof*⟩

lemma *ODnum-equiv-segment*:$[\![Worder\ D;\ x = ordinal\text{-}number\ D]\!] \Longrightarrow$
$ord\text{-}equiv\ (Iod\ ODnods\ (segment\ ODnods\ x))\ D$
⟨*proof*⟩

lemma *ODnods-sub-carrier*:$S \subseteq ODnums \Longrightarrow carrier\ (Iod\ ODnods\ S) = S$
⟨*proof*⟩

lemma *ODnum-sub-well-ordered*:$S \subseteq ODnums \Longrightarrow Worder\ (Iod\ ODnods\ S)$
⟨*proof*⟩

2.2 Pre elements

definition
$ExPre :: - \Rightarrow 'a \Rightarrow bool$ **where**
$ExPre\ D\ a \longleftrightarrow (\exists x.\ x \in carrier\ D \wedge x \prec_D a$
$\wedge \neg (\exists y \in carrier\ D.\ x \prec_D y \wedge y \prec_D a))$

definition
$Pre :: [-, 'a] \Rightarrow 'a$ **where**
$Pre\ D\ a = (SOME\ x.\ x \in carrier\ D \wedge$
$x \prec_D a \wedge$
$\neg (\exists y \in carrier\ D.\ x \prec_D y \wedge y \prec_D a))$

lemma (**in** *Order*) *Pre-mem*:$[\![a \in carrier\ D;\ ExPre\ D\ a]\!] \Longrightarrow$
$Pre\ D\ a \in carrier\ D$
⟨*proof*⟩

lemma (**in** *Order*) *Not-ExPre*:$a \in carrier\ D \Longrightarrow \neg\ ExPre\ (Iod\ D\ \{a\})\ a$
⟨*proof*⟩

lemma (**in** *Worder*) *UniquePre*:$[\![a \in carrier\ D;\ ExPre\ D\ a;$
$a1 \in carrier\ D \wedge a1 \prec a \wedge \neg (\exists y \in carrier\ D.\ (a1 \prec y \wedge y \prec a))\ ]\!] \Longrightarrow$
$Pre\ D\ a = a1$
⟨*proof*⟩

lemma (**in** *Order*) *Pre-element*:$[\![a \in carrier\ D;\ ExPre\ D\ a]\!] \Longrightarrow$
$Pre\ D\ a \in carrier\ D \wedge (Pre\ D\ a) \prec a \wedge$
$\neg (\exists y \in carrier\ D.\ ((Pre\ D\ a) \prec y \wedge y \prec a))$
⟨*proof*⟩

lemma (**in** *Order*) *Pre-in-segment*:$[\![a \in carrier\ D;\ ExPre\ D\ a]\!] \Longrightarrow$
$Pre\ D\ a \in segment\ D\ a$
⟨*proof*⟩

lemma (**in** *Worder*) *segment-forall*:$[\![a \in segment\ D\ b;\ b \in carrier\ D;$
$x \in segment\ D\ b;\ x \prec a;\ \forall y \in segment\ D\ b.\ x \prec y \longrightarrow \neg\ y \prec a]\!] \Longrightarrow$
$\forall y \in carrier\ D.\ x \prec y \longrightarrow \neg\ y \prec a$

⟨*proof*⟩

lemma (**in** *Worder*) *segment-Expre*:$a \in$ *segment* $D\ b \Longrightarrow$
ExPre (*Iod* D (*segment* $D\ b$)) $a =$ *ExPre* $D\ a$
⟨*proof*⟩

lemma (**in** *Worder*) *Pre-segment*:⟦$a \in$ *segment* $D\ b$;
ExPre (*Iod* D (*segment* $D\ b$)) a⟧ $\Longrightarrow$
ExPre $D\ a \wedge$ *Pre* $D\ a =$ *Pre* (*Iod* D (*segment* $D\ b$)) a
⟨*proof*⟩

lemma (**in** *Worder*) *Pre2segment*:⟦$a \in$ *carrier* D; $b \in$ *carrier* D; $b \prec a$;
ExPre $D\ b$⟧ $\Longrightarrow$ *ExPre* (*Iod* D (*segment* $D\ a$)) b
⟨*proof*⟩

lemma (**in** *Worder*) *ord-isom-Pre1*:⟦*Worder* E; $a \in$ *carrier* D; *ExPre* $D\ a$;
ord-isom $D\ E\ f$⟧ $\Longrightarrow$ *ExPre* E ($f\ a$)
⟨*proof*⟩

lemma (**in** *Worder*) *ord-isom-Pre11*:⟦*Worder* E; $a \in$ *carrier* D; *ord-isom* $D\ E\ f$⟧
$\Longrightarrow$ *ExPre* $D\ a =$ *ExPre* E ($f\ a$)
⟨*proof*⟩

lemma (**in** *Worder*) *ord-isom-Pre2*:⟦*Worder* E; $a \in$ *carrier* D; *ExPre* $D\ a$;
ord-isom $D\ E\ f$⟧ $\Longrightarrow f$ (*Pre* $D\ a$) = *Pre* E ($f\ a$)
⟨*proof*⟩

2.3 Transfinite induction

lemma (**in** *Worder*) *transfinite-induction*:⟦*minimum-elem* D (*carrier* D) *s0*; P *s0*;
$\forall\, t{\in}$*carrier* D. (($\forall\, u{\in}$ *segment* $D\ t$. $P\ u$) $\longrightarrow P\ t$)⟧ $\Longrightarrow \forall\, x{\in}$*carrier* D. $P\ x$
⟨*proof*⟩

2.4 *Ordered-set2*. Lemmas to prove Zorn's lemma.

definition
adjunct-ord ::[- , $'a$] ⇒ - **where**
adjunct-ord $D\ a = D$ (|*carrier* := *carrier* $D \cup \{a\}$,
rel := {(x,y). $(x,\ y) \in$ *rel* $D \vee$
$(x \in$ (*carrier* $D \cup \{a\}$) $\wedge\ y = a)$}|)

lemma (**in** *Order*) *carrier-adjunct-ord*:
carrier (*adjunct-ord* $D\ a$) = *carrier* $D \cup \{a\}$
⟨*proof*⟩

lemma (**in** *Order*) *Order-adjunct-ord*:$a \notin$ *carrier* $D \Longrightarrow$
Order (*adjunct-ord* $D\ a$)
⟨*proof*⟩

lemma (**in** *Order*) *adjunct-ord-large-a*:$[\![$*Order D*; $a \notin$ *carrier D*$]\!] \Longrightarrow$
$\forall x{\in}$*carrier D*. $x \prec_{\textit{adjunct-ord D a}} a$
⟨*proof*⟩

lemma *carr-Ssegment-adjunct-ord*:$[\![$*Order D*; $a \notin$ *carrier D*$]\!] \Longrightarrow$
carrier D = (*Ssegment* (*adjunct-ord D a*) *a*)
⟨*proof*⟩

lemma (**in** *Order*) *adjunct-ord-selfD*:$a \notin$ *carrier D* $\Longrightarrow$
D = *Iod* (*adjunct-ord D a*) (*carrier D*)
⟨*proof*⟩

lemma *Ssegment-adjunct-ord*:$[\![$*Order D*; $a \notin$ *carrier D*$]\!] \Longrightarrow$
D = *SIod* (*adjunct-ord D a*) (*Ssegment* (*adjunct-ord D a*) *a*)
⟨*proof*⟩

lemma (**in** *Order*) *Torder-adjunction*:$[\![X \subseteq$ *carrier D*; $a \in$ *carrier D*;
$\forall x{\in}X.\ x \preceq a$; *Torder* (*Iod D X*)$]\!] \Longrightarrow$ *Torder* (*Iod D* ($X \cup \{a\}$))
⟨*proof*⟩

lemma *Torder-Sadjunction*:$[\![$*Order D*; $X \subseteq$ *carrier D*; $a \in$ *carrier D*;
$\forall x{\in}X.\ x \preceq_D a$; *Torder* (*SIod D X*)$]\!] \Longrightarrow$ *Torder* (*SIod D* ($X \cup \{a\}$))
⟨*proof*⟩

lemma (**in** *Torder*) *Torder-adjunct-ord*:$a \notin$ *carrier D* $\Longrightarrow$
Torder (*adjunct-ord D a*)
⟨*proof*⟩

lemma (**in** *Order*) *well-ord-adjunction*:$[\![X \subseteq$ *carrier D*; $a \in$ *carrier D*;
$\forall x{\in}X.\ x \preceq a$; *Worder* (*Iod D X*)$]\!] \Longrightarrow$ *Worder* (*Iod D* ($X \cup \{a\}$))
⟨*proof*⟩

lemma *well-ord-Sadjunction*:$[\![$*Order D*; $X \subseteq$ *carrier D*; $a \in$ *carrier D*;
$\forall x{\in}X.\ x \preceq_D a$; *Worder* (*SIod D X*)$]\!] \Longrightarrow$ *Worder* (*SIod D* ($X \cup \{a\}$))
⟨*proof*⟩

lemma (**in** *Worder*) *Worder-adjunct-ord*:$a \notin$ *carrier D* $\Longrightarrow$
Worder (*adjunct-ord D a*)
⟨*proof*⟩

2.5 Zorn's lemma

definition
Chain :: - ⇒ *'a set* ⇒ *bool* **where**
Chain D C ⟷ $C \subseteq$ *carrier D* ∧ *Torder* (*Iod D C*)

definition
upper-bound :: [-, *'a set*, *'a*] ⇒ *bool*

$(\langle(3ub_1/\ -/\ -)\rangle\ [100,101]100)$ **where**
$ub_D\ S\ b \longleftrightarrow b \in carrier\ D \wedge (\forall s{\in}S.\ s \preceq_D b)$

definition
inductive-set :: - ⇒ *bool* **where**
inductive-set $D \longleftrightarrow (\forall C.\ (Chain\ D\ C \longrightarrow (\exists b.\ ub_D\ C\ b)))$

definition
maximal-element :: [-, 'a] ⇒ *bool* $(\langle(maximal_1/\ -)\rangle\ [101]100)$ **where**
$maximal_D\ m \longleftrightarrow m \in carrier\ D \wedge (\forall b{\in}carrier\ D.\ m \preceq_D b \longrightarrow m = b)$

definition
upper-bounds::[-, 'a *set*] ⇒ 'a *set* **where**
upper-bounds $D\ H = \{x.\ ub_D\ H\ x\}$

definition
Sup :: [-, 'a *set*] ⇒ 'a **where**
Sup $D\ X = (THE\ x.\ minimum\text{-}elem\ D\ (upper\text{-}bounds\ D\ X)\ x)$

definition
S-inductive-set :: - ⇒ *bool* **where**
S-inductive-set $D \longleftrightarrow (\forall C.\ Chain\ D\ C \longrightarrow$
$(\exists x{\in}carrier\ D.\ minimum\text{-}elem\ D\ (upper\text{-}bounds\ D\ C)\ x))$

lemma (**in** *Order*) *mem-upper-bounds*:$[\![X \subseteq carrier\ D;\ b \in carrier\ D;$
$\forall x{\in}X.\ x \preceq b]\!] \Longrightarrow ub\ X\ b$
⟨*proof*⟩

lemma (**in** *Order*) *Torder-Chain*:$[\![X \subseteq carrier\ D;\ Torder\ (Iod\ D\ X)]\!]$
$\Longrightarrow Chain\ D\ X$
⟨*proof*⟩

lemma (**in** *Order*) *Chain-Torder*:$Chain\ D\ X \Longrightarrow$
$Torder\ (Iod\ D\ X)$
⟨*proof*⟩

lemma (**in** *Order*) *Chain-sub*:$Chain\ D\ X \Longrightarrow X \subseteq carrier\ D$
⟨*proof*⟩

lemma (**in** *Order*) *Chain-sub-Chain*:$[\![Chain\ D\ X;\ Y \subseteq X\ ]\!] \Longrightarrow Chain\ D\ Y$
⟨*proof*⟩

lemma (**in** *Order*) *upper-bounds-sub*:$X \subseteq carrier\ D \Longrightarrow$
$upper\text{-}bounds\ D\ X \subseteq carrier\ D$
⟨*proof*⟩

lemma (**in** *Order*) *Sup*:$[\![X \subseteq carrier\ D;\ minimum\text{-}elem\ D\ (upper\text{-}bounds\ D\ X)\ a]\!]$
$\Longrightarrow Sup\ D\ X = a$

⟨*proof*⟩

lemma (**in** *Worder*) *Sup-mem*:⟦$X \subseteq$ *carrier D*; $\exists\, b.$ *ub X b*⟧ ⟹
Sup D X ∈ *carrier D*
⟨*proof*⟩

lemma (**in** *Order*) *S-inductive-sup*:⟦*S-inductive-set D*; *Chain D X*⟧ ⟹
minimum-elem D (*upper-bounds D X*) (*Sup D X*)
⟨*proof*⟩

lemma (**in** *Order*) *adjunct-Chain*:⟦*Chain D X*; *b* ∈ *carrier D*; $\forall\, x{\in}X.\ x \preceq b$⟧ ⟹
Chain D (*insert b X*)
⟨*proof*⟩

lemma (**in** *Order*) *S-inductive-sup-mem*:⟦*S-inductive-set D*; *Chain D X*⟧ ⟹
Sup D X ∈ *carrier D*
⟨*proof*⟩

lemma (**in** *Order*) *S-inductive-Sup-min-bounds*:⟦*S-inductive-set D*; *Chain D X*;
ub X b⟧ ⟹ *Sup D X* $\preceq$ *b*
⟨*proof*⟩

lemma (**in** *Order*) *S-inductive-sup-bound*:⟦*S-inductive-set D*; *Chain D X*⟧ ⟹
$\forall\, x{\in}X.\ x \preceq$ (*Sup D X*)
⟨*proof*⟩

lemma (**in** *Order*) *S-inductive-Sup-in-ChainTr*:
⟦*S-inductive-set D*; *Chain D X*; *c* ∈ *carrier* (*Iod D* (*insert* (*Sup D X*) *X*));
Sup D X ∉ *X*;
$\forall\, y{\in}$*carrier* (*Iod D* (*insert* (*Sup D X*) *X*)).
$c \prec_{Iod\ D\ (insert\ (Sup\ D\ X)\ X)}\ y \longrightarrow \neg\ y \prec_{Iod\ D\ (insert\ (Sup\ D\ X)\ X)}\ Sup\ D\ X$⟧ ⟹
c ∈ *upper-bounds D X*
⟨*proof*⟩

lemma (**in** *Order*) *S-inductive-Sup-in-Chain*:⟦*S-inductive-set D*; *Chain D X*;
ExPre (*Iod D* (*insert* (*Sup D X*) *X*)) (*Sup D X*)⟧ ⟹ *Sup D X* ∈ *X*
⟨*proof*⟩

lemma (**in** *Order*) *S-inductive-bounds-compare*:⟦*S-inductive-set D*; *Chain D X1*;
Chain D X2; *X1* $\subseteq$ *X2*⟧ ⟹ *upper-bounds D X2* $\subseteq$ *upper-bounds D X1*
⟨*proof*⟩

lemma (**in** *Order*) *S-inductive-sup-compare*:⟦*S-inductive-set D*; *Chain D X1*;
Chain D X2; *X1* $\subseteq$ *X2*⟧ ⟹ *Sup D X1* $\preceq$ *Sup D X2*
⟨*proof*⟩

definition
Wa :: [-, *'a set*, *'a* ⇒ *'a*, *'a*] ⇒ *bool* **where**

$Wa\ D\ W\ g\ a \longleftrightarrow W \subseteq carrier\ D \wedge Worder\ (Iod\ D\ W) \wedge a \in W \wedge (\forall x{\in}W.\ a \preceq_D x) \wedge$
$(\forall x{\in}W.\ (if\ (ExPre\ (Iod\ D\ W)\ x)\ then\ \ g\ (Pre\ (Iod\ D\ W)\ x) = x\ else$
$(if\ a \neq x\ then\ Sup\ D\ (segment\ (Iod\ D\ W)\ x) = x\ else\ a = a)))$

definition
$WWa :: [-, 'a \Rightarrow 'a, 'a] \Rightarrow 'a\ set\ set$ **where**
$WWa\ D\ g\ a = \{W.\ Wa\ D\ W\ g\ a\}$

lemma (**in** *Order*) *mem-of-WWa*:$[\![W \subseteq carrier\ D;\ Worder\ (Iod\ D\ W);\ a \in W;$
$(\forall x{\in}W.\ a \preceq x);$
$(\forall x{\in}W.\ (if\ (ExPre\ (Iod\ D\ W)\ x)\ then\ \ g\ (Pre\ (Iod\ D\ W)\ x) = x\ else$
$(if\ a \neq x\ then\ Sup\ D\ (segment\ (Iod\ D\ W)\ x) = x\ else\ a = a)))]\!] \Longrightarrow$
$W \in WWa\ D\ g\ a$
⟨*proof*⟩

lemma (**in** *Order*) *mem-WWa-then*:$W \in WWa\ D\ g\ a \Longrightarrow W \subseteq carrier\ D \wedge$
$Worder\ (Iod\ D\ W) \wedge a \in W \wedge (\forall x{\in}W.\ a \preceq x) \wedge$
$(\forall x{\in}W.\ (if\ (ExPre\ (Iod\ D\ W)\ x)\ then\ \ g\ (Pre\ (Iod\ D\ W)\ x) = x\ else$
$(if\ a \neq x\ then\ Sup\ D\ (segment\ (Iod\ D\ W)\ x) = x\ else\ a = a)))$
⟨*proof*⟩

lemma (**in** *Order*) *mem-wwa-Worder*:$W \in WWa\ D\ g\ a \Longrightarrow Worder\ (Iod\ D\ W)$
⟨*proof*⟩

lemma (**in** *Order*) *mem-WWa-sub-carrier*:$W \in WWa\ D\ g\ a \Longrightarrow W \subseteq carrier\ D$
⟨*proof*⟩

lemma (**in** *Order*) *Union-WWa-sub-carrier*:$\bigcup\ (WWa\ D\ g\ a) \subseteq carrier\ D$
⟨*proof*⟩

lemma (**in** *Order*) *mem-WWa-inc-a*:$W \in WWa\ D\ g\ a \Longrightarrow a \in W$
⟨*proof*⟩

lemma (**in** *Order*) *mem-WWa-Chain*:$W \in WWa\ D\ g\ a \Longrightarrow Chain\ D\ W$
⟨*proof*⟩

lemma (**in** *Order*) *Sup-adjunct-Sup*:$[\![S\text{-}inductive\text{-}set\ D;$
$f \in carrier\ D \to carrier\ D;\ a \in carrier\ D;\ \forall x{\in}carrier\ D.\ x \preceq f\ x;$
$W \in WWa\ D\ f\ a;\ Sup\ D\ W \notin W]\!]$
$\Longrightarrow Sup\ D\ (insert\ (Sup\ D\ W)\ W) = Sup\ D\ W$

⟨*proof*⟩

lemma (**in** *Order*) *BNTr1*:$a \in carrier\ D \Longrightarrow Worder\ (Iod\ D\ \{a\})$
⟨*proof*⟩

lemma (**in** *Order*) *BNTr2*:$[\![f \in carrier\ D \to carrier\ D;\ \ a \in carrier\ D;$
$\forall x{\in}carrier\ D.\ x \preceq (f\ x)]\!] \Longrightarrow \{a\} \in WWa\ D\ f\ a$

⟨*proof*⟩

lemma (**in** *Order*) *BNTr2-1*:⟦*f* ∈ *carrier D* → *carrier D*; *a* ∈ *carrier D*;
∀ *x*∈*carrier D*. *x* ≼ (*f x*); *W* ∈ *WWa D f a*⟧ ⟹ ∀ *x*∈*W*. *a* ≼ *x*
⟨*proof*⟩

lemma (**in** *Order*) *BNTr3*:⟦*f* ∈ *carrier D* → *carrier D*; *a* ∈ *carrier D*;
∀ *x*∈*carrier D*. *x* ≼ (*f x*); *W* ∈ *WWa D f a*⟧ ⟹ *minimum-elem* (*Iod D W*)
W a

⟨*proof*⟩

lemma (**in** *Order*) *Adjunct-segment-sub*:⟦*S-inductive-set D*; *Chain D X*⟧ ⟹
segment (*Iod D* (*insert* (*Sup D X*) *X*)) (*Sup D X*) ⊆ *X*
⟨*proof*⟩

lemma (**in** *Order*) *Adjunct-segment-eq*:⟦*S-inductive-set D*; *Chain D X*;
Sup D X ∉ *X*⟧ ⟹
segment (*Iod D* (*insert* (*Sup D X*) *X*)) (*Sup D X*) = *X*
⟨*proof*⟩

definition
fixp :: ['*a* ⇒ '*a*, '*a*] ⇒ *bool* **where**
fixp f a ⟷ *f a* = *a*

lemma (**in** *Order*) *fixp-same*:⟦*W1* ⊆ *carrier D*; *W2* ⊆ *carrier D*; *t* ∈ *W1*;
b ∈ *carrier D*; *ord-isom* (*Iod D W1*) (*Iod* (*Iod D W2*) (*segment* (*Iod D W2*) *b*))
g;
∀ *u*∈*segment* (*Iod D W1*) *t*. *fixp g u*⟧ ⟹
segment (*Iod D W1*) *t* = *segment* (*Iod D W2*) (*g t*)

⟨*proof*⟩

lemma (**in** *Order*) *BNTr4-1*:⟦*f* ∈ *carrier D* → *carrier D*; *a* ∈ *carrier D*;
b ∈ *carrier D*; ∀ *x*∈*carrier D*. *x* ≼ (*f x*); *W1* ∈ *WWa D f a*; *W2* ∈ *WWa D f*
a;
ord-isom (*Iod D W1*) (*Iod D* (*segment* (*Iod D W2*) *b*)) *g*⟧ ⟹
∀ *x*∈*W1*. *g x* = *x*
⟨*proof*⟩

lemma (**in** *Order*) *BNTr4-2*:⟦*f* ∈ *carrier D* → *carrier D*; *a* ∈ *carrier D*;
b ∈ *carrier D*; ∀ *x*∈*carrier D*. *x* ≼ (*f x*); *W1* ∈ *WWa D f a*; *W2* ∈ *WWa D*
f a;
ord-equiv (*Iod D W1*) (*Iod D* (*segment* (*Iod D W2*) *b*))⟧ ⟹
W1 = *segment* (*Iod D W2*) *b*
⟨*proof*⟩

lemma (**in** *Order*) *BNTr4*:⟦*f* ∈ *carrier D* → *carrier D*; *a* ∈ *carrier D*;
∀ *x*∈*carrier D*. *x* ≼ (*f x*); *W1* ∈ *WWa D f a*; *W2* ∈ *WWa D f a*;

$\exists\, b{\in}carrier\ D.\ ord\text{-}equiv\ (Iod\ D\ W1)\ (Iod\ D\ (segment\ (Iod\ D\ W2)\ b))]\!] \Longrightarrow$
$W1 \subseteq W2$
⟨*proof*⟩

lemma (**in** *Order*) *Iod-same*:$A = B \Longrightarrow Iod\ D\ A = Iod\ D\ B$
⟨*proof*⟩

lemma (**in** *Order*) *eq-ord-equivTr*:$[\![ord\text{-}equiv\ D\ E;\ E = F]\!] \Longrightarrow ord\text{-}equiv\ D\ F$
⟨*proof*⟩

lemma (**in** *Order*) *BNTr5*:$[\![f \in carrier\ D \to carrier\ D;\ a \in carrier\ D;$
$\forall\, x{\in}carrier\ D.\ x \preceq (f\ x);\ W1 \in WWa\ D\ f\ a;\ W2 \in WWa\ D\ f\ a;$
$ord\text{-}equiv\ (Iod\ D\ W1)\ (Iod\ D\ W2)]\!] \Longrightarrow W1 \subseteq W2$
⟨*proof*⟩

lemma (**in** *Order*) *BNTr6*:$[\![f \in carrier\ D \to carrier\ D;\ a \in carrier\ D;$
$\forall\, x{\in}carrier\ D.\ x \preceq (f\ x);\ W1 \in WWa\ D\ f\ a;\ W2 \in WWa\ D\ f\ a;\ W1 \subset W2]\!] \Longrightarrow$
$(\exists\, b{\in}carrier\ (Iod\ D\ W2).\ ord\text{-}equiv\ (Iod\ D\ W1)\ (Iod\ D\ (segment\ (Iod\ D\ W2)\ b)))$
⟨*proof*⟩

lemma (**in** *Order*) *BNTr6-1*:$[\![f \in carrier\ D \to carrier\ D;\ a \in carrier\ D;$
$\forall\, x{\in}carrier\ D.\ x \preceq (f\ x);\ W1 \in WWa\ D\ f\ a;\ W2 \in WWa\ D\ f\ a;\ W1 \subset W2]\!] \Longrightarrow$
$(\exists\, b{\in}carrier\ (Iod\ D\ W2).\ W1 = (segment\ (Iod\ D\ W2)\ b))$
⟨*proof*⟩

lemma (**in** *Order*) *BNTr7*:$[\![f \in carrier\ D \to carrier\ D;\ a \in carrier\ D;$
$\forall\, x{\in}carrier\ D.\ x \preceq (f\ x);\ W1 \in WWa\ D\ f\ a;\ W2 \in WWa\ D\ f\ a]\!] \Longrightarrow$
$W1 \subseteq W2 \vee W2 \subseteq W1$
⟨*proof*⟩

lemma (**in** *Order*) *BNTr7-1*:$[\![f \in carrier\ D \to carrier\ D;\ a \in carrier\ D;$
$\forall\, x{\in}carrier\ D.\ x \preceq f\ x;\ x \in W;\ W \in WWa\ D\ f\ a;\ xa \in \bigcup\ (WWa\ D\ f\ a);$
$xa \prec_{Iod\ D\ (\bigcup\ (WWa\ D\ f\ a))}\ x]\!] \Longrightarrow xa \in W$
⟨*proof*⟩

lemma (**in** *Order*) *BNTr7-1-1*:$[\![f \in carrier\ D \to carrier\ D;\ a \in carrier\ D;$
$\forall\, x{\in}carrier\ D.\ x \preceq f\ x;\ x \in W;\ W \in WWa\ D\ f\ a;\ xa \in \bigcup\ (WWa\ D\ f\ a);$
$xa \prec x]\!] \Longrightarrow xa \in W$
⟨*proof*⟩

lemma (**in** *Order*) *BNTr7-2*: $[\![f \in carrier\ D \to carrier\ D;\ a \in carrier\ D;$
$\forall\, x{\in}carrier\ D.\ x \preceq f\ x;\ x \in \bigcup(WWa\ D\ f\ a);\ ExPre\ (Iod\ D\ (\bigcup(WWa\ D\ f\ a)))$
$x\ ]\!]$
$\Longrightarrow \forall\, W{\in}WWa\ D\ f\ a.\ (x \in W \longrightarrow ExPre\ (Iod\ D\ W)\ x\ )$
⟨*proof*⟩

lemma (**in** *Order*) *BNTr7-3*: $[\![f \in carrier\ D \to carrier\ D;\ a \in carrier\ D;$
$\forall\, x{\in}carrier\ D.\ x \preceq f\ x;\ x \in \bigcup(WWa\ D\ f\ a);\ ExPre\ (Iod\ D\ (\bigcup(WWa\ D\ f\ a)))$
$x\ ]\!]$

$\Longrightarrow \forall\, W{\in}WWa\ D\ f\ a.\ (x \in W \longrightarrow Pre\ (Iod\ D\ (\bigcup(WWa\ D\ f\ a)))\ x = Pre\ (Iod\ D\ W)\ x)$
⟨*proof*⟩

lemma (**in** *Order*) *BNTr7-4*:⟦$f \in carrier\ D \to carrier\ D$; $a \in carrier\ D$;
$\forall\, x{\in}carrier\ D.\ x \preceq f\ x$; $x \in W$; $W \in WWa\ D\ f\ a$⟧ $\Longrightarrow$
$ExPre\ (Iod\ D\ (\bigcup(WWa\ D\ f\ a)))\ x = ExPre\ (Iod\ D\ W)\ x$
⟨*proof*⟩

lemma (**in** *Order*) *BNTr7-5*: ⟦$f \in carrier\ D \to carrier\ D$; $a \in carrier\ D$;
$\forall\, x{\in}carrier\ D.\ x \preceq f\ x$; $x \in W$; $W \in WWa\ D\ f\ a$⟧
$\Longrightarrow (segment\ (Iod\ D\ (\bigcup(WWa\ D\ f\ a)))\ x) = segment\ (Iod\ D\ W)\ x$
⟨*proof*⟩

lemma (**in** *Order*) *BNTr7-6*:⟦$f \in carrier\ D \to carrier\ D$;
$a \in carrier\ D$; $\forall\, x{\in}carrier\ D.\ x \preceq (f\ x)$⟧ $\Longrightarrow a \in \bigcup(WWa\ D\ f\ a)$
⟨*proof*⟩

lemma (**in** *Order*) *BNTr7-7*:⟦*S-inductive-set D*; $f \in carrier\ D \to carrier\ D$;
$a \in carrier\ D$; $\forall\, x{\in}carrier\ D.\ x \preceq (f\ x)$; $\exists\, xa.\ Wa\ D\ xa\ f\ a \wedge x \in xa$⟧ $\Longrightarrow$
$x \in \bigcup(WWa\ D\ f\ a)$
⟨*proof*⟩

lemma (**in** *Order*) *BNTr7-8*:⟦*S-inductive-set D*; $f \in carrier\ D \to carrier\ D$; $a \in carrier\ D$; $\forall\, x{\in}carrier\ D.\ x \preceq (f\ x)$; $\exists\, xa.\ Wa\ D\ xa\ f\ a \wedge x \in xa$⟧ $\Longrightarrow x \in carrier\ D$
⟨*proof*⟩

lemma (**in** *Order*) *BNTr7-9*:⟦$f \in carrier\ D \to carrier\ D$; $a \in carrier\ D$;
$\forall\, x{\in}carrier\ D.\ x \preceq (f\ x)$; $x \in \bigcup(WWa\ D\ f\ a)$ ⟧ $\Longrightarrow x \in carrier\ D$
⟨*proof*⟩

lemma (**in** *Order*) *BNTr7-10*:⟦*S-inductive-set D*; $f \in carrier\ D \to carrier\ D$;
$a \in carrier\ D$; $\forall\, x{\in}carrier\ D.\ x \preceq (f\ x)$; $W \in WWa\ D\ f\ a$; $Sup\ D\ W \notin W$⟧
$\Longrightarrow$
$\neg\ ExPre\ (Iod\ D\ (insert\ (Sup\ D\ W)\ W))\ (Sup\ D\ W)$
⟨*proof*⟩

lemma (**in** *Order*) *BNTr7-11*:⟦*S-inductive-set D*; $f \in carrier\ D \to carrier\ D$;
$a \in carrier\ D$; $b \in carrier\ D$; $\forall\, x{\in}carrier\ D.\ x \preceq f\ x$; $W \in WWa\ D\ f\ a$;
$\forall\, x{\in}W.\ x \preceq b$; $x \in W$⟧ $\Longrightarrow$
$ExPre\ (Iod\ D\ (insert\ b\ W))\ x = ExPre\ (Iod\ D\ W)\ x$
⟨*proof*⟩

lemma (**in** *Order*) *BNTr7-12*:⟦*S-inductive-set D*; $f \in carrier\ D \to carrier\ D$;
$a \in carrier\ D$; $b \in carrier\ D$; $\forall\, x{\in}carrier\ D.\ x \preceq f\ x$; $W \in WWa\ D\ f\ a$;
$\forall\, x{\in}W.\ x \preceq b$; $x \in W$; $ExPre\ (Iod\ D\ W)\ x$⟧ $\Longrightarrow$
$Pre\ (Iod\ D\ (insert\ b\ W))\ x = Pre\ (Iod\ D\ W)\ x$

⟨*proof*⟩

lemma (**in** *Order*) *BNTr7-13*:⟦*S-inductive-set D*; $f \in$ *carrier D* $\to$ *carrier D*;
$a \in$ *carrier D*; $b \in$ *carrier D*; $\forall x \in$ *carrier D*. $x \preceq f\ x$; $W \in$ *WWa D f a*;
$\forall x \in W.\ x \preceq b$; $x \in W$⟧ $\Longrightarrow$
(*segment* (*Iod D* (*insert b W*)) *x*) = *segment* (*Iod D W*) *x*
⟨*proof*⟩

lemma (**in** *Order*) *BNTr7-14*:⟦*S-inductive-set D*; $f \in$ *carrier D* $\to$ *carrier D*;
$a \in$ *carrier D*; $\forall x \in$ *carrier D*. $x \preceq (f\ x)$; $W \in$ *WWa D f a*⟧ $\Longrightarrow$
(*insert* (*Sup D W*) *W*) $\in$ *WWa D f a*
⟨*proof*⟩

lemma (**in** *Order*) *BNTr7-15*:⟦*S-inductive-set D*; $f \in$ *carrier D* $\to$ *carrier D*;
$a \in$ *carrier D*; $\forall x \in$ *carrier D*. $x \preceq (f\ x)$; $W \in$ *WWa D f a*;
f (*Sup D W*) $\neq$ *Sup D W*⟧ $\Longrightarrow$
ExPre (*Iod D* (*insert* (f (*Sup D W*)) (*insert* (*Sup D W*) *W*))) (f (*Sup D W*))
⟨*proof*⟩

lemma (**in** *Order*) *BNTr7-16*:⟦*S-inductive-set D*; $f \in$ *carrier D* $\to$ *carrier D*;
$a \in$ *carrier D*; $\forall x \in$ *carrier D*. $x \preceq (f\ x)$; $W \in$ *WWa D f a*;
f (*Sup D W*) $\neq$ (*Sup D W*)⟧ $\Longrightarrow$
Pre (*Iod D* (*insert* (f (*Sup D W*)) (*insert* (*Sup D W*) *W*))) (f (*Sup D W*)) =
(*Sup D W*)
⟨*proof*⟩

lemma (**in** *Order*) *BNTr7-17*:⟦*S-inductive-set D*; $f \in$ *carrier D* $\to$ *carrier D*;
$a \in$ *carrier D*; $\forall x \in$ *carrier D*. $x \preceq (f\ x)$; $W \in$ *WWa D f a*⟧ $\Longrightarrow$
(*insert* (f (*Sup D W*)) (*insert* (*Sup D W*) *W*)) $\in$ *WWa D f a*
⟨*proof*⟩

lemma (**in** *Order*) *BNTr8*:⟦$f \in$ *carrier D* $\to$ *carrier D*; $a \in$ *carrier D*;
$\forall x \in$ *carrier D*. $x \preceq (f\ x)$⟧ $\Longrightarrow$ $\bigcup$ (*WWa D f a*) $\in$ (*WWa D f a*)
⟨*proof*⟩

lemma (**in** *Order*) *BNTr10*:⟦*S-inductive-set D*; $f \in$ *carrier D* $\to$ *carrier D*;
$a \in$ *carrier D*; $\forall x \in$ *carrier D*. $x \preceq (f\ x)$⟧ $\Longrightarrow$
(*Sup D* ($\bigcup$(*WWa D f a*))) $\in$ ($\bigcup$(*WWa D f a*))
⟨*proof*⟩

lemma (**in** *Order*) *BNTr11*:⟦*S-inductive-set D*; $f \in$ *carrier D* $\to$ *carrier D*;
$a \in$ *carrier D*; $\forall x \in$ *carrier D*. $x \preceq (f\ x)$⟧ $\Longrightarrow$
f (*Sup D* ($\bigcup$(*WWa D f a*))) = (*Sup D* ($\bigcup$(*WWa D f a*)))
⟨*proof*⟩

lemma (**in** *Order*) *Bourbaki-Nakayama*:⟦*S-inductive-set D*;

$f \in carrier\ D \to carrier\ D;\ a \in carrier\ D;\ \forall\, x{\in}carrier\ D.\ x \preceq (f\ x)\rrbracket \Longrightarrow$
$\exists\, x0{\in}carrier\ D.\ f\ x0 = x0$
⟨*proof*⟩

definition
maxl-fun :: - ⇒ 'a ⇒ 'a **where**
maxl-fun D = (λ*x*∈*carrier D*. *if* ∃ *y*. *y*∈(*upper-bounds D* {*x*}) ∧ *y* ≠ *x then*
SOME z. *z* ∈ (*upper-bounds D* {*x*}) ∧ *z* ≠ *x else x*)

lemma (**in** *Order*) *maxl-funTr*:⟦*x* ∈ *carrier D*;
∃ *y*. *y* ∈ *upper-bounds D* {*x*} ∧ *y* ≠ *x*⟧ ⟹
(*SOME z*. *z* ∈ *upper-bounds D* {*x*} ∧ *z* ≠ *x*) ∈ *carrier D*
⟨*proof*⟩

lemma (**in** *Order*) *maxl-fun-func*:*maxl-fun D* ∈ *carrier D* → *carrier D*
⟨*proof*⟩

lemma (**in** *Order*) *maxl-fun-gt*:⟦*x* ∈ *carrier D*;
∃ *y*∈ *carrier D*. *x* ≼ *y* ∧ *x* ≠ *y* ⟧ ⟹
x ≼ (*maxl-fun D x*) ∧ (*maxl-fun D x*) ≠ *x*
⟨*proof*⟩

lemma (**in** *Order*) *maxl-fun-maxl*:⟦*x* ∈ *carrier D*; *maxl-fun D x* = *x* ⟧
⟹ *maximal x*
⟨*proof*⟩

lemma (**in** *Order*) *maxl-fun-asc*:∀ *x*∈*carrier D*. *x* ≼ (*maxl-fun D x*)
⟨*proof*⟩

lemma (**in** *Order*) *S-inductive-maxl*:⟦*S-inductive-set D*; *carrier D* ≠ {}⟧ ⟹
∃ *m*. *maximal m*
⟨*proof*⟩

lemma (**in** *Order*) *maximal-mem*:*maximal m* ⟹ *m* ∈ *carrier D*
⟨*proof*⟩

definition
Chains :: - ⇒ ('*a set*) *set* **where**
Chains D == {*C*. *Chain D C*}

definition
family-Torder:: - ⇒ ('*a set*) *Order*
(‹(*fTo* -)› [*999*]*1000*) **where**
fTo D = (|*carrier* = *Chains D* , *rel* = {*Z*. *Z* ∈ (*Chains D*) × (*Chains D*) ∧ (*fst Z*) ⊆ (*snd Z*)}|)

lemma (**in** *Order*) *Chain-mem-fTo*:*Chain D C* ⟹ *C* ∈ *carrier* (*fTo D*)
⟨*proof*⟩

lemma (**in** *Order*) *fTOrder*:*Order* (*fTo D*)
⟨*proof*⟩

lemma (**in** *Order*) *fTo-Order-sub*:⟦*A* ∈ *carrier* (*fTo D*); *B* ∈ *carrier* (*fTo D*)⟧
⟹ (*A* $\preceq_{(fTo\ D)}$ *B*) = (*A* ⊆ *B*)
⟨*proof*⟩

lemma (**in** *Order*) *mem-fTo-Chain*:*X* ∈ *carrier* (*fTo D*) ⟹ *Chain D X*
⟨*proof*⟩

lemma (**in** *Order*) *mem-fTo-sub-carrier*:*X* ∈ *carrier* (*fTo D*) ⟹ *X* ⊆ *carrier D*
⟨*proof*⟩

lemma (**in** *Order*) *Un-fTo-Chain*:*Chain* (*fTo D*) *CC* ⟹ *Chain D* (⋃ *CC*)
⟨*proof*⟩

lemma (**in** *Order*) *Un-fTo-Chain-mem-fTo*:*Chain* (*fTo D*) *CC* ⟹
(⋃ *CC*) ∈ *carrier* (*fTo D*)
⟨*proof*⟩

lemma (**in** *Order*) *Un-upper-bound*:*Chain* (*fTo D*) *C* ⟹
⋃ *C* ∈ *upper-bounds* (*fTo D*) *C*
⟨*proof*⟩

lemma (**in** *Order*) *fTo-conditional-inc-C*:*C* ∈ *carrier* (*fTo D*) ⟹
C ∈ *carrier* (*Iod* (*fTo D*) {*S* ∈ *carrier fTo D*. *C* ⊆ *S*})
⟨*proof*⟩

lemma (**in** *Order*) *fTo-conditional-Un-Chain-mem1*: ⟦*C* ∈ *carrier* (*fTo D*);
Chain (*Iod* (*fTo D*) {*S* ∈ *carrier* (*fTo D*). *C* ⊆ *S*}) *Ca*; *Ca* ≠ {}⟧ ⟹
⋃ *Ca* ∈ *upper-bounds* (*Iod* (*fTo D*) {*S* ∈ *carrier fTo D*. *C* ⊆ *S*}) *Ca*

⟨*proof*⟩

lemma (**in** *Order*) *fTo-conditional-min1*: ⟦*C* ∈ *carrier* (*fTo D*);
Chain (*Iod* (*fTo D*) {*S* ∈ *carrier* (*fTo D*). *C* ⊆ *S*}) *Ca*; *Ca* ≠ {}⟧ ⟹
minimum-elem (*Iod* (*fTo D*) {*S* ∈ *carrier fTo D*. *C* ⊆ *S*})
(*upper-bounds* (*Iod* (*fTo D*) {*S* ∈ *carrier* (*fTo D*). *C* ⊆ *S*}) *Ca*) (⋃ *Ca*)
⟨*proof*⟩

lemma (**in** *Order*) *fTo-conditional-Un-Chain-mem2*: ⟦*C* ∈ *carrier* (*fTo D*);
Chain (*Iod* (*fTo D*) {*S* ∈ *carrier fTo D*. *C* ⊆ *S*}) *Ca*; *Ca* = {}⟧ ⟹
C ∈ *upper-bounds* (*Iod* (*fTo D*) {*S* ∈ *carrier* (*fTo D*). *C* ⊆ *S*}) *Ca*
⟨*proof*⟩

lemma (**in** *Order*) *fTo-conditional-min2*: ⟦*C* ∈ *carrier* (*fTo D*);
Chain (*Iod* (*fTo D*) {*S* ∈ *carrier* (*fTo D*). *C* ⊆ *S*}) *Ca*; *Ca* = {}⟧ ⟹

minimum-elem (*Iod* (*fTo D*) {$S \in$ *carrier fTo D*. $C \subseteq S$})
(*upper-bounds* (*Iod* (*fTo D*) {$S \in$ *carrier* (*fTo D*). $C \subseteq S$}) *Ca*) *C*
⟨*proof*⟩

lemma (**in** *Order*) *fTo-S-inductive*:*S-inductive-set* (*fTo D*)
⟨*proof*⟩

lemma (**in** *Order*) *conditional-min-upper-bound*: ⟦$C \in$ *carrier* (*fTo D*);
Chain (*Iod* (*fTo D*) {$S \in$ *carrier fTo D*. $C \subseteq S$}) *Ca*⟧ ⟹
$\exists X$. *minimum-elem* (*Iod* (*fTo D*) {$S \in$ *carrier* (*fTo D*). $C \subseteq S$})
(*upper-bounds* (*Iod* (*fTo D*) {$S \in$ *carrier* (*fTo D*). $C \subseteq S$}) *Ca*) *X*
⟨*proof*⟩

lemma (**in** *Order*) *Hausdorff-acTr*:$C \in$ *carrier* (*fTo D*) ⟹
S-inductive-set (*Iod* (*fTo D*) {S. $S \in$ (*carrier* (*fTo D*)) ∧ $C \subseteq S$})
⟨*proof*⟩

lemma *satisfy-cond-mem-set*:⟦ $x \in A$; $P\ x$ ⟧ ⟹ $x \in \{y \in A.\ P\ y\}$
⟨*proof*⟩

lemma (**in** *Order*) *maximal-conditional-maximal*: ⟦$C \in$ *carrier* (*fTo D*);
$maximal_{Iod\ (fTo\ D)\ \{S \in carrier\ (fTo\ D).\ C \subseteq S\}}\ m$⟧ ⟹ $maximal_{(fTo\ D)}\ m$
⟨*proof*⟩

lemma (**in** *Order*) *Hausdorff-ac*:$C \in$ *carrier* (*fTo D*) ⟹
$\exists M \in$*carrier* (*fTo D*). $C \subseteq M \wedge maximal_{(fTo\ D)}\ M$
⟨*proof*⟩

lemma (**in** *Order*) *Zorn-lemmaTr*:⟦*Chain D C*; $M \in$ *carrier fTo D*; $C \subseteq M$;
$maximal_{fTo\ D}\ M$; $b \in$ *carrier D*; $\forall s \in M.\ s \preceq b$ ⟧ ⟹
maximal b ∧ $b \in$ *upper-bounds D C*
⟨*proof*⟩

lemma (**in** *Order*) *g-Zorn-lemma1*:⟦*inductive-set D*; *Chain D C*⟧ ⟹ $\exists m$. *maximal m* ∧ $m \in$ *upper-bounds D C*
⟨*proof*⟩

lemma (**in** *Order*) *g-Zorn-lemma2*:⟦*inductive-set D*; $a \in$ *carrier D*⟧ ⟹
$\exists m \in$*carrier D*. *maximal m* ∧ $a \preceq m$
⟨*proof*⟩

lemma (**in** *Order*) *g-Zorn-lemma3*:*inductive-set D* ⟹ $\exists m \in$*carrier D*. *maximal m*
⟨*proof*⟩

Chapter 3

Group Theory. Focused on Jordan Hoelder theorem

3.1 Definition of a Group

record $'a$ *Group* = $'a$ *carrier* +
  top :: [$'a$, $'a$] ⇒ $'a$ (**infixl** ‹·1› *70*)
  iop :: $'a$ ⇒ $'a$ (‹ϱ1 -› [*81*] *80*)
  one :: $'a$ (‹**1**1›)

locale *Group* =
 fixes *G* (**structure**)
 assumes *top-closed*: *top G* ∈ *carrier G* → *carrier G* → *carrier G*
 and *tassoc* : ⟦*a* ∈ *carrier G*; *b* ∈ *carrier G*; *c* ∈ *carrier G*⟧ ⟹
 (*a* · *b*) · *c* = *a* · (*b* · *c*)
 and *iop-closed*:*iop G* ∈ *carrier G* → *carrier G*
 and *l-i* :*a* ∈ *carrier G* ⟹ (ϱ *a*) · *a* = **1**
 and *unit-closed*: **1** ∈ *carrier G*
 and *l-unit*:*a* ∈ *carrier G* ⟹ **1** · *a* = *a*

lemma (**in** *Group*) *mult-closed*:⟦*a* ∈ *carrier G*; *b* ∈ *carrier G*⟧ ⟹
 a · *b* ∈ *carrier G*
⟨*proof*⟩

lemma (**in** *Group*) *i-closed*:*a* ∈ *carrier G* ⟹ (ϱ *a*) ∈ *carrier G*
⟨*proof*⟩

lemma (**in** *Group*) *r-mult-eqn*:⟦*a* ∈ *carrier G*; *b* ∈ *carrier G*;
 c ∈ *carrier G*; *a* = *b*⟧ ⟹ *a* · *c* = *b* · *c*
⟨*proof*⟩

lemma (**in** *Group*) *l-mult-eqn*:⟦*a* ∈ *carrier G*; *b* ∈ *carrier G*;
 c ∈ *carrier G*; *a* = *b*⟧ ⟹ *c* · *a* = *c* · *b*
⟨*proof*⟩

lemma (**in** *Group*) *r-i*:$a \in carrier\ G \Longrightarrow$
$a \cdot (\varrho\ a) = \mathbf{1}$
⟨*proof*⟩

lemma (**in** *Group*) *r-unit*:$a \in carrier\ G \Longrightarrow a \cdot \mathbf{1} = a$
⟨*proof*⟩

lemma (**in** *Group*) *l-i-unique*:$[\![a \in carrier\ G;\ b \in carrier\ G;$
$b \cdot a = \mathbf{1}\ ]\!] \Longrightarrow (\varrho\ a) = b$
⟨*proof*⟩

lemma (**in** *Group*) *l-i-i*:$a \in carrier\ G \Longrightarrow (\varrho\ (\varrho\ a)) \cdot (\varrho\ a) = \mathbf{1}$
⟨*proof*⟩

lemma (**in** *Group*) *l-div-eqn*:$[\![a \in carrier\ G;\ x \in carrier\ G;\ y \in carrier\ G;$
$a \cdot x = a \cdot y\ ]\!] \Longrightarrow x = y$
⟨*proof*⟩

lemma (**in** *Group*) *r-div-eqn*:$[\![a \in carrier\ G;\ x \in carrier\ G;\ y \in carrier\ G;$
$x \cdot a = y \cdot a\ ]\!] \Longrightarrow x = y$
⟨*proof*⟩

lemma (**in** *Group*) *l-mult-eqn1*:$[\![a \in carrier\ G;\ x \in carrier\ G;\ \ y \in carrier\ G;$
$(\varrho\ a) \cdot x = (\varrho\ a) \cdot y]\!] \Longrightarrow x = y$
⟨*proof*⟩

lemma (**in** *Group*) *tOp-assocTr41*:$[\![a \in carrier\ G;\ b \in carrier\ G;\ c \in carrier\ G;$
$d \in carrier\ G]\!] \Longrightarrow a \cdot b \cdot c \cdot d = a \cdot b \cdot (c \cdot d)$
⟨*proof*⟩

lemma (**in** *Group*) *tOp-assocTr42*:$[\![a \in carrier\ G;\ b \in carrier\ G;\ c \in carrier\ G;$
$d \in carrier\ G]\!] \Longrightarrow a \cdot b \cdot c \cdot d = a \cdot (b \cdot c) \cdot d$
⟨*proof*⟩

lemma (**in** *Group*) *tOp-assocTr44*:$[\![a \in carrier\ G;\ b \in carrier\ G;\ c \in carrier\ G;$
$d \in carrier\ G\ ]\!] \Longrightarrow (\varrho\ a) \cdot b \cdot ((\varrho\ c) \cdot d) =$
$(\varrho\ \ a) \cdot ((b \cdot (\varrho\ c)) \cdot d)$
⟨*proof*⟩

lemma (**in** *Group*) *tOp-assocTr45*:$[\![a \in carrier\ G;\ b \in carrier\ G;\ c \in carrier\ G;$
$d \in carrier\ G]\!] \Longrightarrow a \cdot b \cdot c \cdot d = a \cdot (b \cdot (c \cdot d))$
⟨*proof*⟩

lemma (**in** *Group*) *one-unique*:$[\![a \in carrier\ G;\ x \in carrier\ G;\ \ x \cdot a = x]\!] \Longrightarrow$
$a = \mathbf{1}$
⟨*proof*⟩

lemma (**in** *Group*) *i-one*:$\varrho\ \mathbf{1} = \mathbf{1}$

⟨*proof*⟩

lemma (**in** *Group*) *eqn-inv1*:⟦$a \in$ *carrier* G; $x \in$ *carrier* G; $a = (\varrho\ x)$ ⟧ ⟹
$x = (\varrho\ a)$
⟨*proof*⟩

lemma (**in** *Group*) *eqn-inv2*:⟦$a \in$ *carrier* G; $x \in$ *carrier* G; $x \cdot a = x \cdot (\varrho\ x)$⟧ ⟹
$x = (\varrho\ a)$
⟨*proof*⟩

lemma (**in** *Group*) *r-one-unique*:⟦$a \in$ *carrier* G; $x \in$ *carrier* G; $a \cdot x = a$⟧ ⟹
$x = \mathbf{1}$
⟨*proof*⟩

lemma (**in** *Group*) *r-i-unique*:⟦$a \in$ *carrier* G; $x \in$ *carrier* G; $a \cdot x = \mathbf{1}$⟧ ⟹
$x = (\varrho\ a)$
⟨*proof*⟩

lemma (**in** *Group*) *iop-i-i*:$a \in$ *carrier* G ⟹ $\varrho\ (\varrho\ a) = a$
⟨*proof*⟩

lemma (**in** *Group*) *i-ab*:⟦$a \in$ *carrier* G; $b \in$ *carrier* G⟧ ⟹
$\varrho\ (a \cdot b) = (\varrho\ b) \cdot (\varrho\ a)$
⟨*proof*⟩

lemma (**in** *Group*) *sol-eq-l*:⟦$a \in$ *carrier* G; $b \in$ *carrier* G; $x \in$ *carrier* G;
$a \cdot x = b$⟧ ⟹ $x = (\varrho\ a) \cdot b$
⟨*proof*⟩

lemma (**in** *Group*) *sol-eq-r*:⟦$a \in$ *carrier* G; $b \in$ *carrier* G; $x \in$ *carrier* G;
$x \cdot a = b$ ⟧ ⟹ $x = b \cdot (\varrho\ a)$
⟨*proof*⟩

lemma (**in** *Group*) *r-div-eq*:⟦$a \in$ *carrier* G; $b \in$ *carrier* G; $a \cdot (\varrho\ b) = \mathbf{1}$⟧ ⟹
$a = b$
⟨*proof*⟩

lemma (**in** *Group*) *l-div-eq*:⟦$a \in$ *carrier* G; $b \in$ *carrier* G; $(\varrho\ a) \cdot b = \mathbf{1}$⟧ ⟹
$a = b$
⟨*proof*⟩

lemma (**in** *Group*) *i-m-closed*:⟦$a \in$ *carrier* G ; $b \in$ *carrier* G⟧ ⟹
$(\varrho\ a) \cdot b \in$ *carrier* G
⟨*proof*⟩

3.2 Subgroups

definition
sg ::[- , *'a set*] ⇒ *bool* (‹- » - › [60, 61]60) **where**

$G \gg H \longleftrightarrow H \neq \{\} \wedge H \subseteq \mathit{carrier}\ G \wedge (\forall\, a \in H.\ \forall\, b \in H.\ a \cdot_G (\varrho_G\ b) \in H)$

definition
$\mathit{Gp} :: \text{-} \Rightarrow 'a\ \mathit{set} \Rightarrow \text{-} \quad (\langle(\natural_1\text{-})\rangle\ 70)$ **where**
$\natural_G H \equiv G \lparr \mathit{carrier} := H,\ \mathit{top} := \mathit{top}\ G,\ \mathit{iop} := \mathit{iop}\ G,\ \mathit{one} := \mathit{one}\ G \rparr$

definition
$\mathit{rcs} :: [\text{-}, 'a\ \mathit{set}, 'a] \Rightarrow 'a\ \mathit{set}$ (**infix** $\langle\cdot_1\rangle$ 70) **where**
$H \cdot_G a = \{b.\ \exists\ h \in H.\ h \cdot_G a = b\}$

definition
$\mathit{lcs} :: [\text{-}, 'a, 'a\ \mathit{set}] \Rightarrow 'a\ \mathit{set}$ (**infix** $\langle\diamondsuit_1\rangle$ 70) **where**
$a \diamondsuit_G H = \{b.\ \exists\ h \in H.\ a \cdot_G h = b\}$

definition
$\mathit{nsg} :: \text{-} \Rightarrow 'a\ \mathit{set} \Rightarrow \mathit{bool} \quad (\langle\text{-} \triangleright \text{-}\rangle\ [60,61]60)$ **where**
$G \triangleright H \longleftrightarrow G \gg H \wedge (\forall\, x \in \mathit{carrier}\ G.\ H \cdot_G x = x \diamondsuit_G H)$

definition
$\mathit{set\text{-}rcs} :: [\text{-}, 'a\ \mathit{set}] \Rightarrow 'a\ \mathit{set}\ \mathit{set}$ **where**
$\mathit{set\text{-}rcs}\ G\ H = \{C.\ \exists\, a \in \mathit{carrier}\ G.\ C = H \cdot_G a\}$

definition
$\mathit{c\text{-}iop} :: [\text{-}, 'a\ \mathit{set}] \Rightarrow 'a\ \mathit{set} \Rightarrow 'a\ \mathit{set}$ **where**
$\mathit{c\text{-}iop}\ G\ H = (\lambda X{\in}\mathit{set\text{-}rcs}\ G\ H.\ \{z.\ \exists\ x \in X\ .\ \exists h \in H.\ h \cdot_G (\varrho_G\ x) = z\})$

definition
$\mathit{c\text{-}top} :: [\text{-}, 'a\ \mathit{set}] \Rightarrow (['a\ \mathit{set}, 'a\ \mathit{set}] \Rightarrow 'a\ \mathit{set})$ **where**
$\mathit{c\text{-}top}\ G\ H = (\lambda X{\in}\mathit{set\text{-}rcs}\ G\ H.\ \lambda Y{\in}\mathit{set\text{-}rcs}\ G\ H.$
$\{z.\ \exists x \in X.\ \exists\ y \in Y.\ x \cdot_G y = z\})$

lemma (**in** *Group*) *sg-subset*:$G \gg H \Longrightarrow H \subseteq \mathit{carrier}\ G$
⟨*proof*⟩

lemma (**in** *Group*) *one-Gp-one*:$G \gg H \Longrightarrow \mathbf{1}_{(\mathit{Gp}\ G\ H)} = \mathbf{1}$
⟨*proof*⟩

lemma (**in** *Group*) *carrier-Gp*:$G \gg H \Longrightarrow (\mathit{carrier}\ (\natural H)) = H$
⟨*proof*⟩

lemma (**in** *Group*) *sg-subset-elem*:$[\![G \gg H;\ h \in H\ ]\!] \Longrightarrow h \in \mathit{carrier}\ G$
⟨*proof*⟩

lemma (**in** *Group*) *sg-mult-closedr*:$[\![G \gg H;\ x \in \mathit{carrier}\ G;\ h \in H]\!] \Longrightarrow$

$x \cdot h \in carrier\ G$

⟨*proof*⟩

lemma (**in** *Group*) *sg-mult-closedl*:⟦G » H; $x \in carrier\ G$; $h \in H$⟧ ⟹
$h \cdot x \in carrier\ G$

⟨*proof*⟩

lemma (**in** *Group*) *sg-condTr1*:⟦$H \subseteq carrier\ G$; $H \neq \{\}$;
$\forall a.\ \forall b.\ a \in H \wedge b \in H \longrightarrow\ a \cdot (\varrho\ b) \in H$⟧ ⟹ $\mathbf{1} \in H$

⟨*proof*⟩

lemma (**in** *Group*) *sg-unit-closed*:G » H ⟹ $\mathbf{1} \in H$

⟨*proof*⟩

lemma (**in** *Group*) *sg-i-closed*:⟦G » H; $x \in H$⟧ ⟹ $(\varrho\ x) \in H$

⟨*proof*⟩

lemma (**in** *Group*) *sg-mult-closed*:⟦G » H; $x \in H$; $y \in H$⟧ ⟹
$x \cdot y \in H$

⟨*proof*⟩

lemma (**in** *Group*) *nsg-sg*: $G \triangleright H$ ⟹ G » H

⟨*proof*⟩

lemma (**in** *Group*) *nsg-subset*:$G \triangleright N$ ⟹ $N \subseteq carrier\ G$

⟨*proof*⟩

lemma (**in** *Group*) *nsg-lr-cst-eq*:⟦$G \triangleright N$; $a \in carrier\ G$⟧ ⟹
$a \diamondsuit N = N \cdot a$

⟨*proof*⟩

lemma (**in** *Group*) *sg-i-m-closed*:⟦G » H; $a \in H$; $b \in H$⟧ ⟹ $(\varrho\ a) \cdot b \in H$

⟨*proof*⟩

lemma (**in** *Group*) *sg-m-i-closed*:⟦G » H; $a \in H$; $b \in H$ ⟧ ⟹ $a \cdot (\varrho\ b) \in H$

⟨*proof*⟩

definition
sg-gen :: [- , *'a set*] ⇒ *'a set* **where**
sg-gen $G\ A = \bigcap \{H.\ G$ » $H \wedge A \subseteq H\}$

lemma (**in** *Group*) *smallest-sg-gen*:⟦$A \subseteq carrier\ G$; G » H; $A \subseteq H$⟧ ⟹
sg-gen $G\ A \subseteq H$

⟨*proof*⟩

lemma (**in** *Group*) *special-sg-G*: G » $(carrier\ G)$

⟨*proof*⟩

lemma (**in** *Group*) *special-sg-self*: $G = \natural(carrier\ G)$

⟨*proof*⟩

lemma (**in** *Group*) *special-sg-e*: $G \gg \{\mathbf{1}\}$
⟨*proof*⟩

lemma (**in** *Group*) *inter-sgs*:$[\![G \gg H;\ G \gg K]\!] \Longrightarrow G \gg (H \cap K)$
⟨*proof*⟩

lemma (**in** *Group*) *subg-generated*:$A \subseteq \mathit{carrier}\ G \implies G \gg (\mathit{sg\text{-}gen}\ G\ A)$
⟨*proof*⟩

definition
$Qg :: [\text{-}, 'a\ set] \Rightarrow$
$\quad (\!|\mathit{carrier}:: 'a\ set\ set, \mathit{top}:: ['a\ set, 'a\ set] \Rightarrow 'a\ set,$
$\quad\ \mathit{iop}:: 'a\ set \Rightarrow 'a\ set, \mathit{one}:: 'a\ set\ |\!)$ **where**
$Qg\ G\ H = (\!|\mathit{carrier} = \mathit{set\text{-}rcs}\ G\ H, \mathit{top} = \mathit{c\text{-}top}\ G\ H, \mathit{iop} = \mathit{c\text{-}iop}\ G\ H, \mathit{one} = H$
$|\!)$

definition
$Pj :: [\text{-}, 'a\ set] \Rightarrow\ (\ 'a => 'a\ set)$ **where**
$Pj\ G\ H = (\lambda x \in \mathit{carrier}\ G.\ H \cdot_G x)$

no-notation *inverse-divide* (**infixl** ‹'/› *70*)

abbreviation
$QGRP :: ([('a, 'more)\ Group\text{-}scheme, 'a\ set] => ('a\ set)\ Group)$
$\quad$ (**infixl** ‹'/› *70*) **where**
$G\ /\ H == Qg\ G\ H$

definition
$gHom ::[('a, 'more)\ Group\text{-}scheme, ('b, 'more1)\ Group\text{-}scheme] \Rightarrow$
$\quad ('a\ \Rightarrow 'b)\ set$ **where**
$gHom\ G\ F = \{f.\ (f \in \mathit{extensional}\ (\mathit{carrier}\ G) \wedge f \in \mathit{carrier}\ G \to \mathit{carrier}\ F) \wedge$
$\quad (\forall x \in \mathit{carrier}\ G.\ \forall y \in \mathit{carrier}\ G.\ f\ (x \cdot_G y) = (f\ x) \cdot_F (f\ y))\}$

definition
$gkernel ::[('a, 'more)\ Group\text{-}scheme\ , ('b, 'more1)\ Group\text{-}scheme, 'a \Rightarrow 'b]$
$\quad \Rightarrow 'a\ set$ **where**
$gkernel\ G\ F\ f = \{x.\ (x \in \mathit{carrier}\ G) \wedge (f\ x = \mathbf{1}_F)\}$

definition
$iim :: [('a, 'more)\ Group\text{-}scheme, ('b, 'more1)\ Group\text{-}scheme, 'a \Rightarrow 'b,$
$\quad 'b\ set]\ \Rightarrow 'a\ set$ **where**
$iim\ G\ F\ f\ K = \{x.\ (x \in \mathit{carrier}\ G) \wedge (f\ x \in K)\}$

abbreviation

GKER :: [('*a*, '*more*) *Group-scheme*, ('*b*, '*more1*) *Group-scheme*, '*a* ⇒ '*b*] ⇒ '*a* *set*
(‹(*3gker*$_{-,-}$ -)› [*88*,*88*,*89*]*88*) **where**
gker$_{G,F}$ *f* == *gkernel G F f*

definition
gsurjec :: [('*a*, '*more*) *Group-scheme*, ('*b*, '*more1*) *Group-scheme*,
'*a* ⇒ '*b*] ⇒ *bool* (‹(*3gsurj*$_{-,-}$ -)› [*88*,*88*,*89*]*88*) **where**
gsurj$_{F,G}$ *f* ⟷ *f* ∈ *gHom F G* ∧ *surj-to f* (*carrier F*) (*carrier G*)

definition
ginjec :: [('*a*, '*more*) *Group-scheme*, ('*b*, '*more1*) *Group-scheme*,
'*a* ⇒ '*b*] ⇒ *bool* (‹(*3ginj*$_{-,-}$ -)› [*88*,*88*,*89*]*88*) **where**
ginj$_{F,G}$ *f* ⟷ *f* ∈ *gHom F G* ∧ *inj-on f* (*carrier F*)

definition
gbijec :: [('*a*, '*m*) *Group-scheme*, ('*b*, '*m1*) *Group-scheme*, '*a* ⇒ '*b*]
⇒ *bool* (‹(*3gbij*$_{-,-}$ -)› [*88*,*88*,*89*]*88*) **where**
gbij$_{F,G}$ *f* ⟷ *gsurj*$_{F,G}$ *f* ∧ *ginj*$_{F,G}$ *f*

definition
Ug :: - ⇒ ('*a*, '*more*) *Group-scheme* **where**
Ug G = ♮$_G$ {**1**$_G$}

definition
Ugp :: - ⇒ *bool* **where**
Ugp G == *Group G* ∧ *carrier G* = {**1**$_G$}

definition
isomorphic :: [('*a*, '*m*) *Group-scheme*, ('*b*, '*m1*) *Group-scheme*]
⇒ *bool* (**infix** ‹≅› *100*) **where**
F ≅ *G* ⟷ (∃*f*. *gbij*$_{F,G}$ *f*)

definition
constghom :: [('*a*, '*m*) *Group-scheme*, ('*b*, '*m1*) *Group-scheme*]
⇒ ('*a* ⇒ '*b*) (‹('*1* '$_{-,-}$)› [*88*,*89*]*88*) **where**
1$_{F,G}$ = (λ*x*∈*carrier F*. **1**$_G$)

definition
cmpghom :: [('*a*, '*m*) *Group-scheme*, '*b* ⇒ '*c*, '*a* ⇒ '*b*] ⇒ '*a* ⇒ '*c* **where**
cmpghom F g f = *compose* (*carrier F*) *g f*

abbreviation
GCOMP :: ['*b* ⇒ '*c*, ('*a*, '*m*) *Group-scheme*, '*a* ⇒ '*b*] ⇒ '*a* ⇒ '*c*
(‹(*3*- ∘$_-$ -)› [*88*, *88*, *89*]*88*) **where**
g ∘$_F$ *f* == *cmpghom F g f*

lemma *Group-Ugp*:*Ugp G* ⟹ *Group G*

⟨*proof*⟩

lemma (**in** *Group*) *r-mult-in-sg*:⟦G » H; $a \in$ *carrier* G; $x \in$ *carrier* G; $x \cdot a \in H$⟧
$\Longrightarrow \exists h \in H.\ h \cdot (\varrho\ a) = x$
⟨*proof*⟩

lemma (**in** *Group*) *r-unit-sg*:⟦G » H; $h \in H$⟧ $\Longrightarrow h \cdot \mathbf{1} = h$
⟨*proof*⟩

lemma (**in** *Group*) *sg-l-unit*:⟦G » H; $h \in H$⟧ $\Longrightarrow \mathbf{1} \cdot h = h$
⟨*proof*⟩

lemma (**in** *Group*) *sg-l-i*:⟦G » H; $x \in H$ ⟧ $\Longrightarrow (\varrho\ x) \cdot x = \mathbf{1}$
⟨*proof*⟩

lemma (**in** *Group*) *sg-tassoc*:⟦G » H; $x \in H$; $y \in H$; $z \in H$⟧ $\Longrightarrow$
$x \cdot y \cdot z = x \cdot (y \cdot z)$
⟨*proof*⟩

lemma (**in** *Group*) *sg-condition*:⟦$H \subseteq$ *carrier* G; $H \neq \{\}$;
$\forall a.\ \forall b.\ a \in H \wedge b \in H \longrightarrow\ a \cdot (\varrho\ b) \in H$⟧ $\Longrightarrow G$ » H
⟨*proof*⟩

definition
Gimage :: [(*'a*, *'m*) *Group-scheme*, (*'b*, *'m1*) *Group-scheme*, *'a* ⇒ *'b*] ⇒
(*'b*, *'m1*) *Group-scheme* **where**
Gimage F G f = *Gp G* (*f* ‘(*carrier F*))

abbreviation
GIMAGE :: [(*'a*, *'m*) *Group-scheme*, (*'b*, *'m1*) *Group-scheme*,
'a ⇒ *'b*] ⇒ (*'b*, *'m1*) *Group-scheme* (‹(*3Img*$_{-,-}$ -)› [*88*,*88*,*89*]*88*) **where**
$Img_{F,G}\ f$ == *Gimage F G f*

lemma (**in** *Group*) *Group-Gp*:G » $H \Longrightarrow$ *Group* (♮ H)
⟨*proof*⟩

lemma (**in** *Group*) *Gp-carrier*:G » $H \Longrightarrow$ *carrier* (*Gp G H*) = H
⟨*proof*⟩

lemma (**in** *Group*) *sg-sg*:⟦G » K; G » H; $H \subseteq K$⟧ $\Longrightarrow$ *Gp G K* » H
⟨*proof*⟩

lemma (**in** *Group*) *sg-subset-of-subG*:⟦G » K; *Gp G K* » H⟧ $\Longrightarrow H \subseteq K$
⟨*proof*⟩

lemma *const-ghom*:⟦*Group F*; *Group G*⟧ $\Longrightarrow 1_{F,G} \in$ *gHom F G*
⟨*proof*⟩

lemma (**in** *Group*) *const-gbij*:$gbij_{(♮\ \{\mathbf{1}\}),(♮\ \{\mathbf{1}\})}\ (1_{(♮\{\mathbf{1}\}),(♮\{\mathbf{1}\})})$

⟨*proof*⟩

lemma (**in** *Group*) *unit-Groups-isom*: $(\natural\ \{\mathbf{1}\}) \cong (\natural\ \{\mathbf{1}\})$
⟨*proof*⟩

lemma *Ugp-const-gHom*:$[\![Ugp\ G;\ Ugp\ E]\!] \Longrightarrow (\lambda x{\in}carrier\ G.\ \mathbf{1}_E) \in gHom\ G\ E$
⟨*proof*⟩

lemma *Ugp-const-gbij*:$[\![Ugp\ G;\ Ugp\ E]\!] \Longrightarrow gbij_{G,E}\ (\lambda x{\in}carrier\ G.\ \mathbf{1}_E)$
⟨*proof*⟩

lemma *Ugps-isomorphic*:$[\![Ugp\ G;\ Ugp\ E]\!] \Longrightarrow G \cong E$
⟨*proof*⟩

lemma (**in** *Group*) *Gp-mult-induced*:$[\![G \lhd\!\lhd L;\ a \in L;\ b \in L]\!]\ \Longrightarrow$
$a\ \cdot_{(Gp\ G\ L)}\ b = a \cdot b$
⟨*proof*⟩

lemma (**in** *Group*) *sg-i-induced*:$[\![G \lhd\!\lhd L;\ a \in L]\!]\ \Longrightarrow \varrho_{(Gp\ G\ L)}\ a = \varrho\ a$
⟨*proof*⟩

lemma (**in** *Group*) *Gp-mult-induced1*:$[\![G \lhd\!\lhd H\ ;\ G \lhd\!\lhd K;\ a \in H \cap K;\ b \in H \cap K]\!]$
$\Longrightarrow a\ \cdot_{\natural(H \cap K)}\ b = a\ \cdot_{(\natural H)}\ b$
⟨*proof*⟩

lemma (**in** *Group*) *Gp-mult-induced2*:$[\![G \lhd\!\lhd H\ ;\ G \lhd\!\lhd K;\ a \in H \cap K;\ b \in H \cap K]\!]$
$\Longrightarrow a\ \cdot_{\natural(H \cap K)}\ b = a\ \cdot_{(\natural K)}\ b$
⟨*proof*⟩

lemma (**in** *Group*) *sg-i-induced1*:$[\![G \lhd\!\lhd H\ ;\ G \lhd\!\lhd K;\ a \in H \cap K]\!]$
$\Longrightarrow\ \varrho_{\natural(H \cap K)}\ a = \varrho_{(\natural H)}\ a$
⟨*proof*⟩

lemma (**in** *Group*) *sg-i-induced2*:$[\![G \lhd\!\lhd H\ ;\ G \lhd\!\lhd K;\ a \in H \cap K]\!]$
$\Longrightarrow\ \varrho_{\natural(H \cap K)}\ a = \varrho_{\natural K}\ a$
⟨*proof*⟩

lemma (**in** *Group*) *subg-sg-sg*:$[\![G \lhd\!\lhd K;\ (Gp\ G\ K) \lhd\!\lhd H\ ]\!] \Longrightarrow G \lhd\!\lhd H$
⟨*proof*⟩

lemma (**in** *Group*) *Gp-inherited*:$[\![G \lhd\!\lhd K;\ G \lhd\!\lhd L;\ K \subseteq L]\!] \Longrightarrow$
$Gp\ (Gp\ G\ L)\ K = Gp\ G\ K$
⟨*proof*⟩

3.3 Cosets

lemma (**in** *Group*) *mem-lcs*:$[\![G \lhd\!\lhd H;\ a \in carrier\ G;\ x \in a \diamondsuit H]\!]\ \Longrightarrow$

$x \in carrier\ G$

⟨*proof*⟩

lemma (**in** *Group*) *lcs-subset*:⟦G » H; $a \in carrier\ G$⟧ ⟹ $a \diamondsuit H \subseteq carrier\ G$
⟨*proof*⟩

lemma (**in** *Group*) *a-in-lcs*:⟦G » H; $a \in carrier\ G$⟧ ⟹ $a \in a \diamondsuit H$
⟨*proof*⟩

lemma (**in** *Group*) *eq-lcs1*:⟦G » H; $a \in carrier\ G$; $b \in carrier\ G$;
$x \in a \diamondsuit H$; $a \diamondsuit H = b \diamondsuit H$⟧ ⟹ $x \in b \diamondsuit H$
⟨*proof*⟩

lemma (**in** *Group*) *eq-lcs2*:⟦G » H; $a \in carrier\ G$; $b \in carrier\ G$;
$a \diamondsuit H = b \diamondsuit H$⟧ ⟹ $a \in b \diamondsuit H$
⟨*proof*⟩

lemma (**in** *Group*) *lcs-mem-ldiv*:⟦G » H; $a \in carrier\ G$; $b \in carrier\ G$⟧ ⟹
$(a \in b \diamondsuit H) = ((\varrho\ b) \cdot a \in H)$
⟨*proof*⟩

lemma (**in** *Group*) *lcsTr5*:⟦G » H; $a \in carrier\ G$; $b \in carrier\ G$;
$(\varrho\ a) \cdot b \in H$; $x \in a \diamondsuit H$⟧ ⟹ $((\varrho\ b) \cdot x) \in H$
⟨*proof*⟩

lemma (**in** *Group*) *lcsTr6*:⟦G » H; $a \in carrier\ G$; $b \in carrier\ G$;
$(\varrho\ a) \cdot b \in H$; $x \in a \diamondsuit H$⟧ ⟹ $x \in b \diamondsuit H$
⟨*proof*⟩

lemma (**in** *Group*) *lcs-Unit1*:G » H ⟹ $\mathbf{1} \diamondsuit H = H$
⟨*proof*⟩

lemma (**in** *Group*) *lcs-Unit2*:⟦G » H; $h \in H$⟧ ⟹ $h \diamondsuit H = H$
⟨*proof*⟩

lemma (**in** *Group*) *lcsTr7*:⟦G » H; $a \in carrier\ G$; $b \in carrier\ G$; $(\varrho\ a) \cdot b \in H$⟧
⟹ $a \diamondsuit H \subseteq b \diamondsuit H$
⟨*proof*⟩

lemma (**in** *Group*) *lcsTr8*:⟦G » H; $a \in carrier\ G$; $h \in H$⟧ ⟹ $a \cdot h \in a \diamondsuit H$
⟨*proof*⟩

lemma (**in** *Group*) *lcs-tool1*:⟦G » H; $a \in carrier\ G$; $b \in carrier\ G$;
$(\varrho\ a) \cdot b \in H$⟧ ⟹ $(\varrho\ b) \cdot a \in H$
⟨*proof*⟩

theorem (**in** *Group*) *lcs-eq*:⟦G » H; $a \in carrier\ G$; $b \in carrier\ G$⟧ ⟹

$((\varrho\ a) \cdot b \in H) = (a \diamondsuit H = b \diamondsuit H)$
⟨*proof*⟩

lemma (**in** *Group*) *rcs-subset*:$[\![G \lessdot H;\ a \in carrier\ G]\!] \Longrightarrow H \cdot a \subseteq carrier\ G$
⟨*proof*⟩

lemma (**in** *Group*) *mem-rcs*:$[\![G \lessdot H;\ x \in H \cdot a]\!] \Longrightarrow \exists h{\in}H.\ h \cdot a = x$
⟨*proof*⟩

lemma (**in** *Group*) *rcs-subset-elem*:$[\![G \lessdot H;\ a \in carrier\ G;\ x \in H \cdot a]\!] \Longrightarrow$
$x \in carrier\ G$
⟨*proof*⟩

lemma (**in** *Group*) *rcs-in-set-rcs*:$[\![G \lessdot H;\ a \in carrier\ G]\!] \Longrightarrow$
$H \cdot a \in set\text{-}rcs\ G\ H$
⟨*proof*⟩

lemma (**in** *Group*) *rcsTr0*:$[\![G \lessdot H;\ a \in carrier\ G;\ b \in carrier\ G]\!] \Longrightarrow$
$H \cdot (a \cdot b) \in set\text{-}rcs\ G\ H$
⟨*proof*⟩

lemma (**in** *Group*) *a-in-rcs*:$[\![G \lessdot H;\ a \in carrier\ G]\!] \Longrightarrow a \in H \cdot a$
⟨*proof*⟩

lemma (**in** *Group*) *rcs-nonempty*:$[\![G \lessdot H;\ X \in set\text{-}rcs\ G\ H]\!] \Longrightarrow X \neq \{\}$
⟨*proof*⟩

lemma (**in** *Group*) *rcs-tool0*:$[\![G \lessdot H;\ a \in carrier\ G;\ b \in carrier\ G;$
$a \cdot (\varrho\ b) \in H]\!] \Longrightarrow b \cdot (\varrho\ a) \in H$
⟨*proof*⟩

lemma (**in** *Group*) *rcsTr1*:$[\![G \lessdot H;\ a \in carrier\ G;\ b \in carrier\ G;$
$x \in H \cdot a;\ H \cdot a = H \cdot b]\!] \Longrightarrow x \in H \cdot b$
⟨*proof*⟩

lemma (**in** *Group*) *rcs-eqTr*:$[\![G \lessdot H;\ a \in carrier\ G;\ b \in carrier\ G;$
$H \cdot a = H \cdot b]\!] \Longrightarrow a \in H \cdot b$
⟨*proof*⟩

lemma (**in** *Group*) *rcs-eqTr1*:$[\![G \lessdot H;\ a \in carrier\ G;\ b \in carrier\ G]\!] \Longrightarrow$
$(a \in H \cdot b) = (a \cdot (\varrho\ b) \in H)$
⟨*proof*⟩

lemma (**in** *Group*) *rcsTr2*:$[\![G \lessdot H;\ a \in carrier\ G;\ b \in H \cdot (\varrho\ a)]\!] \Longrightarrow$
$b \cdot a \in H$
⟨*proof*⟩

lemma (**in** *Group*) *rcsTr5*:$[\![G \lessdot H;\ a \in carrier\ G;\ b \in carrier\ G;$
$b \cdot (\varrho\ a) \in H;\ x \in H \cdot a]\!] \Longrightarrow x \cdot (\varrho\ b) \in H$

⟨*proof*⟩

lemma (**in** *Group*) *rcsTr6*:⟦G » H; $a \in$ *carrier* G; $b \in$ *carrier* G;
$b \cdot (\varrho\ a) \in H$; $x \in H \cdot a$⟧ ⟹ $x \in H \cdot b$
⟨*proof*⟩

lemma (**in** *Group*) *rcs-Unit1*:G » H ⟹ $H \cdot \mathbf{1} = H$
⟨*proof*⟩

lemma (**in** *Group*) *unit-rcs-in-set-rcs*:G » H ⟹ $H \in$ *set-rcs* G H
⟨*proof*⟩

lemma (**in** *Group*) *rcs-Unit2*:⟦G » H; $h \in H$⟧ ⟹ $H \cdot h = H$
⟨*proof*⟩

lemma (**in** *Group*) *rcsTr7*:⟦G » H; $a \in$ *carrier* G; $b \in$ *carrier* G; $b \cdot (\varrho\ a) \in H$⟧
⟹ $H \cdot a \subseteq H \cdot b$
⟨*proof*⟩

lemma (**in** *Group*) *rcs-tool1*:⟦G » H; $a \in$ *carrier* G; $b \in$ *carrier* G;
$b \cdot (\varrho\ a) \in H$⟧ ⟹ $a \cdot (\varrho\ b) \in H$
⟨*proof*⟩

lemma (**in** *Group*) *rcs-tool2*:⟦G » H; $a \in$ *carrier* G; $x \in H \cdot a$⟧ ⟹
$\exists\ h \in H.\ h \cdot a = x$
⟨*proof*⟩

theorem (**in** *Group*) *rcs-eq*:⟦G » H; $a \in$ *carrier* G; $b \in$ *carrier* G⟧ ⟹
$(b \cdot (\varrho\ a) \in H) = (H \cdot a = H \cdot b)$
⟨*proof*⟩

lemma (**in** *Group*) *rcs-eq1*:⟦G » H; $a \in$ *carrier* G; $x \in H \cdot a$⟧ ⟹
$H \cdot a = H \cdot x$
⟨*proof*⟩

lemma (**in** *Group*) *rcs-eq2*:⟦G » H; $a \in$ *carrier* G; $b \in$ *carrier* G;
$(H \cdot a) \cap (H \cdot b) \neq \{\}$⟧ ⟹ $(H \cdot a) = (H \cdot b)$
⟨*proof*⟩

lemma (**in** *Group*) *rcs-meet*:⟦G » H; $X \in$ *set-rcs* G H; $Y \in$ *set-rcs* G H;
$X \cap Y \neq \{\}$⟧ ⟹ $X = Y$
⟨*proof*⟩

lemma (**in** *Group*) *rcsTr8*:⟦G » H; $a \in$ *carrier* G; $h \in H$; $x \in H \cdot a$⟧ ⟹
$h \cdot x \in H \cdot a$
⟨*proof*⟩

lemma (**in** *Group*) *rcsTr9*:⟦G » H; $a \in$ *carrier* G; $h \in H$; $(\varrho\ x) \in H \cdot a$⟧ ⟹
$h \cdot (\varrho\ x) \in H \cdot a$

⟨*proof*⟩

lemma (**in** *Group*) *rcsTr10*:⟦G » H; $a \in$ *carrier* G; $x \in H \cdot a$; $y \in H \cdot a$⟧ ⟹
$x \cdot (\varrho\ y) \in H$
⟨*proof*⟩

lemma (**in** *Group*) *PrSubg4-2*:⟦G » H; $a \in$ *carrier* G; $x \in H \cdot (\varrho\ a)$⟧ ⟹
$x \in \{z.\ \exists\, v{\in}(H \cdot a).\ \exists\, h{\in}H.\ h \cdot (\varrho\ v) = z\}$
⟨*proof*⟩

lemma (**in** *Group*) *rcs-fixed*:⟦G » H; $a \in$ *carrier* G; $H \cdot a = H$⟧ ⟹ $a \in H$
⟨*proof*⟩

lemma (**in** *Group*) *rcs-fixed1*:⟦G » H; $a \in$ *carrier* G; $h \in H$⟧ ⟹
$H \cdot a = (H \cdot (h \cdot a))$
⟨*proof*⟩

lemma (**in** *Group*) *rcs-fixed2*:G » H ⟹ $\forall\, h{\in}H.\ H \cdot h = H$
⟨*proof*⟩

lemma (**in** *Group*) *Gp-rcs*:⟦G » H; G » K; $H \subseteq K$; $x \in K$⟧ ⟹
$H \cdot_{(Gp\ G\ K)} x = (H \cdot x)$
⟨*proof*⟩

lemma (**in** *Group*) *subg-lcs*:⟦G » H; G » K; $H \subseteq K$; $x \in K$⟧ ⟹
$x \diamondsuit_{(Gp\ G\ K)} H = x \diamondsuit H$
⟨*proof*⟩

3.4 Normal subgroups and Quotient groups

lemma (**in** *Group*) *nsg1*:⟦G » H; $b \in$ *carrier* G; $h \in H$;
$\forall\, a{\in}$ *carrier* $G.\ \forall\, h{\in}H.\ (a \cdot h){\cdot}\ (\varrho\ a) \in H$⟧ ⟹ $b \cdot h \cdot (\varrho\ b) \in H$
⟨*proof*⟩

lemma (**in** *Group*) *nsg2*:⟦G » H; $b \in$ *carrier* G; $h \in H$;
$\forall\, a{\in}$*carrier* $G.\ \forall\, h{\in}H.\ (a \cdot h) \cdot (\varrho\ a) \in H$⟧ ⟹ $(\varrho\ b) \cdot h \cdot b \in H$
⟨*proof*⟩

lemma (**in** *Group*) *nsg-subset-elem*:⟦$G \triangleright H$; $h \in H$⟧ ⟹ $h \in$ *carrier* G
⟨*proof*⟩

lemma (**in** *Group*) *nsg-l-rcs-eq*:⟦$G \triangleright N$; $a \in$ *carrier* G⟧ ⟹ $a \diamondsuit N = N \cdot a$
⟨*proof*⟩

lemma (**in** *Group*) *sg-nsg1*:⟦G » H; $\forall\, a{\in}$ *carrier* $G.\ \forall\, h{\in}H.\ (a \cdot h){\cdot}\ (\varrho\ a) \in H$;
$b \in$ *carrier* G ⟧ ⟹ $H \cdot b = b \diamondsuit H$
⟨*proof*⟩

lemma (**in** *Group*) *cond-nsg*:$[\![G \mathbin{»} H;\ \forall\, a{\in}\mathit{carrier}\ G.\ \forall\, h{\in}H.\ a \cdot h \cdot (\varrho\ a) \in H\ ]\!]$
$\Longrightarrow G \triangleright H$
⟨*proof*⟩

lemma (**in** *Group*) *special-nsg-e*:$G \mathbin{»} H \Longrightarrow \mathit{Gp}\ G\ H \triangleright \{\mathbf{1}\}$
⟨*proof*⟩

lemma (**in** *Group*) *special-nsg-G*:$G \triangleright (\mathit{carrier}\ G)$
⟨*proof*⟩

lemma (**in** *Group*) *special-nsg-G1*:$G \mathbin{»} H \Longrightarrow \mathit{Gp}\ G\ H \triangleright H$
⟨*proof*⟩

lemma (**in** *Group*) *nsgTr0*:$[\![G \triangleright N;\ a \in \mathit{carrier}\ G;\ b \in \mathit{carrier}\ G;\ b \in N \cdot a\ ]\!]$
$\Longrightarrow\ (a \cdot (\varrho\ b) \in N) \wedge ((\varrho\ a) \cdot b \in N)$
⟨*proof*⟩

lemma (**in** *Group*) *nsgTr1*:$[\![G \triangleright N;\ a \in \mathit{carrier}\ G;\ b \in \mathit{carrier}\ G;\ b \cdot (\varrho\ a) \in N]\!]$
$\Longrightarrow (\varrho\ b) \cdot a \in N$
⟨*proof*⟩

lemma (**in** *Group*) *nsgTr2*:$[\![a \in \mathit{carrier}\ G;\ b \in \mathit{carrier}\ G;\ a1 \in \mathit{carrier}\ G;$
$b1 \in \mathit{carrier}\ G\ ]\!] \Longrightarrow (a \cdot b) \cdot (\varrho\ (a1 \cdot b1)) =$
$a \cdot (((b \cdot (\varrho\ b1)) \cdot ((\varrho\ a1) \cdot a)) \cdot (\varrho\ a))$
⟨*proof*⟩

lemma (**in** *Group*) *nsgPr1*:$[\![G \triangleright N;\ a \in \mathit{carrier}\ G;\ h \in N]\!] \Longrightarrow$
$a \cdot (h \cdot (\varrho\ a)) \in N$
⟨*proof*⟩

lemma (**in** *Group*) *nsgPr1-1*:$[\![G \triangleright N;\ a \in \mathit{carrier}\ G\ ;\ h \in N]\!] \Longrightarrow$
$(a \cdot h) \cdot (\varrho\ a) \in N$
⟨*proof*⟩

lemma (**in** *Group*) *nsgPr2*:$[\![G \triangleright N;\ a \in \mathit{carrier}\ G;\ h \in N]\!] \Longrightarrow$
$(\varrho\ a) \cdot (h \cdot a) \in N$
⟨*proof*⟩

lemma (**in** *Group*) *nsgPr2-1*:$[\![G \triangleright N;\ a \in \mathit{carrier}\ G;\ h \in N]\!] \Longrightarrow$
$(\varrho\ a) \cdot h \cdot a \in N$
⟨*proof*⟩

lemma (**in** *Group*) *nsgTr3*:$[\![G \triangleright N;\ a \in \mathit{carrier}\ G;\ b \in \mathit{carrier}\ G;$
$a1 \in \mathit{carrier}\ G;\ b1 \in \mathit{carrier}\ G;\ \ a \cdot (\varrho\ a1) \in N;\ b \cdot (\varrho\ b1) \in N]\!] \Longrightarrow$
$(a \cdot b) \cdot (\varrho\ (a1 \cdot b1)) \in N$
⟨*proof*⟩

lemma (**in** *Group*) *nsg-in-Gp*:$[\![G \triangleright N;\ G \mathbin{»} H;\ N \subseteq H]\!] \Longrightarrow (\mathit{Gp}\ G\ H) \triangleright N$
⟨*proof*⟩

lemma (**in** *Group*) *nsgTr4*:⟦$G \triangleright N$; $a \in$ *carrier* G; $x \in N \cdot a$⟧ ⟹
($\varrho\ x$) $\in N \cdot (\varrho\ a)$
⟨*proof*⟩

lemma (**in** *Group*) *c-topTr1*:⟦$G \triangleright N$; $a \in$ *carrier* G; $b \in$ *carrier* G;
$a1 \in$ *carrier* G; $b1 \in$ *carrier* G; $N \cdot a = N \cdot a1$; $N \cdot b = N \cdot b1$⟧ ⟹
$N \cdot (a \cdot b) = N \cdot (a1 \cdot b1)$
⟨*proof*⟩

lemma (**in** *Group*) *c-topTr2*:⟦$G \triangleright N$; $a \in$ *carrier* G; $a1 \in$ *carrier* G;
$N \cdot a = N \cdot a1$ ⟧ ⟹ $N \cdot (\varrho\ a) = N \cdot (\varrho\ a1)$
⟨*proof*⟩

lemma (**in** *Group*) *c-iop-welldefTr1*:⟦$G \triangleright N$; $a \in$ *carrier* G⟧ ⟹
c-iop $G\ N\ (N \cdot a) \subseteq\ N \cdot (\varrho\ a)$
⟨*proof*⟩

lemma (**in** *Group*) *c-iop-welldefTr2*:⟦$G \triangleright N$; $a \in$ *carrier* G⟧ ⟹
$N \cdot (\varrho\ a) \subseteq$ *c-iop* $G\ N\ (N \cdot a)$
⟨*proof*⟩

lemma (**in** *Group*) *c-iop-welldef*:⟦$G \triangleright N$; $a \in$ *carrier* G⟧ ⟹
c-iop $G\ N\ (N \cdot a) =\ N \cdot (\varrho\ a)$
⟨*proof*⟩

lemma (**in** *Group*) *c-top-welldefTr1*:⟦$G \triangleright N$; $a \in$ *carrier* G;
$b \in$ *carrier* G; $x \in N \cdot a$; $y \in N \cdot b$⟧ ⟹ $x \cdot y \in N \cdot (a \cdot b)$
⟨*proof*⟩

lemma (**in** *Group*) *c-top-welldefTr2*:⟦$G \triangleright N$; $a \in$ *carrier* G; $b \in$ *carrier* G ⟧
⟹ *c-top* $G\ N\ (N \cdot a)\ (N \cdot b) \subseteq\ N \cdot (a \cdot b)$
⟨*proof*⟩

lemma (**in** *Group*) *c-top-welldefTr4*:⟦$G \triangleright N$; $a \in$ *carrier* G; $b \in$ *carrier* G;
$x \in N \cdot (a \cdot b)$⟧ ⟹ $x \in$ *c-top* $G\ N\ (N \cdot a)\ (N \cdot b)$
⟨*proof*⟩

lemma (**in** *Group*) *c-top-welldefTr5*:⟦$G \triangleright N$; $a \in$ *carrier* G; $b \in$ *carrier* G⟧ ⟹
$N \cdot (a \cdot b) \subseteq$ *c-top* $G\ N\ (N \cdot a)\ (N \cdot b)$
⟨*proof*⟩

lemma (**in** *Group*) *c-top-welldef*:⟦$G \triangleright N$; $a \in$ *carrier* G; $b \in$ *carrier* G⟧ ⟹
$N \cdot (a \cdot b) =$ *c-top* $G\ N\ (N \cdot a)\ (N \cdot b)$
⟨*proof*⟩

lemma (**in** *Group*) *Qg-unitTr*:⟦$G \triangleright N$; $a \in$ *carrier* G⟧ ⟹
c-top $G\ N\ N\ (N \cdot a) = N \cdot a$
⟨*proof*⟩

lemma (**in** *Group*) *Qg-unit*:$G \triangleright N \implies \forall x{\in}$*set-rcs G N*. *c-top G N N x* = *x*
⟨*proof*⟩

lemma (**in** *Group*) *Qg-iTr*:$[\![G \triangleright N;\ a \in$ *carrier G*$]\!] \implies$
c-top G N (*c-iop G N* (*N* ⋅ *a*)) (*N* ⋅ *a*) = *N*
⟨*proof*⟩

lemma (**in** *Group*) *Qg-i*:$G \triangleright N \implies$
$\forall x \in$ *set-rcs G N*. *c-top G N* (*c-iop G N x*) *x* = *N*
⟨*proof*⟩

lemma (**in** *Group*) *Qg-tassocTr*:
$[\![G \triangleright N;\ a \in$ *carrier G*; $b \in$ *carrier G*; $c \in$ *carrier G* $]\!] \implies$
c-top G N (*N* ⋅ *a*) (*c-top G N* (*N* ⋅ *b*) (*N* ⋅ *c*)) =
c-top G N (*c-top G N* (*N* ⋅ *a*) (*N* ⋅ *b*)) (*N* ⋅ *c*)
⟨*proof*⟩

lemma (**in** *Group*) *Qg-tassoc*: $G \triangleright N \implies$
$\forall X{\in}$*set-rcs G N*. $\forall Y{\in}$*set-rcs G N*. $\forall Z{\in}$*set-rcs G N*. *c-top G N X* (*c-top G N Y Z*)
= *c-top G N* (*c-top G N X Y*) *Z*
⟨*proof*⟩

lemma (**in** *Group*) *Qg-top*:$G \triangleright N \implies$
c-top G N : *set-rcs G N* → *set-rcs G N* → *set-rcs G N*
⟨*proof*⟩

lemma (**in** *Group*) *Qg-top-closed*:$[\![G \triangleright N;\ A \in$ *set-rcs G N*; $B \in$ *set-rcs G N*$]\!] \implies$
c-top G N A B ∈ *set-rcs G N*
⟨*proof*⟩

lemma (**in** *Group*) *Qg-iop*: $G \triangleright N \implies$
c-iop G N :*set-rcs G N* → *set-rcs G N*
⟨*proof*⟩

lemma (**in** *Group*) *Qg-iop-closed*:$[\![G \triangleright N;\ A \in$ *set-rcs G N*$]\!] \implies$
c-iop G N A ∈ *set-rcs G N*
⟨*proof*⟩

lemma (**in** *Group*) *Qg-unit-closed*: $G \triangleright N \implies$ *N* ∈ *set-rcs G N*
⟨*proof*⟩

theorem (**in** *Group*) *Group-Qg*:$G \triangleright N \implies$ *Group* (*Qg G N*)
⟨*proof*⟩

lemma (**in** *Group*) *Qg-one*:$G \triangleright N \implies$ *one* (*G* / *N*) = *N*
⟨*proof*⟩

lemma (**in** *Group*) *Qg-carrier*:*carrier* $(G\ /\ (N::'a\ set)) = set\text{-}rcs\ G\ N$
⟨*proof*⟩

lemma (**in** *Group*) *Qg-unit-group*:$G \triangleright N \Longrightarrow$
$(set\text{-}rcs\ G\ N = \{N\}) = (carrier\ G = N)$
⟨*proof*⟩

lemma (**in** *Group*) *Gp-Qg*:$G \triangleright N \Longrightarrow Gp(G\ /\ N)\ (carrier(G\ /\ N)) = G\ /\ N$
⟨*proof*⟩

lemma (**in** *Group*) *Pj-hom0*:$[\![G \triangleright N;\ x \in carrier\ G;\ y \in carrier\ G]\!]$
$\Longrightarrow Pj\ G\ N\ (x \cdot y) = (Pj\ G\ N\ x) \cdot_{(G\ /\ N)} (Pj\ G\ N\ y)$
⟨*proof*⟩

lemma (**in** *Group*) *Pj-ghom*:$G \triangleright N \Longrightarrow (Pj\ G\ N) \in gHom\ G\ (G\ /\ N)$
⟨*proof*⟩

lemma (**in** *Group*) *Pj-mem*:$[\![G \triangleright N;\ x \in carrier\ G]\!] \Longrightarrow (Pj\ G\ N)\ x = N \cdot x$
⟨*proof*⟩

lemma (**in** *Group*) *Pj-gsurjec*:$G \triangleright N \Longrightarrow gsurjec\ G\ (G/N)\ (Pj\ G\ N)$
⟨*proof*⟩

lemma (**in** *Group*) *lcs-in-Gp*:$[\![G \lessdot H;\ G \lessdot K;\ K \subseteq H;\ a \in H]\!] \Longrightarrow$
$a \diamondsuit K = a \diamondsuit_{(Gp\ G\ H)} K$
⟨*proof*⟩

lemma (**in** *Group*) *rcs-in-Gp*:$[\![G \lessdot H;\ G \lessdot K;\ K \subseteq H;\ a \in H\ ]\!] \Longrightarrow$
$K \cdot a = K \cdot_{(Gp\ G\ H)} a$
⟨*proof*⟩

end

theory *Algebra3* **imports** *Algebra2* **begin**

3.5 Setproducts

definition
commutators:: - ⇒ $'a\ set$ **where**
$commutators\ G = \{z.\ \exists\ a \in carrier\ G.\ \exists\, b \in carrier\ G.$
$((a \cdot_G b) \cdot_G (\varrho_G\ a)) \cdot_G (\varrho_G\ b) = z\}$

lemma (**in** *Group*) *contain-commutator*:$[\![G \lessdot H;\ (commutators\ G) \subseteq H]\!] \Longrightarrow G \triangleright H$
⟨*proof*⟩

definition

s-top :: [- , *'a set*, *'a set*] ⇒ *'a set* **where**
s-top G H K = {*z*. ∃ *x* ∈ *H*. ∃ *y* ∈ *K*. (*x* ·$_G$ *y* = *z*)}

abbreviation
S-TOP :: [(*'a*, *'m*) *Group-scheme*, *'a set*, *'a set*] ⇒ *'a set*
(‹(*3*- ⋄$_1$ -)› [*66*,*67*]*66*) **where**
H ⋄$_G$ *K* == *s-top G H K*

lemma (**in** *Group*) *s-top-induced*:⟦*G* » *L*; *H* ⊆ *L*; *K* ⊆ *L*⟧ ⟹
H ⋄$_{Gp}$ *G L K* = *H* ⋄$_G$ *K*
⟨*proof*⟩

lemma (**in** *Group*) *s-top-l-unit*:*G* » *K* ⟹ {**1**} ⋄$_G$ *K* = *K*
⟨*proof*⟩

lemma (**in** *Group*) *s-top-r-unit*:*G* » *K* ⟹ *K* ⋄$_G$ {**1**} = *K*
⟨*proof*⟩

lemma (**in** *Group*) *s-top-sub*:⟦*G* » *H*; *G* » *K*⟧ ⟹ *H* ⋄$_G$ *K* ⊆ *carrier G*
⟨*proof*⟩

lemma (**in** *Group*) *sg-inc-set-mult*:⟦*G* » *L*; *H* ⊆ *L*; *K* ⊆ *L*⟧ ⟹ *H* ⋄$_G$ *K* ⊆ *L*
⟨*proof*⟩

lemma (**in** *Group*) *s-top-sub1*:⟦*H* ⊆ (*carrier G*); *K* ⊆ (*carrier G*)⟧ ⟹
H ⋄$_G$ *K* ⊆ *carrier G*
⟨*proof*⟩

lemma (**in** *Group*) *s-top-elem*:⟦*G* » *H*; *G* » *K*; *a* ∈ *H*; *b* ∈ *K*⟧ ⟹ *a* · *b* ∈ *H* ⋄$_G$ *K*
⟨*proof*⟩

lemma (**in** *Group*) *s-top-elem1*:⟦*H* ⊆ *carrier G*; *K* ⊆ *carrier G*; *a* ∈ *H*; *b* ∈ *K*⟧ ⟹
a · *b* ∈ *H* ⋄$_G$ *K*
⟨*proof*⟩

lemma (**in** *Group*) *mem-s-top*:⟦*H* ⊆ *carrier G*; *K* ⊆ *carrier G*; *u* ∈ *H* ⋄$_G$ *K*⟧ ⟹
∃ *a* ∈ *H*. ∃ *b* ∈ *K*. (*a* · *b* = *u*)
⟨*proof*⟩

lemma (**in** *Group*) *s-top-mono*:⟦*H* ⊆ *carrier G*; *K* ⊆ *carrier G*; *H1* ⊆ *H*; *K1* ⊆ *K*⟧
⟹ *H1* ⋄$_G$ *K1* ⊆ *H* ⋄$_G$ *K*
⟨*proof*⟩

lemma (**in** *Group*) *s-top-unit-closed*:⟦*G* » *H*; *G* » *K*⟧ ⟹ **1** ∈ *H* ⋄$_G$ *K*
⟨*proof*⟩

lemma (**in** *Group*) *s-top-commute*:$[\![G \gg H;\ G \gg K;\ K \diamond_G H = H \diamond_G K;$
$u \in H \diamond_G K;\ \ v \in\ H \diamond_G K]\!] \Longrightarrow\ u \cdot v \in H \diamond_G K$
⟨*proof*⟩

lemma (**in** *Group*) *s-top-commute1*:$[\![G \gg H;\ G \gg K;\ K \diamond_G H = H \diamond_G K;$
$u \in H \diamond_G K]\!] \Longrightarrow (\varrho\ u) \in H \diamond_G K$
⟨*proof*⟩

lemma (**in** *Group*) *s-top-commute-sg*:$[\![G \gg H;\ G \gg K;\ K \diamond_G H = H \diamond_G K]\!] \Longrightarrow$
$G \gg (H \diamond_G K)$
⟨*proof*⟩

lemma (**in** *Group*) *s-top-assoc*:$[\![G \gg H;\ G \gg K;\ G \gg L]\!] \Longrightarrow$
$(H \diamond_G K) \diamond_G L =\ H \diamond_G (K \diamond_G L)$
⟨*proof*⟩

lemma (**in** *Group*) *s-topTr6*:$[\![G \gg H1;\ G \gg H2;\ G \gg K;\ H1 \subseteq K]\!] \Longrightarrow$
$(H1 \diamond_G H2) \cap K = H1 \diamond_G (H2 \cap K)$
⟨*proof*⟩

lemma (**in** *Group*) *s-topTr6-1*:$[\![G \gg H1;\ G \gg H2;\ G \gg K;\ H2 \subseteq K]\!] \Longrightarrow$
$(H1 \diamond_G H2) \cap K = (H1 \cap K) \diamond_G H2$
⟨*proof*⟩

lemma (**in** *Group*) *l-sub-smult*:$[\![G \gg H;\ G \gg K]\!] \Longrightarrow H \subseteq H \diamond_G K$
⟨*proof*⟩

lemma (**in** *Group*) *r-sub-smult*:$[\![G \gg H;\ G \gg K]\!] \Longrightarrow K \subseteq H \diamond_G K$
⟨*proof*⟩

lemma (**in** *Group*) *s-topTr8*:$G \gg H \Longrightarrow H = H \diamond_G H$
⟨*proof*⟩

3.6 Preliminary lemmas for Zassenhaus

lemma (**in** *Group*) *Gp-sg-subset*:$[\![G \gg H;\ Gp\ G\ H \gg K]\!] \Longrightarrow K \subseteq H$
⟨*proof*⟩

lemma (**in** *Group*) *inter-Gp-nsg*:$[\![G \triangleright N;\ G \gg H\ ]\!] \Longrightarrow (\natural H) \triangleright (H \cap N)$
⟨*proof*⟩

lemma (**in** *Group*) *ZassenhausTr0*:$[\![G \gg H;\ G \gg H1;\ G \gg K;\ G \gg K1;$
$Gp\ G\ H \triangleright H1;\ Gp\ G\ K \triangleright K1]\!] \Longrightarrow Gp\ G\ (H \cap K) \triangleright (H \cap K1)$
⟨*proof*⟩

lemma (**in** *Group*) *lcs-sub-s-mult*:$[\![G \gg H;\ G \gg N;\ a \in H]\!] \Longrightarrow a \diamondsuit N \subseteq H \diamond_G N$
⟨*proof*⟩

lemma (**in** *Group*) *rcs-sub-smult*:$[\![G \gg H;\ G \gg N;\ a \in H]\!] \Longrightarrow N \cdot a \subseteq N \diamond_G H$

⟨*proof*⟩

lemma (**in** *Group*) *smult-commute-sg-nsg*:⟦G » H; G ▷ N⟧ ⟹ H ⋄$_G$ N = N ⋄$_G$ H
⟨*proof*⟩

lemma (**in** *Group*) *smult-sg-nsg*:⟦G » H; G ▷ N⟧ ⟹ G » H ⋄$_G$ N
⟨*proof*⟩

lemma (**in** *Group*) *smult-nsg-sg*:⟦G » H; G ▷ N⟧ ⟹ G » N ⋄$_G$ H
⟨*proof*⟩

lemma (**in** *Group*) *Gp-smult-sg-nsg*:⟦G » H; G ▷ N⟧ ⟹ *Group* (*Gp* G (H ⋄$_G$ N))
⟨*proof*⟩

lemma (**in** *Group*) *N-sg-HN*:⟦G » H; G ▷ N⟧ ⟹ *Gp* G (H ⋄$_G$ N) » N
⟨*proof*⟩

lemma (**in** *Group*) *K-absorb-HK*:⟦G » H; G » K; H ⊆ K⟧ ⟹ H ⋄$_G$ K = K
⟨*proof*⟩

lemma (**in** *Group*) *nsg-Gp-nsg*:⟦G » H; G ▷ N; N ⊆ H⟧ ⟹ *Gp* G H ▷ N
⟨*proof*⟩

lemma (**in** *Group*) *Gp-smult-nsg*:⟦G » H; G ▷ N⟧ ⟹ *Gp* G (H ⋄$_G$ N) ▷ N
⟨*proof*⟩

lemma (**in** *Group*) *Gp-smult-nsg1*:⟦G » H; G ▷ N⟧ ⟹ *Gp* G (N ⋄$_G$ H) ▷ N
⟨*proof*⟩

lemma (**in** *Group*) *ZassenhausTr2-3*:⟦G » H; G » $H1$; *Gp* G H ▷ $H1$⟧ ⟹ $H1$ ⊆ H
⟨*proof*⟩

lemma (**in** *Group*) *ZassenhausTr2-4*:⟦G » H; G » $H1$; *Gp* G H ▷ $H1$; h ∈ H; $h1$ ∈ $H1$⟧ ⟹ h · $h1$ · (ϱ h) ∈ $H1$
⟨*proof*⟩

lemma (**in** *Group*) *ZassenhausTr1*:⟦G » H; G » $H1$; G » K; G » $K1$; *Gp* G H ▷ $H1$; *Gp* G K ▷ $K1$⟧ ⟹ $H1$ ⋄$_G$ (H ∩ $K1$) = (H ∩ $K1$) ⋄$_G$ $H1$
⟨*proof*⟩

lemma (**in** *Group*) *ZassenhausTr1-1*:⟦G » H; G » $H1$; G » K; G » $K1$; *Gp* G H ▷ $H1$; *Gp* G K ▷ $K1$⟧ ⟹ G » ($H1$ ⋄$_G$ (H ∩ $K1$))
⟨*proof*⟩

lemma (**in** *Group*) *ZassenhausTr2*:⟦G » H; G » $H1$; G » K; *Gp* G H ▷ $H1$⟧ ⟹ $H1$ ⋄$_G$ (H ∩ K) = (H ∩ K) ⋄$_G$ $H1$

⟨*proof*⟩

lemma (**in** *Group*) *ZassenhausTr2-1*:⟦G » H; G » $H1$; G » K; $Gp\ G\ H \triangleright H1$⟧
$\Longrightarrow G$ » $H1 \diamond_G (H \cap K)$
⟨*proof*⟩

lemma (**in** *Group*) *ZassenhausTr2-2*:⟦G » H; G » $H1$; G » K; G » $K1$;
$Gp\ G\ H \triangleright H1$; $Gp\ G\ K \triangleright K1$⟧ $\Longrightarrow H1 \diamond_G (H \cap K1) \subseteq H1 \diamond_G (H \cap K)$
⟨*proof*⟩

lemma (**in** *Group*) *ZassenhausTr2-5*:⟦G » H; G » $H1$; G » K; G » $K1$; $Gp\ G\ H \triangleright H1$;
$Gp\ G\ K \triangleright K1$; $a \in H1$; $b \in H \cap K1$; $c \in H1$⟧ $\Longrightarrow$
$a \cdot b \cdot c \in H1 \diamond_G (H \cap K1)$
⟨*proof*⟩

lemma (**in** *Group*) *ZassenhausTr2-6*:⟦$u \in carrier\ G$; $v \in carrier\ G$;
$x \in carrier\ G$; $y \in carrier\ G$⟧ $\Longrightarrow$
$(u \cdot v) \cdot (x \cdot y) \cdot (\varrho\ (u \cdot v)) =$
$u \cdot v \cdot x \cdot (\varrho\ v) \cdot (v \cdot y \cdot (\varrho\ v)) \cdot (\varrho\ u)$
⟨*proof*⟩

lemma (**in** *Group*) *ZassenhausTr2-7*:⟦$a \in carrier\ G$; $x \in carrier\ G$; $y \in carrier\ G$⟧
$\Longrightarrow a \cdot (x \cdot y) \cdot (\varrho\ a) = a \cdot x \cdot (\varrho\ a) \cdot (a \cdot y \cdot (\varrho\ a))$
⟨*proof*⟩

lemma (**in** *Group*) *ZassenhausTr3*:⟦G » H; G » $H1$; G » K; G » $K1$; $Gp\ G\ H \triangleright H1$;
$Gp\ G\ K \triangleright K1$⟧ $\Longrightarrow$ $Gp\ G\ (H1 \diamond_G (H \cap K)) \triangleright (H1 \diamond_G (H \cap K1))$
⟨*proof*⟩

lemma (**in** *Group*) *ZassenhausTr3-2*:⟦G » H; G » $H1$; G » K; G » $K1$; $Gp\ G\ H \triangleright H1$;
$Gp\ G\ K \triangleright K1$⟧ $\Longrightarrow G$ » $H1 \diamond_G (H \cap K1) \diamond_G (H \cap K)$
⟨*proof*⟩

lemma (**in** *Group*) *ZassenhausTr3-3*:⟦G » H; G » $H1$; G » K; G » $K1$; $Gp\ G\ H \triangleright H1$;
$Gp\ G\ K \triangleright K1$⟧ $\Longrightarrow (H1 \cap K) \diamond_G (H \cap K1) = (K1 \cap H) \diamond_G (K \cap H1)$
⟨*proof*⟩

lemma (**in** *Group*) *ZassenhausTr3-4*:⟦G » H; G » $H1$; G » K; G » $K1$; $Gp\ G\ H \triangleright H1$;
$Gp\ G\ K \triangleright K1$; $g \in H \cap K$; $h \in H \cap K1$⟧ $\Longrightarrow g \cdot h \cdot (\varrho\ g) \in H \cap K1$
⟨*proof*⟩

lemma (**in** *Group*) *ZassenhausTr3-5*:⟦G » H; G » $H1$; G » K; G » $K1$; $Gp\ G\ H \triangleright H1$;

$Gp\ G\ K \triangleright K1\rrbracket \Longrightarrow (Gp\ G\ (H \cap K)) \triangleright (H1 \cap K) \diamond_G (H \cap K1)$
⟨*proof*⟩

lemma (**in** *Group*) *ZassenhausTr4*:⟦$G \gg H$; $G \gg H1$; $G \gg K$; $G \gg K1$; $Gp\ G\ H \triangleright H1$;
$Gp\ G\ K \triangleright K1\rrbracket \Longrightarrow (H1 \diamond_G (H \cap K1)) \diamond_G (H1 \diamond_G (H \cap K)) = H1 \diamond_G (H \cap K)$
⟨*proof*⟩

lemma (**in** *Group*) *ZassenhausTr4-0*: ⟦$G \gg H$; $G \gg H1$; $G \gg K$; $G \gg K1$; $Gp\ G\ H \triangleright H1$;
$Gp\ G\ K \triangleright K1\rrbracket \Longrightarrow\ H1 \diamond_G (H \cap K) = (H1 \diamond_G (H \cap K1)) \diamond_G (H \cap K)$
⟨*proof*⟩

lemma (**in** *Group*) *ZassenhausTr4-1*:⟦$G \gg H$; $(Gp\ G\ H) \triangleright H1$; $(Gp\ G\ H) \gg (H \cap K)$⟧
$\Longrightarrow (Gp\ G\ (H1 \diamond_G (H \cap K))) \triangleright H1$
⟨*proof*⟩

3.7 Homomorphism

lemma *gHom*: ⟦*Group F*; *Group G*; $f \in gHom\ F\ G$; $x \in carrier\ F$;
$y \in carrier\ F\rrbracket \Longrightarrow f\ (x \cdot_F y) = (f\ x) \cdot_G (f\ y)$
⟨*proof*⟩

lemma *gHom-mem*:⟦*Group F*; *Group G*; $f \in gHom\ F\ G$; $x \in carrier\ F\rrbracket \Longrightarrow$
$(f\ x) \in carrier\ G$
⟨*proof*⟩

lemma *gHom-func*:⟦*Group F*; *Group G*; $f \in gHom\ F\ G\rrbracket \Longrightarrow$
$f \in carrier\ F \to carrier\ G$
⟨*proof*⟩

lemma *gHomcomp*:⟦*Group F*; *Group G*; *Group H*; $f \in gHom\ F\ G$; $g \in gHom\ G\ H\rrbracket$
$\Longrightarrow (g \circ_F f) \in gHom\ F\ H$
⟨*proof*⟩

lemma *gHom-comp-gsurjec*:⟦*Group F*; *Group G*; *Group H*; $gsurj_{F,G}\ f$;
$gsurj_{G,H}\ g\rrbracket \Longrightarrow gsurj_{F,H}\ (g \circ_F f)$
⟨*proof*⟩

lemma *gHom-comp-ginjec*:⟦*Group F*; *Group G*; *Group H*; $ginj_{F,G}\ f$; $ginj_{G,H}\ g\rrbracket$
$\Longrightarrow$
$ginj_{F,H}\ (g \circ_F f)$
⟨*proof*⟩

lemma *ghom-unit-unit*:⟦*Group F*; *Group G*; $f \in gHom\ F\ G$ ⟧ ⟹
$f\ (\mathbf{1}_F) = \mathbf{1}_G$
⟨*proof*⟩

lemma *ghom-inv-inv*:⟦*Group F*; *Group G*; $f \in gHom\ F\ G$; $x \in carrier\ F$⟧ ⟹
$f\ (\varrho_F\ x) = \varrho_G\ (f\ x)$
⟨*proof*⟩

lemma *ghomTr3*:⟦*Group F*; *Group G*; $f \in gHom\ F\ G$; $x \in carrier\ F$;
$y \in carrier\ F$; $f\ (x \cdot_F (\varrho_F\ y)) = \mathbf{1}_G$ ⟧ ⟹ $f\ x = f\ y$
⟨*proof*⟩

lemma *iim-nonempty*:⟦*Group F*; *Group G*; $f \in gHom\ F\ G$; $G \lessdot K$⟧ ⟹
$(iim\ F\ G\ f\ K) \neq \{\}$
⟨*proof*⟩

lemma *ghomTr4*:⟦*Group F*; *Group G*; $f \in gHom\ F\ G$; $G \lessdot K$⟧ ⟹
$F \lessdot (iim\ F\ G\ f\ K)$
⟨*proof*⟩

lemma (**in** *Group*) *IdTr0*: $idmap\ (carrier\ G) \in gHom\ G\ G$
⟨*proof*⟩

abbreviation
$IDMAP\ (\langle(I\_)\rangle\ [999]1000)$ **where**
$I_F == idmap\ (carrier\ F)$

abbreviation
$INVFUN\ (\langle(3Ifn\ -\ -\ -)\rangle\ [88,88,89]88)$ **where**
$Ifn\ F\ G\ f == invfun\ (carrier\ F)\ (carrier\ G)\ f$

lemma *IdTr1*:⟦*Group F*; $x \in carrier\ F$⟧ ⟹ $(I_F)\ x = x$
⟨*proof*⟩

lemma *IdTr2*:*Group F* ⟹ $gbij_{F,F}\ (I_F)$
⟨*proof*⟩

lemma *Id-l-unit*:⟦*Group G*; $gbij_{G,G}\ f$⟧ ⟹ $I_G \circ_G f = f$
⟨*proof*⟩

3.8 Gkernel

lemma *gkernTr1*:⟦*Group F*; *Group G*; $f \in gHom\ F\ G$; $x \in gker_{F,G}\ f$⟧ ⟹
$x \in carrier\ F$
⟨*proof*⟩

lemma *gkernTr1-1*:⟦*Group F*; *Group G*; $f \in gHom\ F\ G$⟧ ⟹ $gker_{F,G}\ f \subseteq carrier\ F$
⟨*proof*⟩

lemma *gkernTr2*:⟦*Group F*; *Group G*; $f \in gHom\ F\ G$; $x \in gker_{F,G}\ f$; $y \in gker_{F,G}\ f$⟧
$\Longrightarrow (x \cdot_F y) \in gker_{F,G}\ f$
⟨*proof*⟩

lemma *gkernTr3*:⟦*Group F*; *Group G*; $f \in gHom\ F\ G$; $x \in gker_{F,G}\ f$⟧ ⟹
$(\varrho_F\ x) \in gker_{F,G}\ f$
⟨*proof*⟩

lemma *gkernTr6*:⟦*Group F*; *Group G*; $f \in gHom\ F\ G$⟧ ⟹ $(\mathbf{1}_F) \in gker_{F,G}\ f$
⟨*proof*⟩

lemma *gkernTr7*:⟦*Group F*; *Group G*; $f \in gHom\ F\ G$ ⟧ ⟹ $F \lessdot gker_{F,G}\ f$
⟨*proof*⟩

lemma *gker-normal*:⟦*Group F*; *Group G*; $f \in gHom\ F\ G$⟧ ⟹ $F \triangleright gker_{F,G}\ f$
⟨*proof*⟩

lemma *Group-coim*:⟦*Group F*; *Group G*; $f \in gHom\ F\ G$⟧ ⟹ *Group* (F / $gker_{F,G}\ f$)
⟨*proof*⟩

lemma *gkern1*:⟦*Group F*; *Ugp E*; $f \in gHom\ F\ E$⟧ ⟹ $gker_{F,E}\ f$ = *carrier F*
⟨*proof*⟩

lemma *gkern2*:⟦*Group F*; *Group G*; $f \in gHom\ F\ G$; $ginj_{F,G}\ f$⟧ ⟹
$gker_{F,G}\ f = \{\mathbf{1}_F\}$
⟨*proof*⟩

lemma *gkernTr9*:⟦*Group F*; *Group G*; $f \in gHom\ F\ G$; $a \in$ *carrier F*; $b \in$ *carrier F*⟧
⟹ $((gker_{F,G}\ f) \cdot_F a = (gker_{F,G}\ f) \cdot_F b) = (f\ a = f\ b)$

⟨*proof*⟩

lemma *gkernTr11*:⟦*Group F*; *Group G*; $f \in gHom\ F\ G$; $a \in$ *carrier F*⟧ ⟹
$(iim\ F\ G\ f\ \{f\ a\}) = (gker_{F,G}\ f) \cdot_F a$
⟨*proof*⟩

lemma *gbij-comp-bij*:⟦*Group F*; *Group G*; *Group H*; $gbij_{F,G}\ f$; $gbij_{G,H}\ g$⟧
⟹ $gbij_{F,H}\ (g \circ_F f)$
⟨*proof*⟩

lemma *gbij-automorph*:⟦*Group G*; $gbij_{G,G}\ f$; $gbij_{G,G}\ g$⟧ ⟹
$gbij_{G,G}\ (g \circ_G f)$
⟨*proof*⟩

lemma *l-unit-gHom*:$[\![Group\ F;\ Group\ G;\ f \in gHom\ F\ G]\!] \Longrightarrow (I_G) \circ_F f = f$
⟨*proof*⟩

lemma *r-unit-gHom*:$[\![Group\ F;\ Group\ G;\ f \in gHom\ F\ G\ ]\!] \Longrightarrow f \circ_F (I_F) = f$
⟨*proof*⟩

3.9 Image

lemma *inv-gHom*:$[\![Group\ F;\ Group\ G;\ gbij_{F,G}\ f]\!] \Longrightarrow (Ifn\ F\ G\ f) \in gHom\ G\ F$
⟨*proof*⟩

lemma *inv-gbijec-gbijec*:$[\![Group\ F;\ Group\ G;\ gbij_{F,G}\ f]\!] \Longrightarrow gbij_{G,F}\ (Ifn\ F\ G\ f)$
⟨*proof*⟩

lemma *l-inv-gHom*:$[\![Group\ F;\ Group\ G;\ gbij_{F,G}\ f]\!] \Longrightarrow (Ifn\ F\ G\ f) \circ_F f = (I_F)$
⟨*proof*⟩

lemma *img-mult-closed*:$[\![Group\ F;\ Group\ G;\ f \in gHom\ F\ G;\ u \in f\ `(carrier\ F);$
$v \in f\ `(carrier\ F)]\!] \Longrightarrow u \cdot_G v \in f\ `(carrier\ F)$
⟨*proof*⟩

lemma *img-unit-closed*:$[\![Group\ F;\ Group\ G;\ f \in gHom\ F\ G]\!] \Longrightarrow$
$\mathbf{1}_G \in f\ `(carrier\ F)$
⟨*proof*⟩

lemma *imgTr7*:$[\![Group\ F;\ Group\ G;\ f \in gHom\ F\ G;\ u \in f\ `(carrier\ F)]\!]$
$\Longrightarrow \varrho_G\ u\ \in f\ `(carrier\ F)$
⟨*proof*⟩

lemma *imgTr8*:$[\![Group\ F;\ Group\ G;\ f \in gHom\ F\ G;\ F \lhd H;\ u \in f\ `\ H;$
$v \in f`\ H\ ]\!] \Longrightarrow u \cdot_G v \in f\ `\ H$
⟨*proof*⟩

lemma *imgTr9*:$[\![Group\ F;\ Group\ G;\ f \in gHom\ F\ G;\ F \lhd H;\ u \in f\ `\ H]\!] \Longrightarrow$
$\varrho_G\ u \in f\ `\ H$
⟨*proof*⟩

lemma *imgTr10*:$[\![Group\ F;\ Group\ G;\ f \in gHom\ F\ G;\ F \lhd H]\!] \Longrightarrow \mathbf{1}_G \in f\ `\ H$
⟨*proof*⟩

lemma *imgTr11*:$[\![Group\ F;\ Group\ G;\ f \in gHom\ F\ G;\ F \lhd H]\!] \Longrightarrow G \lhd (f\ `\ H)$
⟨*proof*⟩

lemma *sg-gimg*:$[\![Group\ F;\ Group\ G;\ f \in gHom\ F\ G\ ]\!] \Longrightarrow G \lhd f`(carrier\ F)$
⟨*proof*⟩

lemma *Group-Img*:$[\![Group\ F;\ Group\ G;\ f \in gHom\ F\ G\ ]\!] \Longrightarrow Group\ (Img_{F,G}\ f)$
⟨*proof*⟩

lemma *Img-carrier*:⟦*Group F*; *Group G*; $f \in gHom\ F\ G$ ⟧ ⟹
carrier $(Img_{F,G}\ f) = f$ ‘ (*carrier F*)
⟨*proof*⟩

lemma *hom-to-Img*:⟦*Group F*; *Group G*; $f \in gHom\ F\ G$⟧ ⟹ $f \in gHom\ F\ (Img_{F,G}\ f)$
⟨*proof*⟩

lemma *gker-hom-to-img*:⟦*Group F*; *Group G*; $f \in gHom\ F\ G$ ⟧ ⟹
$gker_{F,(Img_{F,G}\ f)}\ f = gker_{F,G}\ f$
⟨*proof*⟩

lemma *Pj-im-subg*:⟦*Group G*; G » H; $G \triangleright K$; $K \subseteq H$⟧ ⟹
Pj G K ‘ *H* = *carrier* ((*Gp G H*) / *K*)
⟨*proof*⟩

lemma (**in** *Group*) *subg-Qsubg*:⟦G » H; $G \triangleright K$; $K \subseteq H$⟧ ⟹
(*G* / *K*) » *carrier* ((*Gp G H*) / *K*)
⟨*proof*⟩

3.10 Induced homomorphisms

lemma *inducedhomTr*:⟦*Group F*; *Group G*; $f \in gHom\ F\ G$;
$S \in$ *set-rcs F* $(gker_{F,G}\ f)$; *s1* ∈ *S*; *s2* ∈ *S* ⟧ ⟹ *f s1* = *f s2*
⟨*proof*⟩

definition
induced-ghom :: [(*'a*, *'more*) *Group-scheme*, (*'b*, *'more1*) *Group-scheme*,
(*'a* ⇒ *'b*)] ⇒ (*'a set* ⇒ *'b*) **where**
induced-ghom F G f = (λ*X*∈ (*set-rcs F* $(gker_{F,G}\ f)$). *f* (*SOME* *x*. *x* ∈ *X*))

abbreviation
INDUCED-GHOM :: [*'a* ⇒ *'b*, (*'a*, *'m*) *Group-scheme*, (*'b*, *'m1*) *Group-scheme*]
⇒ (*'a set* ⇒ *'b*) (‹(*3-¨-,-*)› [*82*,*82*,*83*]*82*) **where**
$f^{¨}_{F,G}$ == *induced-ghom F G f*

lemma *induced-ghom-someTr*:⟦*Group F*; *Group G*; $f \in gHom\ F\ G$;
$X \in$ *set-rcs F* $(gker_{F,G}\ f)$⟧ ⟹ *f* (*SOME xa*. *xa* ∈ *X*) ∈ *f* ‘(*carrier F*)
⟨*proof*⟩

lemma *induced-ghom-someTr1*:⟦*Group F*; *Group G*; $f \in gHom\ F\ G$; $a \in$ *carrier F*⟧ ⟹
f (*SOME xa*. *xa* ∈ $(gker_{F,G}\ f)$ ·$_F$ *a*) = *f a*
⟨*proof*⟩

lemma *inducedHOMTr0*:⟦*Group F*; *Group G*; $f \in gHom\ F\ G$; $a \in$ *carrier F*⟧ ⟹
$(f^{¨}_{F,G})$ $((gker_{F,G}\ f)$ ·$_F$ *a*) = *f a*

⟨*proof*⟩

lemma *inducedHOMTr0-1*:⟦*Group F*; *Group G*; $f \in gHom\ F\ G$⟧ ⟹
$(f^{\cdot\cdot}{}_{F,G}) \in$ *set-rcs F* $(gker_{F,G}\ f) \to$ *carrier G*
⟨*proof*⟩

lemma *inducedHOMTr0-2*:⟦*Group F*; *Group G*; $f \in gHom\ F\ G$⟧ ⟹
$(f^{\cdot\cdot}{}_{F,G}) \in$ *set-rcs F* $(gker_{F,G}\ f) \to f\ `\ (carrier\ F)$
⟨*proof*⟩

lemma *inducedHom*:⟦*Group F*; *Group G*; $f \in gHom\ F\ G$⟧ ⟹
$(f^{\cdot\cdot}{}_{F,G}) \in gHom\ (F/(gker_{F,G}\ f))\ G$
⟨*proof*⟩

lemma *induced-ghom-ginjec*: ⟦*Group F*; *Group G*; $f \in gHom\ F\ G$ ⟧ ⟹
$ginj_{(F/(gker_{F,G}\ f)),G}\ (f^{\cdot\cdot}{}_{F,G})$
⟨*proof*⟩

lemma *inducedhomgsurjec*:⟦*Group F*; *Group G*; $gsurj_{F,G}\ f$⟧ ⟹
$gsurj_{(F/(gker_{F,G}\ f)),G}\ (f^{\cdot\cdot}{}_{F,G})$
⟨*proof*⟩

lemma *homomtr*: ⟦*Group F*; *Group G*; $f \in gHom\ F\ G$⟧ ⟹
$(f^{\cdot\cdot}{}_{F,G}) \in gHom\ (F\ /\ (gker_{F,G}\ f))\ (Img_{F,G}\ f)$
⟨*proof*⟩

lemma *homom2img*: ⟦*Group F*; *Group G*; $f \in gHom\ F\ G$ ⟧ ⟹
$(f^{\cdot\cdot}{}_{F,(Img_{F,G}\ f)}) \in gHom\ (F\ /\ (gker_{F,G}\ f))\ (Img_{F,G}\ f)$
⟨*proof*⟩

lemma *homom2img1*:⟦*Group F*; *Group G*; $f \in gHom\ F\ G$; $X \in$ *set-rcs F* $(gker_{F,G}\ f)$⟧
⟹ $(f^{\cdot\cdot}{}_{F,(Img_{F,G}\ f)})\ X = (f^{\cdot\cdot}{}_{F,G})\ X$
⟨*proof*⟩

3.10.1 Homomorphism therems

definition
iota :: $('a, 'm)$ *Group-scheme* ⇒ $('a \Rightarrow 'a)$

$(‹(\iota\_)› [1000]999)$ **where**
$\iota_F = (\lambda x \in carrier\ F.\ x)$

lemma *iotahomTr0*:⟦*Group G*; $G \lessdot H$; $h \in H$ ⟧ ⟹ $(\iota_{(Gp\ G\ H)})\ h = h$
⟨*proof*⟩

lemma *iotahom*:⟦*Group G*; $G \lessdot H$; $G \triangleright N$⟧ ⟹

$$\iota_{(Gp\ G\ H)} \in gHom\ (Gp\ G\ H)\ (Gp\ G\ (H \diamond_G N))$$
⟨*proof*⟩

lemma *iotaTr0*: ⟦*Group G*; *G » H*; *G ▷ N*⟧ ⟹
$$ginj_{(Gp\ G\ H),(Gp\ G\ (H \diamond_G N))}\ (\iota_{(Gp\ G\ H)})$$
⟨*proof*⟩

theorem *homomthm1*:⟦*Group F*; *Group G*; *f ∈ gHom F G* ⟧ ⟹
$$gbij_{(F/\ (gkernel\ F\ G\ f)),\ (Gimage\ F\ G\ f)}\ (f^{\cdot\cdot}{}_{F,\ (Gimage\ F\ G\ f)})$$
⟨*proof*⟩

lemma *isomTr0* [*simp*]:*Group F* ⟹ *F ≅ F*
⟨*proof*⟩

lemma *isomTr1*:⟦*Group F*; *Group G*; *F ≅ G* ⟧ ⟹ *G ≅ F*
⟨*proof*⟩

lemma *isomTr2*:⟦*Group F*; *Group G*; *Group H*; *F ≅ G*; *G ≅ H* ⟧ ⟹ *F ≅ H*
⟨*proof*⟩

lemma *gisom1*: ⟦*Group F*; *Group G*; *f ∈ gHom F G* ⟧ ⟹
$$(F/\ (gker_{F,G}\ f)) \cong (Img_{F,G}\ f)$$
⟨*proof*⟩

lemma *homomth2Tr0*: ⟦*Group F*; *Group G*; *f ∈ gHom F G*; *G ▷ N*⟧ ⟹
$$F \triangleright (iim\ F\ G\ f\ N)$$
⟨*proof*⟩

lemma *kern-comp-gHom*:⟦*Group F*; *Group G*; $gsurj_{F,G}\ f$; *G ▷ N*⟧ ⟹
$$gker_{F,\ (G/N)}\ ((Pj\ G\ N) \circ_F f) = iim\ F\ G\ f\ N$$
⟨*proof*⟩

lemma *QgrpUnit-1*:⟦*Group G*; *Ugp E*; *G ▷ H*; *(G / H) ≅ E* ⟧ ⟹ *carrier G = H*
⟨*proof*⟩

lemma *QgrpUnit-2*:⟦*Group G*; *Ugp E*; *G ▷ H*; *carrier G = H*⟧ ⟹ *(G/H) ≅ E*
⟨*proof*⟩

lemma *QgrpUnit-3*:⟦*Group G*; *Ugp E*; *G » H*; *G » H1*; *(Gp G H) ▷ H1*;
((Gp G H) / H1) ≅ E ⟧ ⟹ *H = H1*
⟨*proof*⟩

lemma *QgrpUnit-4*:⟦*Group G*; *Ugp E*; *G » H*; *G » H1*; *(Gp G H) ▷ H1*;
¬ *((Gp G H) / H1) ≅ E* ⟧ ⟹ *H ≠ H1*
⟨*proof*⟩

definition
Qmp :: [('*a*, '*m*) *Group-scheme*, '*a set*, '*a set*] ⇒ ('*a set* ⇒ '*a set*) **where**

$Qmp\ G\ H\ N = (\lambda X \in\ set\text{-}rcs\ G\ H.\ \{z.\ \exists\ x \in X.\ \exists\ y \in N.\ (y \cdot_G x = z)\})$

abbreviation
$QP :: [\text{-}, 'a\ set, 'a\ set] \Rightarrow ('a\ set \Rightarrow 'a\ set)$
$(\langle(3Qm\_\ \_,\_)\rangle\ [82,82,83]82)$ **where**
$Qm_{G\ H,N} == Qmp\ G\ H\ N$

lemma (**in** *Group*) *QmpTr0*:$[\![G \lhd\lhd H;\ G \lhd\lhd N;\ H \subseteq N\ ;\ a \in carrier\ G]\!] \Longrightarrow$
$Qmp\ G\ H\ N\ (H \cdot a) = (N \cdot a)$
⟨*proof*⟩

lemma (**in** *Group*) *QmpTr1*:$[\![G \lhd\lhd H;\ G \lhd\lhd N;\ H \subseteq N;\ a \in carrier\ G;\ b \in carrier\ G;$
$H \cdot a = H \cdot b]\!] \Longrightarrow N \cdot a = N \cdot b$
⟨*proof*⟩

lemma (**in** *Group*) *QmpTr2*:$[\![G \lhd\lhd H;\ G \lhd\lhd N;\ H \subseteq N\ ;\ X \in carrier\ (G/H)]\!]$
$\Longrightarrow (Qmp\ G\ H\ N)\ X \in carrier\ (G/N)$
⟨*proof*⟩

lemma (**in** *Group*) *QmpTr2-1*:$[\![G \lhd\lhd H;\ G \lhd\lhd N;\ H \subseteq N\ ]\!] \Longrightarrow$
$Qmp\ G\ H\ N \in carrier\ (G/H) \to carrier\ (G/N)$
⟨*proof*⟩

lemma (**in** *Group*) *QmpTr3*:$[\![G \lhd H;\ G \lhd N;\ H \subseteq N;\ X \in carrier\ (G/H);$
$Y \in carrier\ (G/H)]\!] \Longrightarrow$
$(Qmp\ G\ H\ N)\ (c\text{-}top\ G\ H\ X\ Y) = c\text{-}top\ G\ N\ ((Qmp\ G\ H\ N)\ X)\ ((Qmp\ G\ H\ N)\ Y)$
⟨*proof*⟩

lemma (**in** *Group*) *Gp-s-mult-nsg*:$[\![G \lhd H;\ G \lhd N;\ H \subseteq N;\ a \in N\ ]\!] \Longrightarrow$
$H \cdot_{(Gp\ G\ N)} a = H \cdot a$
⟨*proof*⟩

lemma (**in** *Group*) *QmpTr5*:$[\![G \lhd H;\ G \lhd N;\ H \subseteq N;\ X \in carrier\ (G/H);$
$Y \in carrier\ (G/H)\ ]\!] \Longrightarrow (Qmp\ G\ H\ N)\ (\ X \cdot_{(G\ /\ H)} Y) =$
$((Qmp\ G\ H\ N)\ X) \cdot_{(G\ /\ N)} ((Qmp\ G\ H\ N)\ Y)$
⟨*proof*⟩

lemma (**in** *Group*) *QmpTr*:$[\![G \lhd H;\ G \lhd N;\ H \subseteq N\ ]\!] \Longrightarrow$
$(Qm_{G\ H,N}) \in gHom\ (G\ /\ H)\ (G\ /\ N)$
⟨*proof*⟩

lemma (**in** *Group*) *Qmpgsurjec*:$[\![G \lhd H;\ G \lhd N;\ H \subseteq N\ ]\!] \Longrightarrow$

$$gsurj_{(G\ /\ H),(G\ /\ N)}\ (Qm_{G\ H,N})$$

⟨*proof*⟩

lemma (**in** *Group*) *gkerQmp*:⟦$G \triangleright H$; $G \triangleright N$; $H \subseteq N$ ⟧ ⟹
$gker_{(G\ /\ H),(G\ /\ N)}\ (Qm_{G\ H,N}) = carrier\ ((Gp\ G\ N)/\ H)$

⟨*proof*⟩

theorem (**in** *Group*) *homom2*:⟦$G \triangleright H$; $G \triangleright N$; $H \subseteq N$⟧ ⟹
$gbij_{((G/H)/(carrier\ ((Gp\ G\ N)/H))),(G/N)}\ ((Qm_{G\ H,N})^{\cdot\cdot}{}_{(G/H),(G/N)})$

⟨*proof*⟩

3.11 Isomorphims

theorem (**in** *Group*) *isom2*:⟦$G \triangleright H$; $G \triangleright N$; $H \subseteq N$⟧ ⟹
$((G/H)/(carrier\ ((Gp\ G\ N)/H))) \cong\ (G/N)$

⟨*proof*⟩

theorem *homom3*:⟦ *Group* F; *Group* G; $G \triangleright N$; $gsurj_{F,G}\ f$;
$N1 = (iim\ F\ G\ f)\ N$ ⟧ ⟹ $(F\ /\ N1) \cong (G\ /\ N)$

⟨*proof*⟩

lemma (**in** *Group*) *homom3Tr1*:⟦$G \twoheadrightarrow H$; $G \triangleright N$⟧ ⟹ $H \cap N =$
$gker_{(Gp\ G\ H),((Gp\ G\ (H \diamond_G N))/N)}$
$((Pj\ (Gp\ G\ (H \diamond_G N))\ N) \circ_{(Gp\ G\ H)}\ (\iota_{(Gp\ G\ H)}))$

⟨*proof*⟩

3.11.1 An automorphism groups

definition
automg :: - ⇒
(⦇ *carrier* :: ('*a* ⇒ '*a*) *set*, *top* :: ['*a* ⇒ '*a*,'*a* ⇒ '*a*] ⇒ ('*a* ⇒ '*a*),
iop :: ('*a* ⇒ '*a*) ⇒ ('*a* ⇒ '*a*), *one* :: ('*a* ⇒ '*a*)⦈ **where**
automg G = ⦇ *carrier* = {*f*. $gbij_{G,G}\ f$},
top = λ*g*∈{*f*. $gbij_{G,G}\ f$}. λ*f*∈{*f*. $gbij_{G,G}\ f$}. ($g \circ_G f$),
iop = λ*f*∈{*f*. $gbij_{G,G}\ f$}. (*Ifn G G f*), *one* = I_G ⦈

lemma *automgroupTr1*:⟦*Group* G; $gbij_{G,G}\ f$; $gbij_{G,G}\ g$; $gbij_{G,G}\ h$⟧ ⟹
$(h \circ_G g) \circ_G f = h \circ_G (g \circ_G f)$

⟨*proof*⟩

lemma *automgroup*:*Group* G ⟹ *Group* (*automg G*)

⟨*proof*⟩

3.11.2 Complete system of representatives

definition
gcsrp :: - ⇒ '*a set* ⇒ '*a set* ⇒ *bool* **where**
gcsrp G H S == ∃*f*. (*bij-to f* (*set-rcs G H*) *S*)

definition
gcsrp-map::- ⇒ *'a set* ⇒ *'a set* ⇒ *'a* **where**
gcsrp-map G H == λ*X*∈(*set-rcs G H*). *SOME x*. *x* ∈ *X*

lemma (**in** *Group*) *gcsrp-func*:*G* » *H* ⟹ *gcsrp-map G H* ∈ *set-rcs G H* → *UNIV*
⟨*proof*⟩

lemma (**in** *Group*) *gcsrp-func1*:*G* » *H* ⟹
gcsrp-map G H ∈ *set-rcs G H* → (*gcsrp-map G H*) ‘ (*set-rcs G H*)
⟨*proof*⟩

lemma (**in** *Group*) *gcsrp-map-bij*:*G* » *H* ⟹
bij-to (*gcsrp-map G H*) (*set-rcs G H*) ((*gcsrp-map G H*) ‘(*set-rcs G H*))
⟨*proof*⟩

lemma (**in** *Group*) *image-gcsrp*:*G* » *H* ⟹
gcsrp G H ((*gcsrp-map G H*) ‘(*set-rcs G H*))
⟨*proof*⟩

lemma (**in** *Group*) *gcsrp-exists*:*G* » *H* ⟹ ∃ *S*. *gcsrp G H S*
⟨*proof*⟩

definition
gcsrp-top :: [- , *'a set*] ⇒ *'a* ⇒ *'a* ⇒ *'a* **where**
gcsrp-top G H == λ*x* ∈ ((*gcsrp-map G H*) ‘(*set-rcs G H*)).
λ*y* ∈ ((*gcsrp-map G H*) ‘(*set-rcs G H*)).
gcsrp-map G H
(*c-top G H*
((*invfun* (*set-rcs G H*) ((*gcsrp-map G H*) ‘(*set-rcs G H*)) (*gcsrp-map G H*)) *x*)
((*invfun* (*set-rcs G H*) ((*gcsrp-map G H*) ‘(*set-rcs G H*)) (*gcsrp-map G H*)) *y*))

definition
gcsrp-iop::[- , *'a set*] ⇒ *'a* ⇒ *'a* **where**
gcsrp-iop G H = (λ*x* ∈ ((*gcsrp-map G H*) ‘(*set-rcs G H*)).
gcsrp-map G H
(*c-iop G H*
((*invfun* (*set-rcs G H*) ((*gcsrp-map G H*) ‘(*set-rcs G H*)) (*gcsrp-map G H*))
x)))

definition
gcsrp-one::[- , *'a set*] ⇒ *'a* **where**
gcsrp-one G H = *gcsrp-map G H H*

definition
Gcsrp :: - ⇒ *'a set* ⇒ *'a Group* **where**
Gcsrp G N = (|*carrier* = (*gcsrp-map G N*) ‘(*set-rcs G N*),
top = *gcsrp-top G N*, *iop* = *gcsrp-iop G N*, *one* = *gcsrp-one G N*|)

lemma (**in** *Group*) *gcsrp-top-closed*:⟦*Group G*; *G* ▷ *N*;
a ∈ ((*gcsrp-map G N*) ‘(*set-rcs G N*)); *b* ∈ ((*gcsrp-map G N*) ‘(*set-rcs G N*))⟧
⟹ *gcsrp-top G N a b* ∈ (*gcsrp-map G N*) ‘(*set-rcs G N*)
⟨*proof*⟩

lemma (**in** *Group*) *gcsrp-tassoc*:⟦*Group G*; *G* ▷ *N*;
a ∈ ((*gcsrp-map G N*) ‘(*set-rcs G N*));
b ∈ ((*gcsrp-map G N*) ‘(*set-rcs G N*));
c ∈ ((*gcsrp-map G N*) ‘(*set-rcs G N*))⟧ ⟹
(*gcsrp-top G N* (*gcsrp-top G N a b*) *c*) =
(*gcsrp-top G N a* (*gcsrp-top G N b c*))
⟨*proof*⟩

lemma (**in** *Group*) *gcsrp-l-one*:⟦*Group G*; *G* ▷ *N*;
a ∈ ((*gcsrp-map G N*) ‘(*set-rcs G N*))⟧ ⟹
(*gcsrp-top G N* (*gcsrp-one G N*) *a*) = *a*
⟨*proof*⟩

lemma (**in** *Group*) *gcsrp-l-i*:⟦*G* ▷ *N*; *a* ∈ ((*gcsrp-map G N*) ‘(*set-rcs G N*))⟧ ⟹
gcsrp-top G N (*gcsrp-iop G N a*) *a* = *gcsrp-one G N*
⟨*proof*⟩

lemma (**in** *Group*) *gcsrp-i-closed*:⟦*G* ▷ *N*; *a* ∈ ((*gcsrp-map G N*) ‘(*set-rcs G N*))⟧
⟹ *gcsrp-iop G N a* ∈ ((*gcsrp-map G N*) ‘(*set-rcs G N*))
⟨*proof*⟩

lemma (**in** *Group*) *Group-Gcsrp*:*G* ▷ *N* ⟹ *Group* (*Gcsrp G N*)
⟨*proof*⟩

lemma (**in** *Group*) *gcsrp-map-gbijec*:*G* ▷ *N* ⟹
$gbij_{(G/N),\ (Gcsrp\ G\ N)}$ (*gcsrp-map G N*)
⟨*proof*⟩

lemma (**in** *Group*) *Qg-equiv-Gcsrp*:*G* ▷ *N* ⟹ (*G* / *N*) ≅ *Gcsrp G N*
⟨*proof*⟩

3.12 Zassenhaus

we show $H \to H\ N/N$ is gsurjective

lemma (**in** *Group*) *homom4Tr1*:⟦*G* ▷ *N*; *G* » *H*⟧ ⟹ *Group* ((*Gp G* (*H* $\diamond_G$ *N*))
/ *N*)
⟨*proof*⟩

lemma *homom3Tr2*:⟦*Group G*; *G* » *H*; *G* ▷ *N*⟧ ⟹
$gsurj_{(Gp\ G\ H),((Gp\ G\ (H \diamond_G\ N))/N)}$
((*Pj* (*Gp G* (*H* $\diamond_G$ *N*)) *N*) $\circ_{(Gp\ G\ H)}$ ($\iota_{(Gp\ G\ H)}$))
⟨*proof*⟩

theorem *homom4*:$[\![Group\ G;\ G \triangleright N;\ G \lessdot\!\!\cdot H]\!] \Longrightarrow gbij_{((Gp\ G\ H)/(H \cap N)),((Gp\ G\ (H \diamond_G N))\ /\ N)}$
$(((Pj\ (Gp\ G\ (H \diamond_G N))\ N) \circ_{(Gp\ G\ H)} (\iota_{(Gp\ G\ H)}))^{\cdot\cdot}_{(Gp\ G\ H),((Gp\ G\ (H \diamond_G N))\ /\ N)})$

⟨*proof*⟩

lemma (**in** *Group*) *homom4-2*:$[\![G \triangleright N;\ G \lessdot\!\!\cdot H]\!] \Longrightarrow Group\ ((Gp\ G\ H)\ /\ (H \cap N))$
⟨*proof*⟩

lemma *isom4*:$[\![Group\ G;\ G \triangleright N;\ G \lessdot\!\!\cdot H]\!] \Longrightarrow$
$((Gp\ G\ H)/(H \cap N)) \cong\ ((Gp\ G\ (N \diamond_G H))\ /\ N)$
⟨*proof*⟩

lemma *ZassenhausTr5*:$[\![Group\ G;\ G \lessdot\!\!\cdot H;\ G \lessdot\!\!\cdot H1;\ G \lessdot\!\!\cdot K;\ G \lessdot\!\!\cdot K1;\ Gp\ G\ H \triangleright H1;$
$Gp\ G\ K \triangleright K1]\!] \Longrightarrow$
$((Gp\ G\ (H \cap K))/((H1 \cap K) \diamond_G (H \cap K1))) \cong$
$((Gp\ G\ (H1 \diamond_G (H \cap K)))/(H1 \diamond_G (H \cap K1)))$
⟨*proof*⟩

lemma *ZassenhausTr5-1*:$[\![Group\ G;\ G \lessdot\!\!\cdot H;\ G \lessdot\!\!\cdot H1;\ G \lessdot\!\!\cdot K;\ G \lessdot\!\!\cdot K1;\ Gp\ G\ H \triangleright H1;$
$Gp\ G\ K \triangleright K1]\!] \Longrightarrow\ ((Gp\ G\ (K \cap H))/((K1 \cap H) \diamond_G (K \cap H1))) \cong$
$((Gp\ G\ (K1 \diamond_G (K \cap H)))/(K1 \diamond_G (K \cap H1)))$

⟨*proof*⟩

lemma *ZassenhausTr5-2*: $[\![Group\ G;\ G \lessdot\!\!\cdot H;\ G \lessdot\!\!\cdot H1;\ G \lessdot\!\!\cdot K;\ G \lessdot\!\!\cdot K1;\ Gp\ G\ H$
$\triangleright H1;$
$Gp\ G\ K \triangleright K1]\!] \Longrightarrow$
$((Gp\ G\ (H \cap K))/((H1 \cap K) \diamond_G (H \cap K1))) =$
$((Gp\ G\ (K \cap H))/((K1 \cap H) \diamond_G (K \cap H1)))$
⟨*proof*⟩

lemma *ZassenhausTr6-1*:$[\![Group\ G;\ G \lessdot\!\!\cdot H;\ G \lessdot\!\!\cdot H1;\ G \lessdot\!\!\cdot K;\ G \lessdot\!\!\cdot K1;\ Gp\ G\ H \triangleright H1;$
$Gp\ G\ K \triangleright K1]\!] \Longrightarrow Group\ \ (Gp\ G\ (H \cap K)\ /\ (H1 \cap K \diamond_G H \cap K1))$
⟨*proof*⟩

lemma *ZassenhausTr6-2*:$[\![Group\ G;\ G \lessdot\!\!\cdot H;\ G \lessdot\!\!\cdot H1;\ G \lessdot\!\!\cdot K;\ G \lessdot\!\!\cdot K1;\ Gp\ G\ H \triangleright H1;$
$Gp\ G\ K \triangleright K1]\!] \Longrightarrow Group\ (Gp\ G\ (H1 \diamond_G H \cap K)\ /\ (H1 \diamond_G H \cap K1))$
⟨*proof*⟩

lemma *ZassenhausTr6-3*:$[\![Group\ G;\ G \lessdot\!\!\cdot H;\ G \lessdot\!\!\cdot H1;\ G \lessdot\!\!\cdot K;\ G \lessdot\!\!\cdot K1;\ Gp\ G\ H \triangleright H1;$
$Gp\ G\ K \triangleright K1]\!] \Longrightarrow Group\ (Gp\ G\ (K1 \diamond_G K \cap H)\ /\ (K1 \diamond_G K \cap H1))$

⟨*proof*⟩

theorem *Zassenhaus*:⟦*Group G*; *G* » *H*; *G* » *H1*; *G* » *K*; *G* » *K1*; *Gp G H* ▷ *H1*;

Gp G K ▷ *K1*⟧ ⟹ (*Gp G* (*H1* ⋄$_G$ *H* ∩ *K*) / (*H1* ⋄$_G$ *H* ∩ *K1*)) ≅
(*Gp G* (*K1* ⋄$_G$ *K* ∩ *H*) / (*K1* ⋄$_G$ *K* ∩ *H1*))

⟨*proof*⟩

3.13 Chain of groups I

definition
d-gchain :: [- , *nat*, (*nat* ⇒ *'a set*)] ⇒ *bool* **where**
d-gchain G n g = (*if n=0 then G* » *g 0 else* (∀ *l*≤ *n*. *G* » (*g l*) ∧
(∀ *l* ≤ (*n* − *Suc 0*). *g* (*Suc l*) ⊆ *g l*)))

definition
D-gchain :: [- , *nat*, (*nat* ⇒ *'a set*)] ⇒ *bool* **where**
D-gchain G n g = (*if n* = *0 then G* » (*g 0*) *else* (*d-gchain G n g*) ∧
(∀ *l* ≤ (*n* − *Suc 0*). (*g* (*Suc l*)) ⊂ (*g l*)))

definition
td-gchain :: [- , *nat*, (*nat* ⇒ *'a set*)] ⇒ *bool* **where**
td-gchain G n g = (*if n=0 then g 0* = *carrier G* ∧ *g 0* = {**1**$_G$} *else*
d-gchain G n g ∧ *g 0* = *carrier G* ∧ *g n* = {**1**$_G$})

definition
tD-gchain :: [-, *nat*, (*nat* ⇒ *'a set*)] ⇒ *bool* **where**
tD-gchain G n g = (*if n=0 then g 0* = *carrier G* ∧ *g 0* = {**1**$_G$} *else*
D-gchain G n g ∧ (*g 0* = *carrier G*) ∧ (*g n* = {**1**$_G$}))

definition
w-cmpser :: [- , *nat*, (*nat* ⇒ *'a set*)] ⇒ *bool* **where**
w-cmpser G n g = (*if n* = *0 then d-gchain G n g else d-gchain G n g* ∧
(∀ *l* ≤ (*n* − *1*). (*Gp G* (*g l*)) ▷ (*g* (*Suc l*))))

definition
W-cmpser :: [- , *nat*, (*nat* ⇒ *'a set*)] ⇒ *bool* **where**
W-cmpser G n g = (*if n* = *0 then d-gchain G 0 g else D-gchain G n g* ∧
(∀ *l* ≤ (*n* − *1*). (*Gp G* (*g l*)) ▷ (*g* (*Suc l*))))

definition
tw-cmpser :: [- , *nat*, (*nat* ⇒ *'a set*)] ⇒ *bool* **where**
tw-cmpser G n g = (*if n* = *0 then td-gchain G 0 g else td-gchain G n g* ∧

$(\forall\, l \leq (n - 1).\ (Gp\ G\ (g\ l)) \triangleright (g\ (Suc\ l))))$

definition
tW-cmpser :: [- , *nat*, (*nat* ⇒ *'a set*)] ⇒ *bool* **where**
tW-cmpser G n g = (*if n* = *0 then td-gchain G 0 g else tD-gchain G n g* ∧
$(\forall\, l \leq (n - 1).\ (Gp\ G\ (g\ l)) \triangleright (g\ (Suc\ l))))$

definition
Qw-cmpser :: [- , *nat* ⇒ *'a set*] ⇒ (*nat* ⇒ (*'a set*) *Group*) **where**
Qw-cmpser G f l = ((*Gp G* (*f l*)) / (*f* (*Suc l*)))

definition
red-chn :: [- , *nat*, (*nat* ⇒ *'a set*)] ⇒ (*nat* ⇒ *'a set*) **where**
red-chn G n f = (*SOME g*. $g \in \{h.(tW\text{-}cmpser\ G\ (card\ (f\ `\ \{i.\ i \leq n\}) - 1)\ h)$
∧ *h* ‘{*i*. *i* ≤ (*card* (*f* ‘ {*i*. *i* ≤ *n*}) − *1*)} = *f* ‘{*i*. *i* ≤ *n*}})

definition
chain-cutout :: [*nat*, (*nat* ⇒ *'a set*)] ⇒ (*nat* ⇒ *'a set*) **where**
chain-cutout l f = (λ*j*. *f* (*slide l j*))

lemma (**in** *Group*) *d-gchainTr0*:⟦*0* < *n*; *d-gchain G n f*; *k* ≤ (*n* − *1*)⟧
⟹ *f* (*Suc k*) ⊆ *f k*
⟨*proof*⟩

lemma (**in** *Group*) *d-gchain-mem-sg*:*d-gchain G n f* ⟹ ∀ *i*≤ *n*. *G* » (*f i*)
⟨*proof*⟩

lemma (**in** *Group*) *d-gchain-pre*:*d-gchain G* (*Suc n*) *f* ⟹ *d-gchain G n f*
⟨*proof*⟩

lemma (**in** *Group*) *d-gchainTr1*:*0* < *n* ⟶ (∀ *f*. *d-gchain G n f* ⟶
(∀ *l* ≤ *n*. ∀ *j* ≤ *n*. *l* < *j* ⟶ *f j* ⊆ *f l*))
⟨*proof*⟩

lemma (**in** *Group*) *d-gchainTr2*:⟦*0* < *n*; *d-gchain G n f*; *l* ≤ *n*; *j* ≤ *n*; *l* ≤ *j* ⟧
⟹ *f j* ⊆ *f l*
⟨*proof*⟩

lemma (**in** *Group*) *im-d-gchainTr1*:⟦*d-gchain G n f*;
f l ∈ (*f* ‘ {*i*. *i* ≤ *n*}) − {*f 0*}⟧ ⟹
f (*LEAST j*. *f j* ∈ (*f* ‘ {*i*. *i* ≤ *n*}) − {*f 0*}) ∈ (*f* ‘ {*i*. *i* ≤ *n*} − {*f 0*})
⟨*proof*⟩

lemma (**in** *Group*) *im-d-gchainTr1-0*:⟦*d-gchain G n f*;
f l ∈ (*f* ‘ {*i*. *i* ≤ *n*}) − {*f 0*}⟧ ⟹

$$0 < (LEAST\ j.\ f\ j \in (f\ `\ \{i.\ i \leq n\}) - \{f\ 0\})$$

⟨*proof*⟩

lemma (**in** *Group*) *im-d-gchainTr1-1*:
$[\![d\text{-}gchain\ G\ n\ f;\ \exists\ i.\ f\ i \in (f\ `\ \{i.\ i \leq n\}) - \{f\ 0\}]\!] \Longrightarrow$
$f\ (LEAST\ j.\ f\ j \in ((f\ `\ \{i.\ i \leq n\}) - \{f\ 0\})) \in ((f`\ \{i.\ i \leq n\}) - \{f\ 0\})$

⟨*proof*⟩

lemma (**in** *Group*) *im-d-gchainsTr1-2*:
$[\![d\text{-}gchain\ G\ n\ f;\ i \leq n;\ f\ i \in f\ `\{i.\ i \leq n\} - \{f\ 0\}]\!] \Longrightarrow$
$(LEAST\ j.\ f\ j \in (f\ `\{i.\ i \leq n\} - \{f\ 0\})) \leq i$

⟨*proof*⟩

lemma (**in** *Group*) *im-d-gchainsTr1-3*:$[\![d\text{-}gchain\ G\ n\ f;\ \exists\, i \leq n.$
$f\ i \in f`\ \{i.\ i \leq n\} - \{f\ 0\};$
$k < (LEAST\ j.\ f\ j \in (f\ `\{i.\ i \leq n\} - \{f\ 0\}))]\!] \Longrightarrow f\ k = f\ 0$

⟨*proof*⟩

lemma (**in** *Group*) *im-gdchainsTr1-4*:$[\![d\text{-}gchain\ G\ n\ f;$
$\exists\, v{\in}f\ `\{i.\ i \leq n\}.\ v \notin \{f\ 0\};\ i < (LEAST\ j.\ f\ j \in (f\ `\{i.\ i \leq n\}) \wedge$
$f\ j \neq f\ 0)\ ]\!] \Longrightarrow f\ i = f\ 0$

⟨*proof*⟩

lemma (**in** *Group*) *im-d-gchainsTr1-5*:$[\![0 < n;\ d\text{-}gchain\ G\ n\ f;\ i \leq n;$
$f\ i \in (f\ `\ \{i.\ i \leq n\} - \{f\ 0\});\ (LEAST\ j.\ f\ j \in (f\ `\{i.\ i \leq n\} - \{f\ 0\})) = j]\!]$
$\Longrightarrow \quad f\ `\{i.\ i \leq (j - (Suc\ 0))\} = \{f\ 0\}$

⟨*proof*⟩

lemma (**in** *Group*) *im-d-gchains1*:$[\![0 < n;\ d\text{-}gchain\ G\ n\ f;\ i \leq n;$
$f\ i \in (f\ `\ \{i.\ i \leq n\} - \{f\ 0\});$
$(LEAST\ j.\ f\ j \in (f\ `\{i.\ i \leq n\} - \{f\ 0\})) = j\ ]\!] \Longrightarrow$
$f\ `\{i.\ i \leq n\} = \{f\ 0\} \cup \{f\ i\ |i.\ j \leq i \wedge i \leq n\}$

⟨*proof*⟩

lemma (**in** *Group*) *im-d-gchains1-1*:$[\![d\text{-}gchain\ G\ n\ f;\ f\ n \neq f\ 0]\!] \Longrightarrow$
$f\ `\{i.\ i \leq n\} = \{f\ 0\} \cup$
$\{f\ i\ |i.\ (LEAST\ j.\ f\ j \in (f\ `\{i.\ i \leq n\} - \{f\ 0\})) \leq i \wedge i \leq n\}$

⟨*proof*⟩

lemma (**in** *Group*) *d-gchains-leastTr*:$[\![d\text{-}gchain\ G\ n\ f;\ f\ n \neq f\ 0]\!] \Longrightarrow$
$(LEAST\ j.\ f\ j \in (f\ `\{i.\ i \leq n\} - \{f\ 0\})) \in \{i.\ i \leq n\} \wedge$
$f\ \ (LEAST\ j.\ f\ j \in (f\ `\{i.\ i \leq n\} - \{f\ 0\})) \neq f\ 0$

⟨*proof*⟩

lemma (**in** *Group*) *im-d-gchainTr2*:$[\![d\text{-}gchain\ G\ n\ f;\ j \leq n;\ f\ j \neq f\ 0]\!] \Longrightarrow$
$\forall\, i \leq n.\ f\ 0 = f\ i \longrightarrow \neg\ j \leq i$

⟨*proof*⟩

lemma (**in** *Group*) *D-gchain-pre*:$[\![D\text{-}gchain\ G\ (Suc\ n)\ f]\!] \Longrightarrow D\text{-}gchain\ G\ n\ f$

⟨*proof*⟩

lemma (**in** *Group*) *D-gchain0*:⟦*D-gchain G n f*; $i \leq n$; $j \leq n$; $i < j$⟧ ⟹
$f\ j \subset f\ i$
⟨*proof*⟩

lemma (**in** *Group*) *D-gchain1*:*D-gchain G n f* ⟹ *inj-on f* $\{i.\ i \leq n\}$
⟨*proof*⟩

lemma (**in** *Group*) *card-im-D-gchain*:⟦$0 < n$; *D-gchain G n f*⟧
⟹ *card* $(f\ `\{i.\ i \leq n\}) = Suc\ n$
⟨*proof*⟩

lemma (**in** *Group*) *w-cmpser-gr*:⟦$0 < r$; *w-cmpser G r f*; $i \leq r$⟧
⟹ $G \lhd (f\ i)$
⟨*proof*⟩

lemma (**in** *Group*) *w-cmpser-ns*:⟦$0 < r$; *w-cmpser G r f*; $i \leq (r - 1)$⟧ ⟹
$(Gp\ G\ (f\ i)) \triangleright (f\ (Suc\ i))$
⟨*proof*⟩

lemma (**in** *Group*) *w-cmpser-pre*:*w-cmpser G* $(Suc\ n)\ f$ ⟹ *w-cmpser G n f*
⟨*proof*⟩

lemma (**in** *Group*) *W-cmpser-pre*:*W-cmpser G* $(Suc\ n)\ f$ ⟹ *W-cmpser G n f*
⟨*proof*⟩

lemma (**in** *Group*) *td-gchain-n*:⟦*td-gchain G n f*; *carrier* $G \neq \{\mathbf{1}\}$⟧ ⟹ $0 < n$
⟨*proof*⟩

3.14 Existence of reduced chain

lemma (**in** *Group*) *D-gchain-is-d-gchain*:*D-gchain G n f* ⟹ *d-gchain G n f*
⟨*proof*⟩

lemma (**in** *Group*) *joint-d-gchains*:⟦*d-gchain G n f*; *d-gchain G m g*;
$g\ 0 \subseteq f\ n$ ⟧ ⟹ *d-gchain G* $(Suc\ (n + m))$ $(jointfun\ n\ f\ m\ g)$
⟨*proof*⟩

lemma (**in** *Group*) *joint-D-gchains*:⟦*D-gchain G n f*; *D-gchain G m g*;
$g\ 0 \subset f\ n$ ⟧ ⟹ *D-gchain G* $(Suc\ (n + m))$ $(jointfun\ n\ f\ m\ g)$
⟨*proof*⟩

lemma (**in** *Group*) *w-cmpser-is-d-gchain*:*w-cmpser G n f* ⟹ *d-gchain G n f*
⟨*proof*⟩

lemma (**in** *Group*) *joint-w-cmpser*:⟦*w-cmpser G n f*; *w-cmpser G m g*;
$Gp\ G\ (f\ n) \triangleright (g\ 0)$⟧ ⟹ *w-cmpser G* $(Suc\ (n + m))$ $(jointfun\ n\ f\ m\ g)$
⟨*proof*⟩

lemma (**in** *Group*) *W-cmpser-is-D-gchain*:*W-cmpser G n f* $\Longrightarrow$ *D-gchain G n f*
⟨*proof*⟩

lemma (**in** *Group*) *W-cmpser-is-w-cmpser*:*W-cmpser G n f* $\Longrightarrow$ *w-cmpser G n f*
⟨*proof*⟩

lemma (**in** *Group*) *tw-cmpser-is-w-cmpser*:*tw-cmpser G n f* $\Longrightarrow$ *w-cmpser G n f*
⟨*proof*⟩

lemma (**in** *Group*) *tW-cmpser-is-W-cmpser*:*tW-cmpser G n f* $\Longrightarrow$ *W-cmpser G n f*
⟨*proof*⟩

lemma (**in** *Group*) *joint-W-cmpser*:⟦*W-cmpser G n f*; *W-cmpser G m g*;
(Gp G (f n)) ▷ (g 0); *g 0 ⊂ f n*⟧ $\Longrightarrow$
W-cmpser G (Suc (n + m)) (jointfun n f m g)
⟨*proof*⟩

lemma (**in** *Group*) *joint-d-gchain-n0*:⟦*d-gchain G n f*; *d-gchain G 0 g*;
g 0 ⊆ f n ⟧ $\Longrightarrow$ *d-gchain G (Suc n) (jointfun n f 0 g)*
⟨*proof*⟩

lemma (**in** *Group*) *joint-D-gchain-n0*:⟦*D-gchain G n f*; *D-gchain G 0 g*;
g 0 ⊂ f n ⟧ $\Longrightarrow$ *D-gchain G (Suc n) (jointfun n f 0 g)*
⟨*proof*⟩

lemma (**in** *Group*) *joint-w-cmpser-n0*:⟦*w-cmpser G n f*; *w-cmpser G 0 g*;
(Gp G (f n)) ▷ (g 0)⟧ $\Longrightarrow$ *w-cmpser G (Suc n) (jointfun n f 0 g)*
⟨*proof*⟩

lemma (**in** *Group*) *joint-W-cmpser-n0*:⟦*W-cmpser G n f*; *W-cmpser G 0 g*;
(Gp G (f n)) ▷ (g 0); *g 0 ⊂ f n* ⟧ $\Longrightarrow$
W-cmpser G (Suc n) (jointfun n f 0 g)
⟨*proof*⟩

definition
simple-Group :: - ⇒ *bool* **where**
simple-Group G $\longleftrightarrow$ {*N*. *G* » *N*} = {*carrier G*, $\{\mathbf{1}_G\}$}

definition
compseries:: [- , *nat*, *nat* ⇒ *'a set*] ⇒ *bool* **where**
compseries G n f $\longleftrightarrow$ *tW-cmpser G n f* ∧ (*if n = 0 then f 0* = $\{\mathbf{1}_G\}$ *else*
(∀ *i* ≤ (*n* − *1*). (*simple-Group* ((*Gp G* (*f i*))/(*f* (*Suc i*))))))

definition
length-twcmpser :: [- , *nat*, *nat* ⇒ *'a set*] ⇒ *nat* **where**
length-twcmpser G n f = *card* (*f* ‘{*i*. *i* ≤ *n*}) − *Suc 0*

lemma (**in** *Group*) *compseriesTr0*:⟦*compseries G n f*; $i \leq n$⟧ ⟹
$G \gg (f\ i)$
⟨*proof*⟩

lemma (**in** *Group*) *compseriesTr1*:*compseries G n f* ⟹ *tW-cmpser G n f*
⟨*proof*⟩

lemma (**in** *Group*) *compseriesTr2*:*compseries G n f* ⟹ $f\ 0 = carrier\ G$
⟨*proof*⟩

lemma (**in** *Group*) *compseriesTr3*:*compseries G n f* ⟹ $f\ n = \{\mathbf{1}\}$
⟨*proof*⟩

lemma (**in** *Group*) *compseriesTr4*:*compseries G n f* ⟹ *w-cmpser G n f*
⟨*proof*⟩

lemma (**in** *Group*) *im-jointfun1Tr1*:∀ $l \leq n.\ G \gg (f\ l)$ ⟹
$f \in \{i.\ i \leq n\} \to Collect\ (sg\ G)$
⟨*proof*⟩

lemma (**in** *Group*) *Nset-Suc-im*:∀ $l \leq (Suc\ n).\ G \gg (f\ l)$ ⟹
$insert\ (f\ (Suc\ n))\ (f\ `\ \{i.\ i \leq n\}) = f\ `\ \{i.\ i \leq (Suc\ n)\}$
⟨*proof*⟩

definition
NfuncPair-neq-at::[*nat* ⇒ *'a set*, *nat* ⇒ *'a set*, *nat*] ⇒ *bool* **where**
NfuncPair-neq-at f g i ⟷ $f\ i \neq g\ i$

lemma *LeastTr0*:⟦ $(i::nat) < (LEAST\ l.\ P\ (l))$⟧ ⟹ ¬ $P\ (i)$
⟨*proof*⟩

lemma (**in** *Group*) *funeq-LeastTr1*:⟦∀ $l \leq n.\ G \gg f\ l$; ∀ $l \leq n.\ G \gg g\ l$;
$(l :: nat) < (LEAST\ k.\ (NfuncPair\text{-}neq\text{-}at\ f\ g\ k))$ ⟧ ⟹ $f\ l = g\ l$
⟨*proof*⟩

lemma (**in** *Group*) *funeq-LeastTr1-1*:⟦∀ $l \leq (n::nat).\ G \gg f\ l$; ∀ $l \leq n.\ G \gg g\ l$;
$(l :: nat) < (LEAST\ k.\ (f\ k \neq g\ k))$ ⟧ ⟹ $f\ l = g\ l$
⟨*proof*⟩

lemma (**in** *Group*) *Nfunc-LeastTr2-1*:⟦$i \leq n$; ∀ $l \leq n.\ G \gg f\ l$; ∀ $l \leq n.\ G \gg g\ l$;
NfuncPair-neq-at f g i⟧ ⟹
NfuncPair-neq-at f g $(LEAST\ k.\ (NfuncPair\text{-}neq\text{-}at\ f\ g\ k))$
⟨*proof*⟩

lemma (**in** *Group*) *Nfunc-LeastTr2-2*:⟦$i \leq n$; ∀ $l \leq n.\ G \gg f\ l$; ∀ $l \leq n.\ G \gg g\ l$;
NfuncPair-neq-at f g i⟧ ⟹
$(LEAST\ k.\ (NfuncPair\text{-}neq\text{-}at\ f\ g\ k)) \leq i$
⟨*proof*⟩

lemma (**in** *Group*) *Nfunc-LeastTr2-2-1*:⟦$i \leq (n::nat)$; $\forall\, l \leq n.\ G \gg f\ l$;
$\forall\, l \leq n.\ G \gg g\ l$; $f\ i \neq g\ i$⟧ $\Longrightarrow$ $(LEAST\ k.\ (f\ k \neq g\ k)) \leq i$
⟨*proof*⟩

lemma (**in** *Group*) *Nfunc-LeastTr2-3*:⟦$\forall\, l \leq (n::nat).\ G \gg f\ l$; $\forall\, l \leq n.\ G \gg g\ l$;
$i \leq n$; $f\ i \neq g\ i$⟧ $\Longrightarrow$
$f\ (LEAST\ k.\ (f\ k \neq\ g\ k)) \neq g\ (LEAST\ k.\ (f\ k \neq\ g\ k))$
⟨*proof*⟩

lemma (**in** *Group*) *Nfunc-LeastTr2-4*:⟦$\forall\, l \leq (n::nat).\ G \gg f\ l$; $\forall\, l \leq n.\ G \gg g\ l$;
$i \leq n$; $f\ i \neq g\ i$⟧ $\Longrightarrow$$(LEAST\ k.\ (f\ k \neq g\ k)) \leq n$
⟨*proof*⟩

lemma (**in** *Group*) *Nfunc-LeastTr2-5*:⟦$\forall\, l\leq (n::nat).\ G \gg f\ l$; $\forall\, l \leq n.\ G \gg g\ l$;
$\exists\, i \leq n.\ (f\ i \neq g\ i)$⟧ $\Longrightarrow$
$f\ (LEAST\ k.\ (f\ k \neq g\ k)) \neq g\ ((LEAST\ k.\ f\ k \neq g\ k))$
⟨*proof*⟩

lemma (**in** *Group*) *Nfunc-LeastTr2-6*:⟦$\forall\, l \leq (n::nat).\ G \gg f\ l$; $\forall\, l \leq n.\ G \gg g\ l$;
$\exists\, i \leq n.\ (f\ i \neq g\ i)$⟧ $\Longrightarrow$ $(LEAST\ k.\ (f\ k \neq g\ k)) \leq n$
⟨*proof*⟩

lemma (**in** *Group*) *Nfunc-Least-sym*:⟦$\forall\, l \leq (n::nat).\ G \gg f\ l$; $\forall\, l \leq n.\ G \gg g\ l$;
$\exists\, i \leq n.\ (f\ i \neq g\ i)$⟧ $\Longrightarrow$
$(LEAST\ k.\ (f\ k \neq g\ k)) = (LEAST\ k.\ (g\ k \neq f\ k))$
⟨*proof*⟩

lemma *Nfunc-iNJTr*:⟦*inj-on* $g\ \{i.\ i \leq (n::nat)\}$; $i \leq n$; $j \leq n$; $i < j$ ⟧ $\Longrightarrow$ $g\ i \neq g\ j$
⟨*proof*⟩

lemma (**in** *Group*) *Nfunc-LeastTr2-7*:⟦$\forall\, l \leq (n::nat).\ G \gg f\ l$; $\forall\, l \leq n.\ G \gg g\ l$;
inj-on $g\ \{i.\ i \leq n\}$; $\exists\, i \leq n.\ (f\ i \neq g\ i)$;
$f\ k = g\ (LEAST\ k.(f\ k \neq g\ k))$⟧ $\Longrightarrow$$(LEAST\ k.(f\ k \neq g\ k)) < k$
⟨*proof*⟩

lemma (**in** *Group*) *Nfunc-LeastTr2-8*:⟦$\forall\, l \leq n.\ G \gg f\ l$; $\forall\, l \leq n.\ G \gg g\ l$;
inj-on $g\ \{i.\ i \leq n\}$; $\exists\, i \leq n.\ f\ i \neq g\ i$; $f\ `\{i.\ i \leq n\} = g\ `\{i.\ i \leq n\}$⟧
$\Longrightarrow$
$\exists\ k \in (nset\ (Suc\ (LEAST\ i.\ (f\ i \neq g\ i)))\ n).\ f\ k = g\ (LEAST\ i.\ (f\ i \neq g\ i))$
⟨*proof*⟩

lemma (**in** *Group*) *ex-redchainTr1*:⟦*d-gchain* $G\ n\ f$;
D-gchain $G\ (card\ (f\ `\{i.\ i \leq n\})\ -\ Suc\ 0)\ g$;
$g\ `\{i.\ i \leq (card\ (f\ `\{i.\ i \leq n\})\ -\ Suc\ 0)\} = f\ `\{i.\ i \leq n\}$⟧ $\Longrightarrow$
$g\ (card\ (f\ `\{i.\ i \leq n\})\ -\ Suc\ 0) = f\ n$
⟨*proof*⟩

lemma (**in** *Group*) *ex-redchainTr1-1*:⟦*d-gchain G* (*n*::*nat*) *f*;
D-gchain G (*card* (*f* ‘ {*i*. *i* ≤ *n*}) − *Suc 0*) *g*;
g ‘ {*i*. *i* ≤ (*card* (*f* ‘ {*i*. *i* ≤ *n*}) − *Suc 0*)} = *f* ‘ {*i*. *i* ≤ *n*}⟧ ⟹
g 0 = *f 0*
⟨*proof*⟩

lemma (**in** *Group*) *ex-redchainTr2*:*d-gchain G* (*Suc n*) *f*
⟹ *D-gchain G 0* (*constmap* {*0*::*nat*} {*f* (*Suc n*)})
⟨*proof*⟩

lemma (**in** *Group*) *last-mem-excluded*:⟦*d-gchain G* (*Suc n*) *f*; *f n* ≠ *f* (*Suc n*)⟧
⟹
f (*Suc n*) ∉ *f* ‘ {*i*. *i* ≤ *n*}
⟨*proof*⟩

lemma (**in** *Group*) *ex-redchainTr4*:⟦*d-gchain G* (*Suc n*) *f*; *f n* ≠ *f* (*Suc n*)⟧ ⟹
card (*f* ‘ {*i*. *i* ≤ (*Suc n*)}) = *Suc* (*card* (*f* ‘ {*i*. *i* ≤ *n*}))
⟨*proof*⟩

lemma (**in** *Group*) *ex-redchainTr5*:*d-gchain G n f* ⟹ *0* < *card* (*f* ‘ {*i*. *i*≤ *n*})
⟨*proof*⟩

lemma (**in** *Group*) *ex-redchainTr6*:∀ *f*. *d-gchain G n f* ⟶
(∃ *g*. *D-gchain G* (*card* (*f* ‘{*i*. *i* ≤ *n*}) − *1*) *g* ∧
(*g* ‘ {*i*. *i* ≤ (*card* (*f* ‘{*i*. *i* ≤ *n*}) − *1*)} = *f* ‘{*i*. *i* ≤ *n*}))
⟨*proof*⟩

lemma (**in** *Group*) *ex-redchain*:*d-gchain G n f* ⟹
(∃ *g*. *D-gchain G* (*card* (*f* ‘ {*i*. *i* ≤ *n*}) − *1*) *g* ∧
g ‘ {*i*. *i* ≤ (*card* (*f* ‘ {*i*. *i* ≤ *n*}) − *1*)} = *f* ‘ {*i*. *i* ≤ *n*})
⟨*proof*⟩

lemma (**in** *Group*) *const-W-cmpser*:*d-gchain G* (*Suc n*) *f* ⟹
W-cmpser G 0 (*constmap* {*0*::*nat*} {*f* (*Suc n*)})
⟨*proof*⟩

lemma (**in** *Group*) *ex-W-cmpserTr0m*:∀ *f*. *w-cmpser G m f* ⟶
(∃ *g*. (*W-cmpser G* (*card* (*f* ‘{*i*. *i* ≤ *m*}) − *1*) *g* ∧
g ‘{*i*. *i* ≤ (*card* (*f* ‘{*i*. *i* ≤ *m*}) − *1*)} = *f* ‘{*i*. *i* ≤ *m*}))
⟨*proof*⟩

lemma (**in** *Group*) *ex-W-cmpser*:*w-cmpser G m f* ⟹
∃ *g*. *W-cmpser G* (*card* (*f* ‘ {*i*. *i* ≤ *m*}) − *1*) *g* ∧
g ‘ {*i*. *i* ≤ (*card* (*f* ‘ {*i*. *i* ≤ *m*}) − *1*)} = *f* ‘ {*i*. *i* ≤ *m*}
⟨*proof*⟩

3.15 Existence of reduced chain and composition series

lemma (**in** *Group*) *ex-W-cmpserTr3m1*:⟦*tw-cmpser G* (*m*::*nat*) *f*;
W-cmpser G ((*card* (*f* ‘ {*i*. *i* ≤ *m*})) − *1*) *g*;
g ‘ {*i*. *i* ≤ ((*card* (*f* ‘ {*i*. *i* ≤ *m*})) − *1*)} = *f* ‘{*i*. *i* ≤ *m*}⟧ ⟹
tW-cmpser G ((*card* (*f* ‘ {*i*. *i* ≤ *m*})) − *1*) *g*
⟨*proof*⟩

lemma (**in** *Group*) *ex-W-cmpserTr3m*:*tw-cmpser G m f* ⟹
∃ *g*. *tW-cmpser G* ((*card* (*f* ‘ {*i*. *i* ≤ *m*})) − *1*) *g* ∧
g ‘{ *i*. *i* ≤ (*card* (*f* ‘{*i*. *i* ≤ *m*}) − *1*)} = *f* ‘ {*i*. *i* ≤ *m*}
⟨*proof*⟩

definition
red-ch-cd :: [- , *nat* ⇒ *'a set*, *nat*, *nat* ⇒ *'a set*] ⇒ *bool* **where**
red-ch-cd G f m g ⟷ *tW-cmpser G* (*card* (*f* ‘ {*i*. *i* ≤ *m*}) − *1*) *g* ∧
(*g* ‘{*i* . *i* ≤ (*card* (*f* ‘ {*i*. *i* ≤ *m*}) − *1*)} = *f*‘ {*i*. *i* ≤ *m*})

definition
red-chain :: [- , *nat*, *nat* ⇒ *'a set*] ⇒ (*nat* ⇒ *'a set*) **where**
red-chain G m f = (*SOME g*. *g* ∈ {*h*. *red-ch-cd G f m h*})

lemma (**in** *Group*) *red-chainTr0m1-1*:*tw-cmpser G m f* ⟹
(*SOME g*. *g* ∈ {*h*. *red-ch-cd G f m h*}) ∈ {*h*. *red-ch-cd G f m h*}
⟨*proof*⟩

lemma (**in** *Group*) *red-chain-m*:*tw-cmpser G m f* ⟹
tW-cmpser G (*card* (*f* ‘ {*i*. *i* ≤ *m*}) − *1*) (*red-chain G m f*) ∧
(*red-chain G m f*) ‘{*i*. *i* ≤ (*card* (*f* ‘{*i*. *i* ≤ *m*}) − *1*)} = *f* ‘ {*i*. *i* ≤ *m*}
⟨*proof*⟩

3.16 Chain of groups II

definition
Gchain :: [*nat*, *nat* ⇒ ((*'a set*), *'more*) *Group-scheme*] ⇒ *bool* **where**
Gchain n g ⟷ (∀ *l* ≤ *n*. *Group* (*g l*))

definition
isom-Gchains :: [*nat*, *nat* ⇒ *nat*, *nat* ⇒ ((*'a set*), *'more*) *Group-scheme*,
nat ⇒ ((*'a set*), *'more*) *Group-scheme*] ⇒ *bool* **where**
isom-Gchains n f g h ⟷ (∀ *i* ≤ *n*. (*g i*) ≅ (*h* (*f i*)))

definition
Gch-bridge :: [*nat*, *nat* ⇒ ((*'a set*), *'more*) *Group-scheme*, *nat* ⇒
((*'a set*), *'more*) *Group-scheme*, *nat* ⇒ *nat*] ⇒ *bool* **where**
Gch-bridge n g h f ⟷ (∀ *l* ≤ *n*. *f l* ≤ *n*) ∧ *inj-on f* {*i*. *i* ≤ *n*} ∧

isom-Gchains n f g h

lemma *Gchain-pre*:*Gchain* (*Suc n*) *g* $\Longrightarrow$ *Gchain n g*
⟨*proof*⟩

lemma (**in** *Group*) *isom-unit*:⟦*G* » *H*; *G* » *K*; *H* = {**1**}⟧ $\Longrightarrow$
Gp G H ≅ *Gp G K* ⟶ *K* = {**1**}
⟨*proof*⟩

lemma *isom-gch-unitsTr4*:⟦*Group F*; *Group G*; *Ugp E*; *F* ≅ *G*; *F* ≅ *E*⟧ $\Longrightarrow$
G ≅ *E*
⟨*proof*⟩

lemma *isom-gch-cmp*:⟦*Gchain n g*; *Gchain n h*; *f1* ∈ {*i*. *i* ≤ *n*} → {*i*. *i* ≤ *n*};
f2 ∈ {*i*. *i* ≤ *n*} → {*i*. *i* ≤ *n*}; *isom-Gchains n* (*cmp f2 f1*) *g h*⟧ $\Longrightarrow$
isom-Gchains n f1 g (*cmp h f2*)
⟨*proof*⟩

lemma *isom-gch-transp*:⟦*Gchain n f*; *i* ≤ *n*; *j* ≤ *n*; *i* < *j*⟧ $\Longrightarrow$
isom-Gchains n (*transpos i j*) *f* (*cmp f* (*transpos i j*))
⟨*proof*⟩

lemma *isom-gch-units-transpTr0*:⟦*Ugp E*; *Gchain n g*; *Gchain n h*; *i* ≤ *n*; *j* ≤ *n*;
i < *j*; *isom-Gchains n* (*transpos i j*) *g h*⟧ $\Longrightarrow$
{*i*. *i* ≤ *n* ∧ *g i* ≅ *E*} − {*i*, *j*} ={*i*. *i* ≤ *n* ∧ *h i* ≅ *E*} − {*i*, *j*}
⟨*proof*⟩

lemma *isom-gch-units-transpTr1*:⟦*Ugp E*; *Gchain n g*; *i* ≤ *n*; *j* ≤ *n*; *g j* ≅ *E*;
i ≠ *j*⟧ $\Longrightarrow$
insert j ({*i*. *i* ≤ *n* ∧ *g i* ≅ *E*} − {*i*, *j*}) = {*i*. *i* ≤ *n* ∧ *g i* ≅ *E*} − {*i*}
⟨*proof*⟩

lemma *isom-gch-units-transpTr2*:⟦*Ugp E*; *Gchain n g*; *i* ≤ *n*; *j* ≤ *n*; *i* < *j*;
g i ≅ *E*⟧ $\Longrightarrow$
{*i*. *i* ≤ *n* ∧ *g i* ≅ *E*} = *insert i* ({*i*. *i* ≤ *n* ∧ *g i* ≅ *E*} − {*i*})
⟨*proof*⟩

lemma *isom-gch-units-transpTr3*:⟦*Ugp E*; *Gchain n g*; *i* ≤ *n*⟧
$\Longrightarrow$ *finite* ({*i*. *i* ≤ *n* ∧ *g i* ≅ *E*} − {*i*})
⟨*proof*⟩

lemma *isom-gch-units-transpTr4*:⟦*Ugp E*; *Gchain n g*; *i* ≤ *n*⟧
$\Longrightarrow$ *finite* ({*i*. *i* ≤ *n* ∧ *g i* ≅ *E*} − {*i*, *j*})
⟨*proof*⟩

lemma *isom-gch-units-transpTr5-1*:⟦*Ugp E*; *Gchain n g*; *Gchain n h*; *i* ≤ (*n*::*nat*);
j ≤ *n*; *i* < *j*; *isom-Gchains n* (*transpos i j*) *g h*⟧ $\Longrightarrow$ *g i* ≅ *h j*
⟨*proof*⟩

lemma *isom-gch-units-transpTr5-2*:⟦*Ugp E*; *Gchain n g*; *Gchain n h*; $i \leq n$;
$j \leq n$; $i < j$; *isom-Gchains n* (*transpos i j*) *g h*⟧ ⟹ $g\ j \cong h\ i$
⟨*proof*⟩

lemma *isom-gch-units-transpTr6*:⟦*Gchain n g*; $i \leq n$⟧ ⟹ *Group* (*g i*)
⟨*proof*⟩

lemma *isom-gch-units-transpTr7*:⟦*Ugp E*; $i \leq n$; $j \leq n$; $g\ j \cong h\ i$;
Group (*h i*); *Group* (*g j*); ¬ $g\ j \cong E$⟧ ⟹ ¬ $h\ i \cong E$
⟨*proof*⟩

lemma *isom-gch-units-transpTr8-1*:⟦*Ugp E*; *Gchain n g*; $i \leq n$; $j \leq n$;
$g\ i \cong E$; ¬ $g\ j \cong E$⟧ ⟹
$\{i.\ i \leq n \wedge g\ i \cong E\} = \{i.\ i \leq n \wedge g\ i \cong E\} - \{\ j\ \}$
⟨*proof*⟩

lemma *isom-gch-units-transpTr8-2*:⟦*Ugp E*; *Gchain n g*; $i \leq n$; $j \leq n$;
¬ $g\ i \cong E$; ¬ $g\ j \cong E$⟧ ⟹
$\{i.\ i \leq n \wedge g\ i \cong E\} = \{i.\ i \leq n \wedge g\ i \cong E\} - \{i, j\ \}$
⟨*proof*⟩

lemma *isom-gch-units-transp*:⟦*Ugp E*; *Gchain n g*; *Gchain n h*; $i \leq n$; $j \leq n$;
$i < j$; *isom-Gchains n* (*transpos i j*) *g h*⟧ ⟹
card $\{i.\ i \leq n \wedge g\ i \cong E\}$ = *card* $\{i.\ i \leq n \wedge h\ i \cong E\}$
⟨*proof*⟩

lemma *TR-isom-gch-units*:⟦*Ugp E*; *Gchain n f*; $i \leq n$; $j \leq n$; $i < j$⟧ ⟹
card $\{k.\ k \leq n \wedge f\ k \cong E\}$ = *card* $\{k.\ k \leq n \wedge$
(*cmp f* (*transpos i j*)) $k \cong E\}$
⟨*proof*⟩

lemma *TR-isom-gch-units-1*:⟦*Ugp E*; *Gchain n f*; $i \leq n$; $j \leq n$; $i < j$⟧ ⟹
card $\{k.\ k \leq n \wedge f\ k \cong E\}$ = *card* $\{k.\ k \leq n \wedge f$ (*transpos i j k*) $\cong E\}$
⟨*proof*⟩

lemma *isom-tgch-unitsTr0-1*:⟦*Ugp E*; *Gchain* (*Suc n*) *g*; *g* (*Suc n*) $\cong E$⟧ ⟹
$\{i.\ i \leq$ (*Suc n*) $\wedge\ g\ i \cong E\}$ = *insert* (*Suc n*) $\{i.\ i \leq n \wedge g\ i \cong E\}$
⟨*proof*⟩

lemma *isom-tgch-unitsTr0-2*:*Ugp E* ⟹ *finite* $(\{i.\ i \leq (n::nat) \wedge g\ i \cong E\})$
⟨*proof*⟩

lemma *isom-tgch-unitsTr0-3*:⟦*Ugp E*; *Gchain* (*Suc n*) *g*; ¬ *g* (*Suc n*) $\cong E$⟧
⟹ $\{i.\ i \leq$ (*Suc n*) $\wedge\ g\ i \cong E\} = \{i.\ i \leq n \wedge g\ i \cong E\}$
⟨*proof*⟩

lemma *isom-tgch-unitsTr0*:⟦*Ugp E*;
card $\{i.\ i \leq n \wedge g\ i \cong E\}$ = *card* $\{i.\ i \leq n \wedge h\ i \cong E\}$;
Gchain (*Suc n*) *g* ∧ *Gchain* (*Suc n*) *h* ∧ *Gch-bridge* (*Suc n*) *g h f*;

$f\ (Suc\ n) = Suc\ n]\!] \Longrightarrow$
$card\ \{i.\ i \leq (Suc\ n) \wedge g\ i \cong E\} =$
$card\ \{i.\ i \leq (Suc\ n) \wedge h\ i \cong E\}$
⟨*proof*⟩

lemma *isom-gch-unitsTr1-1*: $[\![Ugp\ E;\ Gchain\ (Suc\ n)\ g \wedge\ Gchain\ (Suc\ n)\ h$
$\wedge\ Gch\text{-}bridge\ (Suc\ n)\ g\ h\ f;\ f\ (Suc\ n) = Suc\ n]\!] \Longrightarrow$
$Gchain\ n\ g \wedge Gchain\ n\ h \wedge Gch\text{-}bridge\ n\ g\ h\ f$
⟨*proof*⟩

lemma *isom-gch-unitsTr1-2*:$[\![Ugp\ E;\ f\ (Suc\ n) \neq Suc\ n;\ inj\text{-}on\ f\ \{i.\ i \leq (Suc\ n)\};$
$\forall\, l \leq (Suc\ n).\ f\ l \leq (Suc\ n)]\!] \Longrightarrow$
$(cmp\ (transpos\ (f\ (Suc\ n))\ (Suc\ n))\ f)\ (Suc\ n) = Suc\ n$
⟨*proof*⟩

lemma *isom-gch-unitsTr1-3*:$[\![Ugp\ E;\ f\ (Suc\ n) \neq Suc\ n;$
$\forall\, l \leq (Suc\ n).\ f\ l \leq (Suc\ n);\ inj\text{-}on\ f\ \{i.\ i \leq (Suc\ n)\}]\!] \Longrightarrow$
$inj\text{-}on\ (cmp\ (transpos\ (f\ (Suc\ n))\ (Suc\ n))\ f)\ \{i.\ i \leq (Suc\ n)\}$
⟨*proof*⟩

lemma *isom-gch-unitsTr1-4*:$[\![Ugp\ E;\ f\ (Suc\ n) \neq Suc\ n;\ inj\text{-}on\ f\ \{i.\ i \leq (Suc\ n)\};$
$\forall\, l \leq (Suc\ n).\ f\ l \leq (Suc\ n)]\!] \Longrightarrow$
$inj\text{-}on\ (cmp\ (transpos\ (f\ (Suc\ n))\ (Suc\ n))\ f)\ \{i.\ i \leq n\}$
⟨*proof*⟩

lemma *isom-gch-unitsTr1-5*:$[\![Ugp\ E;\ Gchain\ (Suc\ n)\ g \wedge Gchain\ (Suc\ n)\ h \wedge$
$Gch\text{-}bridge\ (Suc\ n)\ g\ h\ f;\ f\ (Suc\ n) \neq Suc\ n\ ]\!] \Longrightarrow$
$Gchain\ n\ g \wedge Gchain\ n\ (cmp\ h\ (transpos\ (f\ (Suc\ n))\ (Suc\ n))) \wedge$
$Gch\text{-}bridge\ n\ g\ (cmp\ h\ (transpos\ (f\ (Suc\ n))\ (Suc\ n)))$
$(cmp\ (transpos\ (f\ (Suc\ n))\ (Suc\ n))\ f)$
⟨*proof*⟩

lemma *isom-gch-unitsTr1-6*:$[\![Ugp\ E;\ f\ (Suc\ n) \neq Suc\ n;\ Gchain\ (Suc\ n)\ g \wedge$
$Gchain\ (Suc\ n)\ h \wedge Gch\text{-}bridge\ (Suc\ n)\ g\ h\ f]\!] \Longrightarrow Gchain\ (Suc\ n)\ g \wedge$
$Gchain\ (Suc\ n)\ (cmp\ h\ (transpos\ (f\ (Suc\ n))\ (Suc\ n))) \wedge$
$Gch\text{-}bridge\ (Suc\ n)\ g\ (cmp\ h\ (transpos\ (f\ (Suc\ n))\ (Suc\ n)))$
$(cmp\ (transpos\ (f\ (Suc\ n))\ (Suc\ n))\ f)$
⟨*proof*⟩

lemma *isom-gch-unitsTr1-7-0*:$[\![Gchain\ (Suc\ n)\ h;\ k \neq Suc\ n;\ k \leq (Suc\ n)]\!]$
$\Longrightarrow Gchain\ (Suc\ n)\ (cmp\ h\ (transpos\ k\ (Suc\ n)))$
⟨*proof*⟩

lemma *isom-gch-unitsTr1-7-1*:$[\![Ugp\ E;\ Gchain\ (Suc\ n)\ h;\ k \neq Suc\ n;\ k \leq (Suc\ n)]\!]$
$\Longrightarrow \{i.\ i \leq (Suc\ n) \wedge cmp\ h\ (transpos\ k\ (Suc\ n))\ i \cong E\} - \{k\ ,\ Suc\ n\} =$
$\{i.\ i \leq (Suc\ n) \wedge h\ i \cong E\} - \{k\ ,\ Suc\ n\}$
⟨*proof*⟩

lemma *isom-gch-unitsTr1-7-2*:⟦*Ugp E*; *Gchain* (*Suc n*) *h*; *k* ≠ *Suc n*;
k ≤ (*Suc n*); *h* (*Suc n*) ≅ *E*⟧ ⟹
cmp h (*transpos k* (*Suc n*)) *k* ≅ *E*
⟨*proof*⟩

lemma *isom-gch-unitsTr1-7-3*:⟦*Ugp E*; *Gchain* (*Suc n*) *h*; *k* ≠ *Suc n*;
k ≤ (*Suc n*); *h k* ≅ *E*⟧ ⟹ *cmp h* (*transpos k* (*Suc n*)) (*Suc n*) ≅ *E*
⟨*proof*⟩

lemma *isom-gch-unitsTr1-7-4*:⟦*Ugp E*; *Gchain* (*Suc n*) *h*; *k* ≠ *Suc n*;
k ≤ (*Suc n*); ¬ *h* (*Suc n*) ≅ *E*⟧ ⟹
¬ *cmp h* (*transpos k* (*Suc n*)) *k* ≅ *E*
⟨*proof*⟩

lemma *isom-gch-unitsTr1-7-5*:⟦*Ugp E*; *Gchain* (*Suc n*) *h*; *k* ≠ *Suc n*;
k ≤ (*Suc n*); ¬ *h k* ≅ *E*⟧ ⟹
¬ *cmp h* (*transpos k* (*Suc n*)) (*Suc n*) ≅ *E*
⟨*proof*⟩

lemma *isom-gch-unitsTr1-7-6*:⟦*Ugp E*; *Gchain* (*Suc n*) *h*; *k* ≠ *Suc n*;
k ≤ (*Suc n*); *h* (*Suc n*) ≅ *E*; *h k* ≅ *E*⟧ ⟹
{*i*. *i* ≤ (*Suc n*) ∧ *h i* ≅ *E*} =
insert k (*insert* (*Suc n*) ({*i*. *i* ≤ (*Suc n*) ∧ *h i* ≅ *E*} − {*k*, *Suc n*}))
⟨*proof*⟩

lemma *isom-gch-unitsTr1-7-7*:⟦*Ugp E*; *Gchain* (*Suc n*) *h*; *k* ≠ *Suc n*;
k ≤ (*Suc n*); *h* (*Suc n*) ≅ *E*; ¬ *h k* ≅ *E*⟧ ⟹
{*i*. *i* ≤ (*Suc n*) ∧ *h i* ≅ *E*} =
insert (*Suc n*) ({*i*. *i* ≤ (*Suc n*) ∧ *h i* ≅ *E*} − {*k*, *Suc n*})
⟨*proof*⟩

lemma *isom-gch-unitsTr1-7-8*:⟦*Ugp E*; *Gchain* (*Suc n*) *h*; *k* ≠ *Suc n*;
k ≤ (*Suc n*); ¬ *h* (*Suc n*) ≅ *E*; *h k* ≅ *E*⟧ ⟹
{*i*. *i* ≤ (*Suc n*) ∧ *h i* ≅ *E*} =
insert k ({*i*. *i* ≤ (*Suc n*) ∧ *h i* ≅ *E*} − {*k*, *Suc n*})
⟨*proof*⟩

lemma *isom-gch-unitsTr1-7-9*:⟦*Ugp E*; *Gchain* (*Suc n*) *h*; *k* ≠ *Suc n*;
k ≤ (*Suc n*); ¬ *h* (*Suc n*) ≅ *E*; ¬ *h k* ≅ *E*⟧ ⟹
{*i*. *i* ≤ (*Suc n*) ∧ *h i* ≅ *E*} =
{*i*. *i* ≤ (*Suc n*) ∧ *h i* ≅ *E*} − {*k*, *Suc n*}
⟨*proof*⟩

lemma *isom-gch-unitsTr1-7*:⟦*Ugp E*; *Gchain* (*Suc n*) *h*; *k* ≠ *Suc n*;
k ≤ (*Suc n*)⟧ ⟹ *card* {*i*. *i* ≤ (*Suc n*) ∧
cmp h (*transpos k* (*Suc n*)) *i* ≅ *E*} = *card* {*i*. *i* ≤ (*Suc n*) ∧ *h i* ≅ *E*}
⟨*proof*⟩

lemma *isom-gch-unitsTr1*:*Ugp E* ⟹ ∀ *g*. ∀ *h*. ∀ *f*. *Gchain n g* ∧

$Gchain\ n\ h \wedge Gch\text{-}bridge\ n\ g\ h\ f \longrightarrow\ card\ \{i.\ i \leq n \wedge g\ i \cong E\} =$
$card\ \{i.\ i \leq n \wedge h\ i \cong E\}$
⟨*proof*⟩

lemma *isom-gch-units*:⟦*Ugp E*; *Gchain n g*; *Gchain n h*; *Gch-bridge n g h f*⟧ ⟹
$card\ \{i.\ i \leq n \wedge g\ i \cong E\} = card\ \{i.\ i \leq n \wedge h\ i \cong E\}$
⟨*proof*⟩

lemma *isom-gch-units-1*:⟦*Ugp E*; *Gchain n g*; *Gchain n h*; ∃*f*. *Gch-bridge n g h f*⟧
⟹ $card\ \{i.\ i \leq n \wedge g\ i \cong E\} = card\ \{i.\ i \leq n \wedge h\ i \cong E\}$
⟨*proof*⟩

3.17 Jordan Hoelder theorem

3.17.1 *Rfn-tools*. Tools to treat refinement of a cmpser, rtos.

lemma *rfn-tool1*:⟦ $0 < (r::nat)$; $(k::nat) = i * r + j$; $j < r$ ⟧
$\Longrightarrow (k\ div\ r) = i$
⟨*proof*⟩

lemma *pos-mult-pos*:⟦ $0 < (r::nat)$; $0 < s$⟧ ⟹ $0 < r * s$
⟨*proof*⟩

lemma *rfn-tool1-1*:⟦ $0 < (r::nat)$; $j < r$ ⟧
⟹ $(i * r + j)\ div\ r = i$
⟨*proof*⟩

lemma *rfn-tool2*:$(a::nat) < s \Longrightarrow a \leq s - Suc\ 0$
⟨*proof*⟩

lemma *rfn-tool3*:$(0::nat) \leq m \Longrightarrow (m + n) - n = m$
⟨*proof*⟩

lemma *rfn-tool11*:⟦$0 < b$; $(a::nat) \leq b - Suc\ 0$⟧ ⟹ $a < b$
⟨*proof*⟩

lemma *rfn-tool12*:⟦$0 < (s::nat)$; $(i::nat)\ mod\ s = s - 1$ ⟧ ⟹
$Suc\ (i\ div\ s) = (Suc\ i)\ div\ s$
⟨*proof*⟩

lemma *rfn-tool12-1*:⟦$0 < (s::nat)$; $(l::nat)\ mod\ s < s - 1$ ⟧ ⟹
$Suc\ (l\ mod\ s) = (Suc\ l)\ mod\ s$
⟨*proof*⟩

lemma *rfn-tool12-2*:⟦$0 < (s::nat)$; $(i::nat)\ mod\ s = s - Suc\ 0$⟧ ⟹
$(Suc\ i)\ mod\ s = 0$
⟨*proof*⟩

lemma *rfn-tool13*:⟦ (0::*nat*) < *r*; *a* = *b* ⟧ ⟹ *a mod r* = *b mod r*
⟨*proof*⟩

lemma *rfn-tool13-1*:⟦ (0::*nat*) < *r*; *a* = *b* ⟧ ⟹ *a div r* = *b div r*
⟨*proof*⟩

lemma *div-Tr1*:⟦ (0::*nat*) < *r*; 0 < *s*; *l* ≤ *s* ∗ *r*⟧ ⟹ *l div s* ≤ *r*
⟨*proof*⟩

lemma *div-Tr2*:⟦(0::*nat*) < *r*; 0 < *s*; *l* < *s* ∗ *r*⟧ ⟹ *l div s* ≤ *r* − *Suc 0*
⟨*proof*⟩

lemma *div-Tr3*:⟦(0::*nat*) < *r*; 0 < *s*; *l* < *s* ∗ *r*⟧ ⟹ *Suc* (*l div s*) ≤ *r*
⟨*proof*⟩

lemma *div-Tr3-1*:⟦(0::*nat*) < *r*; 0 < *s*; *l mod s* = *s* − *1*⟧ ⟹ *Suc l div s* = *Suc*
(*l div s*)
⟨*proof*⟩

lemma *div-Tr3-2*:⟦(0::*nat*) < *r*; 0 < *s*; *l mod s* < *s* − *1*⟧ ⟹
l div s = *Suc l div s*
⟨*proof*⟩

lemma *mod-div-injTr*:⟦(0::*nat*) < *r*; *x mod r* = *y mod r*; *x div r* = *y div r*⟧
⟹ *x* = *y*
⟨*proof*⟩

definition
rtos :: [*nat*, *nat*] ⇒ (*nat* ⇒ *nat*) **where**
rtos r s i = (*if i* < *r* ∗ *s then* (*i mod s*) ∗ *r* + *i div s else r* ∗ *s*)

lemma *rtos-hom0*:⟦(0::*nat*) < *r*; (0::*nat*) < *s*; *i* ≤ (*r* ∗ *s* − *Suc 0*)⟧ ⟹
i div s < *r*
⟨*proof*⟩

lemma *rtos-hom1*:⟦(0::*nat*) < *r*; 0 < *s*; *l* ≤ (*r* ∗ *s* − *Suc 0*)⟧ ⟹
(*rtos r s*) *l* ≤ (*s* ∗ *r* − *Suc 0*)
⟨*proof*⟩

lemma *rtos-hom2*:⟦(0::*nat*) < *r*; (0::*nat*) < *s*; *l* ≤ (*r* ∗ *s* − *Suc 0*)⟧ ⟹
rtos r s l ≤ (*r* ∗ *s* − *Suc 0*)
⟨*proof*⟩

lemma *rtos-hom3*:⟦(0::*nat*) < *r*; 0 < *s*; *i* ≤ (*r* ∗ *s* − *Suc 0*) ⟧ ⟹
(*rtos r s i div r*) = *i mod s*
⟨*proof*⟩

lemma *rtos-hom3-1*:⟦(0::*nat*) < *r*; (0::*nat*) < *s*; *i* ≤ (*r* ∗ *s* − *Suc 0*) ⟧ ⟹
(*rtos r s i mod r*) = *i div s*

⟨*proof*⟩

lemma *rtos-hom5*:⟦(*0*::*nat*) < *r*; (*0*::*nat*) < *s*; *i* ≤ (*r* ∗*s* − *Suc 0*);
i div s = *r* − *Suc 0* ⟧ ⟹ *Suc* (*rtos r s i*) *div r* = *Suc* (*i mod s*)
⟨*proof*⟩

lemma *rtos-hom7*:⟦(*0*::*nat*) < *r*; (*0*::*nat*) < *s*; *i* ≤ (*r* ∗ *s* − *Suc 0*);
i div s = *r* − *Suc 0* ⟧ ⟹ *Suc* (*rtos r s i*) *mod r* = *0*
⟨*proof*⟩

lemma *rtos-inj*:⟦ (*0*::*nat*) < *r*; (*0*::*nat*) < *s* ⟧ ⟹
inj-on (*rtos r s*) {*i*. *i* ≤ (*r* ∗ *s* − *Suc 0*)}
⟨*proof*⟩

lemma *rtos-rs-Tr1*:⟦(*0*::*nat*) < *r*; *0* < *s* ⟧ ⟹ *rtos r s* (*r* ∗ *s*) = *r* ∗ *s*
⟨*proof*⟩

lemma *rtos-rs-Tr2*:⟦(*0*::*nat*) < *r*; *0* < *s* ⟧ ⟹
∀ *l* ≤ (*r* ∗ *s*). *rtos r s l* ≤ (*r* ∗ *s*)
⟨*proof*⟩

lemma *rtos-rs-Tr3*:⟦(*0*::*nat*) < *r*; *0* < *s* ⟧ ⟹
inj-on (*rtos r s*) {*i*. *i* ≤ (*r* ∗ *s*)}
⟨*proof*⟩

lemma *Qw-cmpser*:⟦*Group G*; *w-cmpser G* (*Suc n*) *f* ⟧ ⟹
Gchain n (*Qw-cmpser G f*)
⟨*proof*⟩

definition
wcsr-rfns :: [- , *nat*, *nat* ⇒ *'a set*, *nat*] ⇒ (*nat* ⇒ *'a set*) *set* **where**
wcsr-rfns G r f s = {*h*. *tw-cmpser G* (*s* ∗ *r*) *h* ∧
(∀ *i* ≤ *r*. *h* (*i* ∗ *s*) = *f i*)}

definition
trivial-rfn :: [- , *nat*, *nat* ⇒ *'a set*, *nat*] ⇒ (*nat* ⇒ *'a set*) **where**
trivial-rfn G r f s k == *if k* < (*s* ∗ *r*) *then f* (*k div s*) *else f r*

lemma (**in** *Group*) *rfn-tool8*:⟦*compseries G r f*; *0* < *r*⟧ ⟹ *d-gchain G r f*
⟨*proof*⟩

lemma (**in** *Group*) *rfn-tool16*:⟦*0* < *r*; *0* < *s*; *i* ≤ (*s* ∗ *r* − *Suc 0*);
G » *f* (*i div s*); (*Gp G* (*f* (*i div s*))) ▷ *f* (*Suc* (*i div s*));
(*Gp G* (*f* (*i div s*))) » (*f* (*i div s*) ∩ *g* (*s* − *Suc 0*))⟧ ⟹
(*Gp G* ((*f* (*Suc* (*i div s*)) ⋄$_G$ (*f* (*i div s*) ∩ *g* (*s* − *Suc 0*))))) ▷
(*f* (*Suc* (*i div s*)))
⟨*proof*⟩

Show existence of the trivial refinement. This is not necessary to prove JHS

lemma *rfn-tool30*:$[\![0 < r;\ 0 < s;\ l\ div\ s * s + s < s * r]\!]$
$\Longrightarrow Suc\ (l\ div\ s) < r$
⟨*proof*⟩

lemma (**in** *Group*) *simple-grouptr0*:$[\![G \gg H;\ G \triangleright K;\ K \subseteq H;\ simple\text{-}Group\ (G\ /\ K)]\!]$
$\Longrightarrow H = carrier\ G \vee H = K$
⟨*proof*⟩

lemma (**in** *Group*) *compser-nsg*:$[\![0 < n;\ compseries\ G\ n\ f;\ i \leq (n - 1)]\!]$
$\Longrightarrow\ Gp\ G\ (f\ i) \triangleright (f\ (Suc\ i))$
⟨*proof*⟩

lemma (**in** *Group*) *compseriesTr5*:$[\![0 < n;\ compseries\ G\ n\ f;\ i \leq (n - Suc\ 0)]\!]$
$\Longrightarrow (f\ (Suc\ i)) \subseteq\ (f\ i)$
⟨*proof*⟩

lemma (**in** *Group*) *refine-cmpserTr0*:$[\![0 < n;\ compseries\ G\ n\ f;\ i \leq (n - 1);$
$G \gg H;\ f\ (Suc\ i) \subseteq H \wedge H \subseteq f\ i]\!] \Longrightarrow H = f\ (Suc\ i) \vee H = f\ i$
⟨*proof*⟩

lemma *div-Tr4*:$[\![\ (0::nat) < r;\ 0 < s;\ j < s * r\ ]\!] \Longrightarrow j\ div\ s * s + s \leq r * s$
⟨*proof*⟩

lemma (**in** *Group*) *compseries-is-tW-cmpser*:$[\![0 < r;\ compseries\ G\ r\ f]\!] \Longrightarrow$
$tW\text{-}cmpser\ G\ r\ f$
⟨*proof*⟩

lemma (**in** *Group*) *compseries-is-td-gchain*:$[\![0 < r;\ compseries\ G\ r\ f]\!] \Longrightarrow$
$td\text{-}gchain\ G\ r\ f$
⟨*proof*⟩

lemma (**in** *Group*) *compseries-is-D-gchain*:$[\![0 < r;\ compseries\ G\ r\ f]\!] \Longrightarrow$
$D\text{-}gchain\ G\ r\ f$
⟨*proof*⟩

lemma *divTr5*:$[\![0 < r;\ 0 < s;\ l < (r * s)]\!]\ \Longrightarrow$
$l\ div\ s * s \leq l \wedge l\ \leq (Suc\ (l\ div\ s)) * s$
⟨*proof*⟩

lemma (**in** *Group*) *rfn-compseries-iMTr1*:$[\![0 < r;\ 0 < s;\ compseries\ G\ r\ f;$
$h \in wcsr\text{-}rfns\ G\ r\ f\ s]\!] \Longrightarrow\ f\ `\ \{i.\ i \leq r\} \subseteq\ h\ `\ \{i.\ i \leq (s * r)\}$
⟨*proof*⟩

lemma *rfn-compseries-iMTr2*:$[\![0 < r;\ 0 < s;\ xa < s * r\ ]\!] \Longrightarrow$
$xa\ div\ s * s \leq r * s \wedge Suc\ (xa\ div\ s) * s \leq r * s$
⟨*proof*⟩

lemma (**in** *Group*) *rfn-compseries-iMTr3*:⟦*0* < *r*; *0* < *s*; *compseries G r f*;
j ≤ *r*; ∀ *i* ≤ *r*. *h* (*i* ∗ *s*) = *f i*⟧ ⟹ *h* (*j* ∗ *s*) = *f j*
⟨*proof*⟩

lemma (**in** *Group*) *rfn-compseries-iM*:⟦*0* < *r*; *0* < *s*; *compseries G r f*;
h ∈ *wcsr-rfns G r f s*⟧ ⟹ *card* (*h* ‘{*i*. *i* ≤ (*s* ∗ *r*)}) = *r* + *1*
⟨*proof*⟩

definition
cmp-rfn :: [- , *nat*, *nat* ⇒ *'a set*, *nat*, *nat* ⇒ *'a set*] ⇒ (*nat* ⇒ *'a set*) **where**
cmp-rfn G r f s g = (λ*i*. (*if i* < *s* ∗ *r then*
f (*Suc* (*i div s*)) ⋄$_G$ (*f* (*i div s*) ∩ *g* (*i mod s*)) *else* {**1**$_G$}))

lemma (**in** *Group*) *cmp-rfn0*:⟦*0* < *r*; *0* < *s*; *compseries G r f*; *compseries G s g*;
i ≤ (*r* − *1*); *j* ≤ (*s* − *1*)⟧ ⟹ *G* » *f* (*Suc i*) ⋄$_G$ ((*f i*) ∩ (*g j*))
⟨*proof*⟩

lemma (**in** *Group*) *cmp-rfn1*:⟦*0* < *r*; *0* < *s*; *compseries G r f*; *compseries G s g*⟧
⟹ *f* (*Suc 0*) ⋄$_G$ ((*f 0*) ∩ (*g 0*)) = *carrier G*
⟨*proof*⟩

lemma (**in** *Group*) *cmp-rfn2*:⟦*0* < *r*; *0* < *s*; *compseries G r f*; *compseries G s g*;
l ≤ (*s* ∗ *r*)⟧ ⟹ *G* » *cmp-rfn G r f s g l*
⟨*proof*⟩

lemma (**in** *Group*) *cmp-rfn3*:⟦*0* < *r*; *0* < *s*; *compseries G r f*; *compseries G s g*⟧
⟹ *cmp-rfn G r f s g 0* = *carrier G* ∧ *cmp-rfn G r f s g* (*s* ∗ *r*) = {**1**}
⟨*proof*⟩

lemma *rfn-tool20*:⟦(*0*::*nat*) < *m*; *a* = *b* ∗ *m* + *c*; *c* < *m* ⟧ ⟹ *a mod m* = *c*
⟨*proof*⟩

lemma *Suci-mod-s-2*:⟦*0* < *r*; *0* < *s*; *i* ≤ *r* ∗ *s* − *Suc 0*; *i mod s* < *s* − *Suc 0*⟧
⟹ (*Suc i*) *mod s* = *Suc* (*i mod s*)
⟨*proof*⟩

lemma (**in** *Group*) *inter-sgsTr1*:⟦*0* < *r*; *0* < *s*; *compseries G r f*; *compseries G s g*; *i* < *r* ∗ *s*⟧ ⟹ *G* » *f* (*i div s*) ∩ *g* (*s* − *Suc 0*)
⟨*proof*⟩

lemma (**in** *Group*) *JHS-Tr0-2*:⟦*0* < *r*; *0* < *s*; *compseries G r f*; *compseries G s g*⟧
⟹ ∀ *i* ≤ (*s* ∗ *r* − *Suc 0*). *Gp G* (*cmp-rfn G r f s g i*) ▷
cmp-rfn G r f s g (*Suc i*)

⟨*proof*⟩

lemma (**in** *Group*) *cmp-rfn4*:⟦*0* < *r*; *0* < *s*; *compseries G r f*;
compseries G s g; *l* ≤ (*s* ∗ *r* − *Suc 0*)⟧ ⟹
cmp-rfn G r f s g (*Suc l*) ⊆ *cmp-rfn G r f s g l*
⟨*proof*⟩

lemma (**in** *Group*) *cmp-rfn5*:⟦*0* < *r*; *0* < *s*; *compseries G r f*; *compseries G s g*⟧
⟹ ∀ *i* ≤ *r*. *cmp-rfn G r f s g* (*i* ∗ *s*) = *f i*
⟨*proof*⟩

lemma (**in** *Group*) *JHS-Tr0*:⟦(*0*::*nat*) < *r*; *0* < *s*; *compseries G r f*;
compseries G s g⟧ ⟹ *cmp-rfn G r f s g* ∈ *wcsr-rfns G r f s*
⟨*proof*⟩

lemma *rfn-tool17*:(*a*::*nat*) = *b* ⟹ *a* − *c* = *b* − *c*
⟨*proof*⟩

lemma *isom4b*:⟦*Group G*; *G* ▷ *N*; *G* » *H*⟧ ⟹
(*Gp G* (*N* ⋄$_G$ *H*) / *N*) ≅ (*Gp G H* / (*H* ∩ *N*))
⟨*proof*⟩

lemma *Suc-rtos-div-r-1*:⟦ *0* < *r*; *0* < *s*; *i* ≤ *r* ∗ *s* − *Suc 0*;
Suc (*rtos r s i*) < *r* ∗ *s*; *i mod s* = *s* − *Suc 0*;
i div s < *r* − *Suc 0*⟧ ⟹ *Suc* (*rtos r s i*) *div r* = *i mod s*
⟨*proof*⟩

lemma *Suc-rtos-mod-r-1*:⟦*0* < *r*; *0* < *s*; *i* ≤ *r* ∗ *s* − *Suc 0*; *Suc* (*rtos r s i*) < *r*
∗ *s*; *i mod s* = *s* − *Suc 0*; *i div s* < *r* − *Suc 0*⟧
⟹ *Suc* (*rtos r s i*) *mod r* = *Suc* (*i div s*)
⟨*proof*⟩

lemma *i-div-s-less*:⟦*0* < *r*; *0* < *s*; *i* ≤ *r* ∗ *s* − *Suc 0*; *Suc* (*rtos r s i*) < *r* ∗ *s*;
i mod s = *s* − *Suc 0*; *Suc i* < *s* ∗ *r* ⟧ ⟹ *i div s* < *r* − *Suc 0*
⟨*proof*⟩

lemma *rtos-mod-r-1*:⟦ *0* < *r*; *0* < *s*; *i* ≤ *r* ∗ *s* − *Suc 0*; *rtos r s i* < *r* ∗ *s*;
i mod s = *s* − *Suc 0* ⟧ ⟹ *rtos r s i mod r* = *i div s*
⟨*proof*⟩

lemma *Suc-i-mod-s-0-1*:⟦*0* < *r*; *0* < *s*; *i* ≤ *r* ∗ *s* − *Suc 0*; *i mod s* = *s* − *Suc 0*⟧
⟹ *Suc i mod s* = *0*
⟨*proof*⟩

lemma *Suci-div-s-2*:⟦*0* < *r*; *0* < *s*; *i* ≤ *r* ∗ *s* − *Suc 0*; *i mod s* < *s* − *Suc 0*⟧
⟹ *Suc i div s* = *i div s*
⟨*proof*⟩

lemma *rtos-i-mod-r-2*:⟦*0* < *r*; *0* < *s*; *i* ≤ *r* ∗ *s* − *Suc 0*⟧ ⟹ *rtos r s i mod r* =

i *div* s
⟨*proof*⟩

lemma *Suc-rtos-i-mod-r-2*:⟦$0 < r$; $0 < s$; $i \leq r * s - Suc\ 0$;
$i\ div\ s = r - Suc\ 0$⟧ ⟹ $Suc\ (rtos\ r\ s\ i)\ mod\ r = 0$
⟨*proof*⟩

lemma *Suc-rtos-i-mod-r-3*:⟦$0 < r$; $0 < s$; $i \leq r * s - Suc\ 0$;
$i\ div\ s < r - Suc\ 0$⟧ ⟹ $Suc\ (rtos\ r\ s\ i)\ mod\ r = Suc\ (i\ div\ s)$
⟨*proof*⟩

lemma *Suc-rtos-div-r-3*:⟦$0 < r$; $0 < s$; $i\ mod\ s < s - Suc\ 0$; $i \leq r * s - Suc\ 0$;
$Suc\ (rtos\ r\ s\ i) < r * s$; $i\ div\ s < r - Suc\ 0$⟧ ⟹
$Suc\ (rtos\ r\ s\ i)\ div\ r = i\ mod\ s$
⟨*proof*⟩

lemma *r-s-div-s*:⟦$0 < r$; $0 < s$⟧ ⟹ $(r * s - Suc\ 0)\ div\ s = r - Suc\ 0$
⟨*proof*⟩

lemma *r-s-mod-s*:⟦$0 < r$; $0 < s$⟧ ⟹ $(r * s - Suc\ 0)\ mod\ s = s - Suc\ 0$
⟨*proof*⟩

lemma *rtos-r-s*:⟦$0 < r$; $0 < s$⟧ ⟹ $rtos\ r\ s\ (r * s - Suc\ 0) = r * s - Suc\ 0$
⟨*proof*⟩

lemma *rtos-rs-1*:⟦ $0 < r$; $0 < s$; $rtos\ r\ s\ i < r * s$;
$\neg\ Suc\ (rtos\ r\ s\ i) < r * s$⟧ ⟹ $rtos\ r\ s\ i = r * s - Suc\ 0$
⟨*proof*⟩

lemma *rtos-rs-i-rs*:⟦ $0 < r$; $0 < s$; $i \leq r * s - Suc\ 0$;
$rtos\ r\ s\ i = r * s - Suc\ 0$⟧ ⟹ $i = r * s - Suc\ 0$
⟨*proof*⟩

lemma *JHS-Tr1-1*:⟦$Group\ G$; $0 < r$; $0 < s$; $compseries\ G\ r\ f$; $compseries\ G\ s\ g$⟧
⟹ $f\ (Suc\ ((r * s - Suc\ 0)\ div\ s))\ \diamond_G\ (f\ ((r * s - Suc\ 0)\ div\ s) \cap g\ ((r * s - Suc\ 0)\ mod\ s)) = f\ (r - Suc\ 0) \cap g\ (s - Suc\ 0)$
⟨*proof*⟩

lemma *JHS-Tr1-2*:⟦$Group\ G$; $0 < r$; $0 < s$; $compseries\ G\ r\ f$; $compseries\ G\ s\ g$;
$k < r - Suc\ 0$⟧ ⟹ $((Gp\ G\ (f\ (Suc\ k)\ \diamond_G\ (f\ k \cap g\ (s - Suc\ 0)))) /$
$(f\ (Suc\ (Suc\ k))\ \diamond_G\ (f\ (Suc\ k) \cap g\ 0))) \cong$
$((Gp\ G\ (g\ s\ \diamond_G\ (g\ (s - Suc\ 0) \cap f\ k))) /$
$(g\ s\ \diamond_G\ (g\ (s - Suc\ 0) \cap f\ (Suc\ k))))$
⟨*proof*⟩

lemma *JHS-Tr1-3*:⟦$Group\ G$; $0 < r$; $0 < s$; $compseries\ G\ r\ f$; $compseries\ G\ s\ g$;
$i \leq s * r - Suc\ 0$; $Suc\ (rtos\ r\ s\ i) < s * r$; $Suc\ i < s * r$;
$i\ mod\ s < s - Suc\ 0$; $Suc\ i\ div\ s \leq r - Suc\ 0$; $i\ div\ s = r - Suc\ 0$⟧
⟹ $Group\ (Gp\ G\ (f\ r\ \diamond_G\ (f\ (r - Suc\ 0) \cap g\ (i\ mod\ s))) /$

$(f\ r \diamond_G (f\ (r - Suc\ 0) \cap g\ (Suc\ (i\ mod\ s)))))$
⟨*proof*⟩

lemma *JHS-Tr1-4*:⟦*Group G*; *0 < r*; *0 < s*; *compseries G r f*; *compseries G s g*;
*i ≤ s * r − Suc 0*; *Suc (rtos r s i) < s * r*; *Suc i < s * r*;
i mod s < s − Suc 0; *Suc i div s ≤ r − Suc 0*; *i div s = r − Suc 0*⟧ ⟹
Group (Gp G (g (Suc (i mod s)) ◇G (g (i mod s) ∩ f (r − Suc 0))) /
(g (Suc (Suc (i mod s))) ◇G (g (Suc (i mod s)) ∩ f 0)))
⟨*proof*⟩

lemma *JHS-Tr1-5*:⟦*Group G*; *0 < r*; *0 < s*; *compseries G r f*; *compseries G s g*;
*i ≤ s * r − Suc 0*; *Suc (rtos r s i) < s * r*; *Suc i < s * r*;
i mod s < s − Suc 0; *i div s < r − Suc 0*⟧
⟹ *(Gp G (f (Suc (i div s)) ◇G (f (i div s) ∩ g (i mod s))) /*
(f (Suc (i div s)) ◇G (f (i div s) ∩ g (Suc (i mod s))))) ≅
(Gp G (g (Suc (i mod s)) ◇G (g (i mod s) ∩ f (i div s))) /
(g (Suc (Suc (rtos r s i) div r)) ◇G
(g (Suc (rtos r s i) div r) ∩ f (Suc (rtos r s i) mod r))))
⟨*proof*⟩

lemma *JHS-Tr1-6*: ⟦*Group G*; *0 < r*; *0 < s*; *compseries G r f*; *compseries G s g*;
*i ≤ r * s − Suc 0*; *Suc (rtos r s i) < r * s*⟧ ⟹
((Gp G (cmp-rfn G r f s g i)) / (cmp-rfn G r f s g (Suc i))) ≅
((Gp G (g (Suc (rtos r s i div r)) ◇G
(g (rtos r s i div r) ∩ f (rtos r s i mod r)))) /
(g (Suc (Suc (rtos r s i) div r)) ◇G
(g (Suc (rtos r s i) div r) ∩ f (Suc (rtos r s i) mod r))))
⟨*proof*⟩

lemma *JHS-Tr1*:⟦ *Group G*; *0 < r*; *0 < s*; *compseries G r f*; *compseries G s g*⟧
⟹ *isom-Gchains (r * s − 1) (rtos r s) (Qw-cmpser G (cmp-rfn G r f s g))*
(Qw-cmpser G (cmp-rfn G s g r f))
⟨*proof*⟩

lemma *abc-SucTr0*:⟦*(0::nat) < a*; *c ≤ b*; *a − Suc 0 = b − c*⟧ ⟹ *a = (Suc b) − c*
⟨*proof*⟩

lemma *length-wcmpser0-0*:⟦*Group G*; *Ugp E*; *w-cmpser G (Suc 0) f*⟧ ⟹
f ‘ {i. i ≤ (Suc 0)} = {f 0, f (Suc 0)}
⟨*proof*⟩

lemma *length-wcmpser0-1*:⟦*Group G*; *Ugp E*; *w-cmpser G (Suc n) f*; *i∈{i. i ≤ n}*;
(Qw-cmpser G f) i ≅ E⟧ ⟹ *f i = f (Suc i)*
⟨*proof*⟩

lemma *length-wcmpser0-2*:⟦*Group G*; *Ugp E*; *w-cmpser G (Suc n) f*; *i ≤ n*;
¬ (Qw-cmpser G f) i ≅ E⟧ ⟹ *f i ≠ f (Suc i)*

⟨*proof*⟩

lemma *length-wcmpser0-3*:⟦*Group G*; *Ugp E*; *w-cmpser G* (*Suc* (*Suc n*)) *f*;
f (*Suc n*) ≠ *f* (*Suc* (*Suc n*))⟧ ⟹ *f* (*Suc* (*Suc n*)) ∉ *f* ‘ {*i*. *i* ≤ (*Suc n*)}
⟨*proof*⟩

lemma *length-wcmpser0-4*:⟦*Group G*; *Ugp E*; *w-cmpser G* (*Suc 0*) *f*⟧ ⟹
card (*f* ‘ {*i*. *i* ≤ *Suc 0*}) − *1* = *Suc 0* − *card* {*i*. *i* = *0* ∧
Qw-cmpser G f i ≅ *E*}

⟨*proof*⟩

lemma *length-wcmpser0-5*: ⟦*Group G*; *Ugp E*; *w-cmpser G* (*Suc* (*Suc n*)) *f*;
w-cmpser G (*Suc n*) *f*;
card (*f* ‘ {*i*. *i* ≤ (*Suc n*)}) − *1* = *Suc n* −
card {*i*. *i* ≤ *n* ∧ *Qw-cmpser G f i* ≅ *E*};
Qw-cmpser G f (*Suc n*) ≅ *E*⟧ ⟹
card (*f* ‘ {*i* . *i* ≤ (*Suc* (*Suc n*))}) − *1* =
Suc (*Suc n*) − *card* {*i*. *i* ≤ (*Suc n*) ∧ *Qw-cmpser G f i* ≅ *E*}
⟨*proof*⟩

lemma *length-wcmpser0-6*:⟦*Group G*; *w-cmpser G* (*Suc* (*Suc n*)) *f*⟧ ⟹
0 < *card* (*f* ‘ {*i*. *i* ≤ (*Suc n*)})
⟨*proof*⟩

lemma *length-wcmpser0-7*:⟦*Group G*; *w-cmpser G* (*Suc* (*Suc n*)) *f*⟧ ⟹
card {*i*. *i* ≤ *n* ∧ *Qw-cmpser G f i* ≅ *E*} ≤ *Suc n*
⟨*proof*⟩

lemma *length-wcmpser0*:⟦*Group G*; *Ugp E*⟧ ⟹∀ *f*. *w-cmpser G* (*Suc n*) *f* ⟶
card (*f* ‘ {*i*. *i* ≤ (*Suc n*)}) − *1* = (*Suc n*) − (*card* {*i*. *i* ≤ *n* ∧
((*Qw-cmpser G f*) *i* ≅ *E*)})
⟨*proof*⟩

lemma *length-of-twcmpser*:⟦*Group G*; *Ugp E*; *tw-cmpser G* (*Suc n*) *f* ⟧ ⟹
length-twcmpser G (*Suc n*) *f* =
(*Suc n*) − (*card* {*i*. *i* ≤ *n* ∧ ((*Qw-cmpser G f*) *i* ≅ *E*)})
⟨*proof*⟩

lemma *JHS-1*:⟦*Group G*; *Ugp E*; *compseries G r f*; *compseries G s g*; *0*<*r*; *0* <
s⟧
⟹ *r* = *r* ∗ *s* − *card* {*i*. *i* ≤ (*r* ∗ *s* − *Suc 0*) ∧
Qw-cmpser G (*cmp-rfn G r f s g*) *i* ≅ *E*}
⟨*proof*⟩

lemma *J-H-S*:⟦*Group G*; *Ugp E*; *compseries G r f*; *compseries G s g*; *0*<*r*;
(*0*::*nat*)<*s* ⟧ ⟹ *r* = *s*

⟨*proof*⟩

end

theory *Algebra4*
imports *Algebra3*
begin

3.18 Abelian groups

record *'a aGroup = 'a carrier +*
  pop :: *['a, 'a] ⇒ 'a* (**infixl** ‹±1› *62*)
  mop :: *'a ⇒ 'a* (‹(−$_{a1}$ -)› *[64]63*)
  zero :: *'a* (‹**0**1›)

locale *aGroup =*
 fixes *A* (**structure**)
 assumes
  pop-closed: *pop A ∈ carrier A → carrier A → carrier A*
 and *aassoc* : *⟦a ∈ carrier A; b ∈ carrier A; c ∈ carrier A⟧ ⟹*
  (a ± b) ± c = a ± (b ± c)
 and *pop-commute*:*⟦a ∈ carrier A; b ∈ carrier A⟧ ⟹ a ± b = b ± a*
 and *mop-closed*:*mop A ∈ carrier A → carrier A*
 and *l-m* :*a ∈ carrier A ⟹* (−$_a$ *a*) ± *a* = **0**
 and *ex-zero*: **0** ∈ *carrier A*
 and *l-zero*:*a ∈ carrier A ⟹* **0** ± *a* = *a*

definition
  b-ag :: - ⇒
   (⦇*carrier*:: *'a set*, *top*:: *['a, 'a] ⇒ 'a* , *iop*:: *'a ⇒ 'a*, *one*:: *'a* ⦈ **where**
  b-ag A = (⦇*carrier = carrier A*, *top = pop A*, *iop = mop A*, *one = zero A* ⦈

definition
  asubGroup :: [- , *'a set*] ⇒ *bool* **where**
  asubGroup A H ⟷ (*b-ag A*) » *H*

definition
  aqgrp :: [- , *'a set*] ⇒
    (⦇ *carrier*::*'a set set*, *pop*::[*'a set*, *'a set*] ⇒ *'a set*,
      mop::*'a set* ⇒ *'a set*, *zero* :: *'a set* ⦈ **where**
  aqgrp A H = (⦇*carrier = set-rcs* (*b-ag A*) *H*,
    pop = *λX*. *λY*. (*c-top* (*b-ag A*) *H X Y*),
    mop = *λX*. (*c-iop* (*b-ag A*) *H X*), *zero* = *H* ⦈

definition
  ag-idmap :: - ⇒ (*'a* ⇒ *'a*) (‹(*aI*_)›) **where**
  aI $_A$ = (*λx∈carrier A*. *x*)

abbreviation
ASubG :: [('*a*, '*more*) *aGroup-scheme*, '*a set*] => *bool* (**infixl** ‹+>› *58*) **where**
A +> *H* == *asubGroup A H*

definition
Ag-ind :: [- , '*a* ⇒ '*d*] ⇒ '*d aGroup* **where**
Ag-ind A f = (|*carrier* = *f*‘(*carrier A*),
pop = λ*x* ∈ *f*‘(*carrier A*). λ*y* ∈ *f*‘(*carrier A*).
f(((*invfun* (*carrier A*) (*f*‘(*carrier A*)) *f*) *x*) ±$_A$
((*invfun* (*carrier A*) (*f*‘(*carrier A*)) *f*) *y*)),
mop = λ*x*∈(*f*‘(*carrier A*)). *f* (−$_a$ $_A$ ((*invfun* (*carrier A*) (*f*‘(*carrier A*)) *f*) *x*)),
zero = *f* (**0**$_A$)|)

definition
Agii :: [- , '*a* ⇒ '*d*] ⇒ ('*a* ⇒ '*d*) **where**
Agii A f = (λ*x*∈*carrier A*. *f x*)

lemma (**in** *aGroup*) *ag-carrier-carrier*:*carrier* (*b-ag A*) = *carrier A*
⟨*proof*⟩

lemma (**in** *aGroup*) *ag-pOp-closed*:⟦*x* ∈ *carrier A*; *y* ∈ *carrier A*⟧ ⟹
pop A x y ∈ *carrier A*
⟨*proof*⟩

lemma (**in** *aGroup*) *ag-mOp-closed*:*x* ∈ *carrier A* ⟹ (−$_a$ *x*) ∈ *carrier A*
⟨*proof*⟩

lemma (**in** *aGroup*) *asubg-subset*:*A* +> *H* ⟹ *H* ⊆ *carrier A*
⟨*proof*⟩

lemma (**in** *aGroup*) *ag-pOp-commute*:⟦*x* ∈ *carrier A*; *y* ∈ *carrier A*⟧ ⟹
pop A x y = *pop A y x*
⟨*proof*⟩

lemma (**in** *aGroup*) *b-ag-group*:*Group* (*b-ag A*)
⟨*proof*⟩

lemma (**in** *aGroup*) *agop-gop*:*top* (*b-ag A*) = *pop A*
⟨*proof*⟩

lemma (**in** *aGroup*) *agiop-giop*:*iop* (*b-ag A*) = *mop A*
⟨*proof*⟩

lemma (**in** *aGroup*) *agunit-gone*:*one* (*b-ag A*) = **0**
⟨*proof*⟩

lemma (**in** *aGroup*) *ag-pOp-add-r*:⟦*a* ∈ *carrier A*; *b* ∈ *carrier A*; *c* ∈ *carrier A*;
a = *b*⟧ ⟹ *a* ± *c* = *b* ± *c*
⟨*proof*⟩

lemma (**in** *aGroup*) *ag-add-commute*:$[\![a \in carrier\ A;\ b \in carrier\ A]\!] \Longrightarrow$
$a \pm b = b \pm a$

⟨*proof*⟩

lemma (**in** *aGroup*) *ag-pOp-add-l*:$[\![a \in carrier\ A;\ b \in carrier\ A;\ c \in carrier\ A;$
$a = b]\!] \Longrightarrow c \pm a = c \pm b$

⟨*proof*⟩

lemma (**in** *aGroup*) *asubg-pOp-closed*:$[\![asubGroup\ A\ H;\ x \in H;\ y \in H]\!]$
$\Longrightarrow pop\ A\ x\ y \in H$

⟨*proof*⟩

lemma (**in** *aGroup*) *asubg-mOp-closed*:$[\![asubGroup\ A\ H;\ x \in H]\!] \Longrightarrow -_a\ x \in H$

⟨*proof*⟩

lemma (**in** *aGroup*) *asubg-subset1*:$[\![asubGroup\ A\ H;\ x \in H]\!] \Longrightarrow x \in carrier\ A$

⟨*proof*⟩

lemma (**in** *aGroup*) *asubg-inc-zero*:$asubGroup\ A\ H \Longrightarrow \mathbf{0} \in H$

⟨*proof*⟩

lemma (**in** *aGroup*) *ag-inc-zero*:$\mathbf{0} \in carrier\ A$

⟨*proof*⟩

lemma (**in** *aGroup*) *ag-l-zero*:$x \in carrier\ A \Longrightarrow \mathbf{0} \pm x = x$

⟨*proof*⟩

lemma (**in** *aGroup*) *ag-r-zero*:$x \in carrier\ A \Longrightarrow x \pm \mathbf{0} = x$

⟨*proof*⟩

lemma (**in** *aGroup*) *ag-l-inv1*:$x \in carrier\ A \Longrightarrow (-_a\ x) \pm x = \mathbf{0}$

⟨*proof*⟩

lemma (**in** *aGroup*) *ag-r-inv1*:$x \in carrier\ A \Longrightarrow x \pm (-_a\ x) = \mathbf{0}$

⟨*proof*⟩

lemma (**in** *aGroup*) *ag-pOp-assoc*:$[\![x \in carrier\ A;\ y \in carrier\ A;\ z \in carrier\ A]\!]$
$\Longrightarrow (x \pm y) \pm z = x \pm (y \pm z)$

⟨*proof*⟩

lemma (**in** *aGroup*) *ag-inv-unique*:$[\![x \in carrier\ A;\ y \in carrier\ A;\ x \pm y = \mathbf{0}]\!] \Longrightarrow$
$y = -_a\ x$

⟨*proof*⟩

lemma (**in** *aGroup*) *ag-inv-inj*:$[\![x \in carrier\ A;\ y \in carrier\ A;\ x \neq y]\!] \Longrightarrow$
$(-_a\ x) \neq (-_a\ y)$

⟨*proof*⟩

lemma (**in** *aGroup*) *pOp-assocTr41*:$[\![a \in carrier\ A;\ b \in carrier\ A;\ c \in carrier\ A;$
$d \in carrier\ A]\!] \Longrightarrow a \pm b \pm c \pm d = a \pm b \pm (c \pm d)$
⟨*proof*⟩

lemma (**in** *aGroup*) *pOp-assocTr42*:$[\![a \in carrier\ A;\ b \in carrier\ A;$
$c \in carrier\ A;\ d \in carrier\ A]\!] \Longrightarrow a \pm b \pm c \pm d = a \pm (b \pm c) \pm d$
⟨*proof*⟩

lemma (**in** *aGroup*) *pOp-assocTr43*:$[\![a \in carrier\ A;\ b \in carrier\ A;$
$c \in carrier\ A;\ d \in carrier\ A]\!] \Longrightarrow a \pm b \pm (c \pm d) = a \pm (b \pm c) \pm d$
⟨*proof*⟩

lemma (**in** *aGroup*) *pOp-assoc-cancel*:$[\![a \in carrier\ A;\ b \in carrier\ A;$
$c \in carrier\ A]\!] \Longrightarrow a \pm -_a\ b \pm (b \pm -_a\ c) = a \pm -_a\ c$
⟨*proof*⟩

lemma (**in** *aGroup*) *ag-p-inv*:$[\![x \in carrier\ A;\ y \in carrier\ A]\!] \Longrightarrow$
$(-_a\ (x \pm y)) = (-_a\ x) \pm (-_a\ y)$
⟨*proof*⟩

lemma (**in** *aGroup*) *gEQAddcross*: $[\![l1 \in carrier\ A;\ l2 \in carrier\ A;$
$r1 \in carrier\ A;\ r1 \in carrier\ A;\ l1 = r2;\ l2 = r1]\!] \Longrightarrow$
$l1 \pm l2 = r1 \pm r2$
⟨*proof*⟩

lemma (**in** *aGroup*) *ag-eq-sol1*:$[\![a \in carrier\ A;\ x \in carrier\ A;\ b \in carrier\ A;$
$a \pm x = b]\!] \Longrightarrow x = (-_a\ a) \pm b$
⟨*proof*⟩

lemma (**in** *aGroup*) *ag-eq-sol2*:$[\![a \in carrier\ A;\ x \in carrier\ A;\ b \in carrier\ A;$
$x \pm a = b]\!] \Longrightarrow x = b \pm (-_a\ a)$
⟨*proof*⟩

lemma (**in** *aGroup*) *ag-add4-rel*:$[\![a \in carrier\ A;\ b \in carrier\ A;\ c \in carrier\ A;$
$d \in carrier\ A\ ]\!] \Longrightarrow a \pm b \pm (c \pm d) = \ a \pm c \pm (b \pm d)$
⟨*proof*⟩

lemma (**in** *aGroup*) *ag-inv-inv*:$x \in carrier\ A \Longrightarrow -_a\ (-_a\ x) = x$
⟨*proof*⟩

lemma (**in** *aGroup*) *ag-inv-zero*:$-_a\ \mathbf{0} = \mathbf{0}$
⟨*proof*⟩

lemma (**in** *aGroup*) *ag-diff-minus*:$[\![a \in carrier\ A;\ b \in carrier\ A;\ c \in carrier\ A;$
$a \pm (-_a\ b) = c]\!] \Longrightarrow b \pm (-_a\ a) = (-_a\ c)$
⟨*proof*⟩

lemma (**in** *aGroup*) *pOp-cancel-l*:$[\![a \in carrier\ A;\ b \in carrier\ A;\ c \in carrier\ A;$
$c \pm a = \ c \pm b\ ]\!] \Longrightarrow a = b$

⟨*proof*⟩

lemma (**in** *aGroup*) *pOp-cancel-r*:⟦$a \in$ *carrier* A; $b \in$ *carrier* A; $c \in$ *carrier* A; $a \pm c = b \pm c$ ⟧ $\Longrightarrow a = b$
⟨*proof*⟩

lemma (**in** *aGroup*) *ag-eq-diffzero*:⟦$a \in$ *carrier* A; $b \in$ *carrier* A⟧ $\Longrightarrow$
$(a = b) = (a \pm (-_a\ b) = \mathbf{0})$
⟨*proof*⟩

lemma (**in** *aGroup*) *ag-eq-diffzero1*:⟦$a \in$ *carrier* A; $b \in$ *carrier* A⟧ $\Longrightarrow$
$(a = b) = ((-_a\ a) \pm b = \mathbf{0})$
⟨*proof*⟩

lemma (**in** *aGroup*) *ag-neq-diffnonzero*:⟦$a \in$ *carrier* A; $b \in$ *carrier* A⟧ $\Longrightarrow$
$(a \neq b) = (a \pm (-_a\ b) \neq \mathbf{0})$
⟨*proof*⟩

lemma (**in** *aGroup*) *ag-plus-zero*:⟦$x \in$ *carrier* A; $y \in$ *carrier* A⟧ $\Longrightarrow$
$(x = -_a\ y) = (x \pm y = \mathbf{0})$
⟨*proof*⟩

lemma (**in** *aGroup*) *asubg-nsubg*:A +> $H \Longrightarrow$ (*b-ag* A) ▷ H
⟨*proof*⟩

lemma (**in** *aGroup*) *subg-asubg*:*b-ag* G » $H \Longrightarrow G$ +> H
⟨*proof*⟩

lemma (**in** *aGroup*) *asubg-test*:⟦$H \subseteq$ *carrier* A; $H \neq \{\}$;
$\forall a{\in}H.\ \forall b{\in}H.\ (a \pm (-_a\ b) \in H)$⟧ $\Longrightarrow A$ +> H
⟨*proof*⟩

lemma (**in** *aGroup*) *asubg-zero*:A +> $\{\mathbf{0}\}$
⟨*proof*⟩

lemma (**in** *aGroup*) *asubg-whole*:A +> *carrier* A
⟨*proof*⟩

lemma (**in** *aGroup*) *Ag-ind-carrier*:*bij-to* f (*carrier* A) (D::'*d set*) $\Longrightarrow$
carrier (*Ag-ind* A f) = f ‘ (*carrier* A)
⟨*proof*⟩

lemma (**in** *aGroup*) *Ag-ind-aGroup*:⟦$f \in$ *carrier* $A \to D$;
bij-to f (*carrier* A) (D::'*d set*)⟧ $\Longrightarrow$ *aGroup* (*Ag-ind* A f)
⟨*proof*⟩

3.18.1 Homomorphism of abelian groups

definition

aHom :: [('*a*, '*m*) *aGroup-scheme*, ('*b*, '*m1*) *aGroup-scheme*] ⇒ ('*a* ⇒ '*b*) *set*
where
aHom A B = {*f*. *f* ∈ *carrier A* → *carrier B* ∧ *f* ∈ *extensional* (*carrier A*) ∧
(∀ *a*∈*carrier A*. ∀ *b*∈*carrier A*. *f* (*a* ±$_A$ *b*) = (*f a*) ±$_B$ (*f b*))}

definition
compos :: [('*a*, '*m*) *aGroup-scheme*, '*b* ⇒ '*c*, '*a* ⇒ '*b*] ⇒ '*a* ⇒ '*c* **where**
compos A g f = *compose* (*carrier A*) *g f*

definition
ker :: [('*a*, '*m*) *aGroup-scheme*, ('*b*, '*m1*) *aGroup-scheme*] ⇒ ('*a* ⇒ '*b*)
⇒ '*a set* (‹(*3ker*$_{-,-}$ -)› [*82*,*82*,*83*]*82*) **where**
ker$_{F,G}$ *f* = {*a*. *a* ∈ *carrier F* ∧ *f a* = (**0**$_G$)}

definition
injec :: [('*a*, '*m*) *aGroup-scheme*, ('*b*, '*m1*) *aGroup-scheme*, '*a* ⇒ '*b*]
⇒ *bool* (‹(*3injec*$_{-,-}$ -)› [*82*,*82*,*83*]*82*) **where**
injec$_{F,G}$ *f* ⟷ *f* ∈ *aHom F G* ∧ *ker*$_{F,G}$ *f* = {**0**$_F$}

definition
surjec :: [('*a*, '*m*) *aGroup-scheme*, ('*b*, '*m1*) *aGroup-scheme*, '*a* ⇒ '*b*]
⇒ *bool* (‹(*3surjec*$_{-,-}$ -)› [*82*,*82*,*83*]*82*) **where**
surjec$_{F,G}$ *f* ⟷ *f* ∈ *aHom F G* ∧ *surj-to f* (*carrier F*) (*carrier G*)

definition
bijec :: [('*a*, '*m*) *aGroup-scheme*, ('*b*, '*m1*) *aGroup-scheme*, '*a* ⇒ '*b*]
⇒ *bool* (‹(*3bijec*$_{-,-}$ -)› [*82*,*82*,*83*]*82*) **where**
bijec$_{F,G}$ *f* ⟷ *injec*$_{F,G}$ *f* ∧ *surjec*$_{F,G}$ *f*

definition
ainvf :: [('*a*, '*m*) *aGroup-scheme*, ('*b*, '*m1*) *aGroup-scheme*, '*a* ⇒ '*b*]
⇒ ('*b* ⇒ '*a*) (‹(*3ainvf*$_{-,-}$ -)› [*82*,*82*,*83*]*82*) **where**
ainvf$_{F,G}$ *f* = *invfun* (*carrier F*) (*carrier G*) *f*

lemma *aHom-mem*:⟦*aGroup F*; *aGroup G*; *f* ∈ *aHom F G*; *a* ∈ *carrier F*⟧ ⟹
f a ∈ *carrier G*
⟨*proof*⟩

lemma *aHom-func*:*f* ∈ *aHom F G* ⟹ *f* ∈ *carrier F* → *carrier G*
⟨*proof*⟩

lemma *aHom-add*:⟦*aGroup F*; *aGroup G*; *f* ∈ *aHom F G*; *a* ∈ *carrier F*;
b ∈ *carrier F*⟧ ⟹ *f* (*a* ±$_F$ *b*) = (*f a*) ±$_G$ (*f b*)
⟨*proof*⟩

lemma *aHom-0-0*:⟦*aGroup F*; *aGroup G*; *f* ∈ *aHom F G*⟧ ⟹ *f* (**0**$_F$) = **0**$_G$
⟨*proof*⟩

lemma *ker-inc-zero*:⟦*aGroup F*; *aGroup G*; *f* ∈ *aHom F G*⟧ ⟹ **0**$_F$ ∈ *ker*$_{F,G}$ *f*

⟨*proof*⟩

lemma *aHom-inv-inv*:⟦*aGroup F*; *aGroup G*; $f \in aHom\ F\ G$; $a \in carrier\ F$⟧ ⟹
$f\ (-_{a\,F}\ a) = -_{a\,G}\ (f\ a)$
⟨*proof*⟩

lemma *aHom-compos*:⟦*aGroup L*; *aGroup M*; *aGroup N*; $f \in aHom\ L\ M$; $g \in aHom\ M\ N$ ⟧
⟹ $compos\ L\ g\ f \in aHom\ L\ N$
⟨*proof*⟩

lemma *aHom-compos-assoc*:⟦*aGroup K*; *aGroup L*; *aGroup M*; *aGroup N*; $f \in aHom\ K\ L$;
$g \in aHom\ L\ M$; $h \in aHom\ M\ N$ ⟧ ⟹
$compos\ K\ h\ (compos\ K\ g\ f) = compos\ K\ (compos\ L\ h\ g)\ f$
⟨*proof*⟩

lemma *injec-inj-on*:⟦*aGroup F*; *aGroup G*; $injec_{F,G}\ f$⟧ ⟹ $inj\text{-}on\ f\ (carrier\ F)$
⟨*proof*⟩

lemma *surjec-surj-to*:$surjec_{R,S}\ f$ ⟹ $surj\text{-}to\ f\ (carrier\ R)\ (carrier\ S)$
⟨*proof*⟩

lemma *compos-bijec*:⟦*aGroup E*; *aGroup F*; *aGroup G*; $bijec_{E,F}\ f$; $bijec_{F,G}\ g$⟧ ⟹
$bijec_{E,G}\ (compos\ E\ g\ f)$
⟨*proof*⟩

lemma *ainvf-aHom*:⟦*aGroup F*; *aGroup G*; $bijec_{F,G}\ f$⟧ ⟹
$ainvf_{F,G}\ f \in aHom\ G\ F$
⟨*proof*⟩

lemma *ainvf-bijec*:⟦*aGroup F*; *aGroup G*; $bijec_{F,G}\ f$⟧ ⟹ $bijec_{G,F}\ (ainvf_{F,G}\ f)$
⟨*proof*⟩

lemma *ainvf-l*:⟦*aGroup E*; *aGroup F*; $bijec_{E,F}\ f$; $x \in carrier\ E$⟧ ⟹
$(ainvf_{E,F}\ f)\ (f\ x) = x$
⟨*proof*⟩

lemma (**in** *aGroup*) *aI-aHom*:$aI_A \in aHom\ A\ A$
⟨*proof*⟩

lemma *compos-aI-l*:⟦*aGroup A*; *aGroup B*; $f \in aHom\ A\ B$⟧ ⟹ $compos\ A\ aI_B\ f = f$
⟨*proof*⟩

lemma *compos-aI-r*:⟦*aGroup A*; *aGroup B*; $f \in aHom\ A\ B$⟧ ⟹ $compos\ A\ f\ aI_A = f$
⟨*proof*⟩

lemma *compos-aI-surj*:⟦*aGroup A*; *aGroup B*; $f \in$ *aHom A B*; $g \in$ *aHom B A*;
compos A g f = aI_A⟧ ⟹ $surjec_{B,A}$ *g*
⟨*proof*⟩

lemma *compos-aI-inj*:⟦*aGroup A*; *aGroup B*; $f \in$ *aHom A B*; $g \in$ *aHom B A*;
compos A g f = aI_A⟧ ⟹ $injec_{A,B}$ *f*
⟨*proof*⟩

lemma (**in** *aGroup*) *Ag-ind-aHom*:⟦$f \in$ *carrier A* → *D*;
bij-to f (*carrier A*) (*D*::'*d set*)⟧ ⟹ *Agii A f* ∈ *aHom A* (*Ag-ind A f*)
⟨*proof*⟩

lemma (**in** *aGroup*) *Agii-mem*:⟦$f \in$ *carrier A* → *D*; $x \in$ *carrier A*;
bij-to f (*carrier A*) (*D*::'*d set*)⟧ ⟹ *Agii A f x* ∈ *carrier* (*Ag-ind A f*)
⟨*proof*⟩

lemma *Ag-ind-bijec*:⟦*aGroup A*; $f \in$ *carrier A* → *D*;
bij-to f (*carrier A*) (*D*::'*d set*)⟧ ⟹ $bijec_{A,\ (Ag\text{-}ind\ A\ f)}$ (*Agii A f*)
⟨*proof*⟩

definition
aimg :: [('*b*, '*m1*) *aGroup-scheme*, -, '*b* ⇒ '*a*]
⇒ '*a aGroup* (‹(*3aimg*$_{-,-}$ -)› [*82*,*82*,*83*]*82*) **where**
$aimg_{F,A}$ *f* = *A* (| *carrier* := *f* ‘ (*carrier F*), *pop* := *pop A*, *mop* := *mop A*,
zero := *zero A*|)

lemma *ker-subg*:⟦*aGroup F*; *aGroup G*; $f \in$ *aHom F G* ⟧ ⟹ *F* +> $ker_{F,G}$ *f*
⟨*proof*⟩

3.18.2 Quotient abelian group

definition
ar-coset :: ['*a*, - , '*a set*] ⇒ '*a set*
(‹(*3*- ⊎$_-$ -)› [*66*,*66*,*67*]*66*) **where**
ar-coset a A H = *H* ·$_{(b\text{-}ag\ A)}$ *a*

definition
set-ar-cos :: [- , '*a set*] ⇒ '*a set set* **where**
set-ar-cos A I = {*X*. ∃ *a*∈*carrier A*. *X* = *ar-coset a A I*}

definition
aset-sum :: [- , '*a set*, '*a set*] ⇒ '*a set* **where**
aset-sum A H K = *s-top* (*b-ag A*) *H K*

abbreviation
ASBOP1 (**infix** ‹∓1› *60*) **where**
H ∓$_A$ *K* == *aset-sum A H K*

lemma (**in** *aGroup*) *ag-a-in-ar-cos*:⟦*A* +> *H*; *a* ∈ *carrier A*⟧ ⟹ *a* ∈ *a* ⊎$_A$ *H*

⟨*proof*⟩

lemma (**in** *aGroup*) *r-cos-subset*:⟦*A +> H*; *X* ∈ *set-rcs* (*b-ag A*) *H*⟧ ⟹
X ⊆ *carrier A*
⟨*proof*⟩

lemma (**in** *aGroup*) *asubg-costOp-commute*:⟦*A +> H*; *x* ∈ *set-rcs* (*b-ag A*) *H*;
y ∈ *set-rcs* (*b-ag A*) *H*⟧ ⟹
c-top (*b-ag A*) *H x y* = *c-top* (*b-ag A*) *H y x*
⟨*proof*⟩

lemma (**in** *aGroup*) *Subg-Qgroup*:*A +> H* ⟹ *aGroup* (*aqgrp A H*)
⟨*proof*⟩

lemma (**in** *aGroup*) *plus-subgs*:⟦*A +> H1*; *A +> H2*⟧ ⟹ *A +> H1* ∓ *H2*
⟨*proof*⟩

lemma (**in** *aGroup*) *set-sum*:⟦*H* ⊆ *carrier A*; *K* ⊆ *carrier A*⟧ ⟹
H ∓ *K* = {*x*. ∃ *h*∈*H*. ∃ *k*∈*K*. *x* = *h* ± *k*}
⟨*proof*⟩

lemma (**in** *aGroup*) *mem-set-sum*:⟦*H* ⊆ *carrier A*; *K* ⊆ *carrier A*;
x ∈ *H* ∓ *K* ⟧ ⟹ ∃ *h*∈*H*. ∃ *k*∈*K*. *x* = *h* ± *k*
⟨*proof*⟩

lemma (**in** *aGroup*) *mem-sum-subgs*:⟦*A +> H*; *A +> K*; *h* ∈ *H*; *k* ∈ *K*⟧ ⟹
h ± *k* ∈ *H* ∓ *K*
⟨*proof*⟩

lemma (**in** *aGroup*) *aqgrp-carrier*:*A +> H* ⟹
set-rcs (*b-ag A*) *H* = *set-ar-cos A H*
⟨*proof*⟩

lemma (**in** *aGroup*) *unit-in-set-ar-cos*:*A +> H* ⟹ *H* ∈ *set-ar-cos A H*
⟨*proof*⟩

lemma (**in** *aGroup*) *aqgrp-pOp-maps*:⟦*A +> H*; *a* ∈ *carrier A*; *b* ∈ *carrier A*⟧ ⟹
pop (*aqgrp A H*) (*a* ⊎$_A$ *H*) (*b* ⊎$_A$ *H*) = (*a* ± *b*) ⊎$_A$ *H*
⟨*proof*⟩

lemma (**in** *aGroup*) *aqgrp-mOp-maps*:⟦*A +> H*; *a* ∈ *carrier A*⟧ ⟹
mop (*aqgrp A H*) (*a* ⊎$_A$ *H*) = (−$_a$ *a*) ⊎$_A$ *H*
⟨*proof*⟩

lemma (**in** *aGroup*) *aqgrp-zero*:*A +> H* ⟹ *zero* (*aqgrp A H*) = *H*
⟨*proof*⟩

lemma (**in** *aGroup*) *arcos-fixed*:⟦*A +> H*; *a* ∈ *carrier A*; *h* ∈ *H* ⟧ ⟹
a ⊎$_A$ *H* = (*h* ± *a*) ⊎$_A$ *H*

⟨*proof*⟩

definition
rind-hom :: [('*a*, '*more*) *aGroup-scheme*, ('*b*, '*more1*) *aGroup-scheme*, ('*a* ⇒ '*b*)] ⇒ ('*a set* ⇒ '*b*) **where**
rind-hom A B f = (λ*X*∈(*set-ar-cos A* (*ker*$_{A,B}$ *f*)). *f* (*SOME x*. *x* ∈ *X*))

abbreviation
RIND-HOM (‹(*3*-°-,-)› [*82*,*82*,*83*]*82*) **where**
f°$_{F,G}$ == *rind-hom F G f*

3.19 Direct product and direct sum of abelian groups, in general case

definition
Un-carrier :: ['*i set*, '*i* ⇒ ('*a*, '*more*) *aGroup-scheme*] ⇒ '*a set* **where**
Un-carrier I A = ⋃{*X*. ∃ *i*∈*I*. *X* = *carrier* (*A i*)}

definition
carr-prodag :: ['*i set*, '*i* ⇒ ('*a*, '*more*) *aGroup-scheme*] ⇒ ('*i* ⇒ '*a*) *set* **where**
carr-prodag I A = {*f*. *f* ∈ *extensional I* ∧ *f* ∈ *I* → (*Un-carrier I A*) ∧ (∀ *i*∈*I*. *f i* ∈ *carrier* (*A i*))}

definition
prod-pOp :: ['*i set*, '*i* ⇒ ('*a*, '*more*) *aGroup-scheme*] ⇒ ('*i* ⇒ '*a*) ⇒ ('*i* ⇒ '*a*) ⇒ ('*i* ⇒ '*a*) **where**
prod-pOp I A = (λ*f*∈*carr-prodag I A*. λ*g*∈*carr-prodag I A*. λ*x*∈*I*. (*f x*) ±$_{(A\ x)}$ (*g x*))

definition
prod-mOp :: ['*i set*, '*i* ⇒ ('*a*, '*more*) *aGroup-scheme*] ⇒ ('*i* ⇒ '*a*) ⇒ ('*i* ⇒ '*a*) **where**
prod-mOp I A = (λ*f*∈*carr-prodag I A*. λ*x*∈*I*. (−$_{a(A\ x)}$ (*f x*)))

definition
prod-zero :: ['*i set*, '*i* ⇒ ('*a*, '*more*) *aGroup-scheme*] ⇒ ('*i* ⇒ '*a*) **where**
prod-zero I A = (λ*x*∈*I*. **0**$_{(A\ x)}$)

definition
prodag :: ['*i set*, '*i* ⇒ ('*a*, '*more*) *aGroup-scheme*] ⇒ ('*i* ⇒ '*a*) *aGroup* **where**
prodag I A = (| *carrier* = *carr-prodag I A*, *pop* = *prod-pOp I A*, *mop* = *prod-mOp I A*, *zero* = *prod-zero I A* |)

definition
PRoject :: ['*i set*, '*i* ⇒ ('*a*, '*more*) *aGroup-scheme*, '*i*] ⇒ ('*i* ⇒ '*a*) ⇒ '*a* (‹(*3*π-,-,-)› [*82*,*82*,*83*]*82*) **where**
PRoject I A x = (λ*f* ∈ *carr-prodag I A*. *f x*)

abbreviation
$PRODag\ \ (\langle(a\Pi_{-}\ \text{-})\rangle\ [72,73]\,72)$ **where**
$a\Pi_I\ A == prodag\ I\ A$

lemma *prodag-comp-i*:$[\![a \in carr\text{-}prodag\ I\ A;\ i \in I]\!] \Longrightarrow (a\ i) \in carrier\ (A\ i)$
⟨*proof*⟩

lemma *prod-pOp-func*:$\forall\, k{\in}I.\ aGroup\ (A\ k) \Longrightarrow$
$prod\text{-}pOp\ I\ A \in carr\text{-}prodag\ I\ A \to carr\text{-}prodag\ I\ A \to carr\text{-}prodag\ I\ A$
⟨*proof*⟩

lemma *prod-pOp-mem*:$[\![\forall\, k{\in}I.\ aGroup\ (A\ k);\ X \in carr\text{-}prodag\ I\ A;$
$Y \in carr\text{-}prodag\ I\ A]\!] \Longrightarrow prod\text{-}pOp\ I\ A\ X\ Y \in carr\text{-}prodag\ I\ A$
⟨*proof*⟩

lemma *prod-pOp-mem-i*:$[\![\forall\, k{\in}I.\ aGroup\ (A\ k);\ X \in carr\text{-}prodag\ I\ A;$
$Y \in carr\text{-}prodag\ I\ A;\ i \in I]\!] \Longrightarrow prod\text{-}pOp\ I\ A\ X\ Y\ i = (X\ i) \pm_{(A\ i)} (Y\ i)$
⟨*proof*⟩

lemma *prod-mOp-func*:$\forall\, k{\in}I.\ aGroup\ (A\ k) \Longrightarrow$
$prod\text{-}mOp\ I\ A \in carr\text{-}prodag\ I\ A \to carr\text{-}prodag\ I\ A$
⟨*proof*⟩

lemma *prod-mOp-mem*:$[\![\forall\, j{\in}I.\ aGroup\ (A\ j);\ X \in carr\text{-}prodag\ I\ A]\!] \Longrightarrow$
$prod\text{-}mOp\ I\ A\ X \in carr\text{-}prodag\ I\ A$
⟨*proof*⟩

lemma *prod-mOp-mem-i*:$[\![\forall\, j{\in}I.\ aGroup\ (A\ j);\ X \in carr\text{-}prodag\ I\ A;\ i \in I]\!] \Longrightarrow$
$prod\text{-}mOp\ I\ A\ X\ i = -_{a(A\ i)}\ (X\ i)$
⟨*proof*⟩

lemma *prod-zero-func*:$\forall\, k{\in}I.\ aGroup\ (A\ k) \Longrightarrow$
$prod\text{-}zero\ I\ A \in carr\text{-}prodag\ I\ A$
⟨*proof*⟩

lemma *prod-zero-i*:$[\![\forall\, k{\in}I.\ aGroup\ (A\ k);\ i \in I]\!] \Longrightarrow$
$prod\text{-}zero\ I\ A\ i = \mathbf{0}_{(A\ i)}$
⟨*proof*⟩

lemma *carr-prodag-mem-eq*:$[\![\forall\, k{\in}I.\ aGroup\ (A\ k);\ X \in carr\text{-}prodag\ I\ A;$
$Y \in carr\text{-}prodag\ I\ A;\ \forall\, l{\in}I.\ (X\ l) = (Y\ l)\ ]\!] \Longrightarrow X = Y$
⟨*proof*⟩

lemma *prod-pOp-assoc*:$[\![\forall\, k{\in}I.\ aGroup\ (A\ k);\ a \in carr\text{-}prodag\ I\ A;$
$b \in carr\text{-}prodag\ I\ A;\ c \in carr\text{-}prodag\ I\ A]\!] \Longrightarrow$
$prod\text{-}pOp\ I\ A\ (prod\text{-}pOp\ I\ A\ a\ b)\ c =$
$prod\text{-}pOp\ I\ A\ a\ (prod\text{-}pOp\ I\ A\ b\ c)$
⟨*proof*⟩

lemma *prod-pOp-commute*:$[\![\forall\, k{\in}I.\ aGroup\ (A\ k);\ a \in carr\text{-}prodag\ I\ A;$
$b \in carr\text{-}prodag\ I\ A]\!] \Longrightarrow$
$prod\text{-}pOp\ I\ A\ a\ b = prod\text{-}pOp\ I\ A\ b\ a$
⟨*proof*⟩

lemma *prodag-aGroup*:$\forall\, k{\in}I.\ aGroup\ (A\ k) \Longrightarrow aGroup\ (prodag\ I\ A)$
⟨*proof*⟩

lemma *prodag-carrier*:$\forall\, k{\in}I.\ aGroup\ (A\ k) \Longrightarrow$
$carrier\ (prodag\ I\ A) = carr\text{-}prodag\ I\ A$
⟨*proof*⟩

lemma *prodag-elemfun*:$[\![\forall\, k{\in}I.\ aGroup\ (A\ k);\ f \in carrier\ (prodag\ I\ A)]\!] \Longrightarrow$
$f \in extensional\ I$
⟨*proof*⟩

lemma *prodag-component*:$[\![f \in carrier\ (prodag\ I\ A);\ i \in I\ ]\!] \Longrightarrow$
$f\ i \in carrier\ (A\ i)$
⟨*proof*⟩

lemma *prodag-pOp*:$\forall\, k{\in}I.\ aGroup\ (A\ k) \Longrightarrow$
$pop\ (prodag\ I\ A) = prod\text{-}pOp\ I\ A$
⟨*proof*⟩

lemma *prodag-iOp*:$\forall\, k{\in}I.\ aGroup\ (A\ k) \Longrightarrow$
$mop\ (prodag\ I\ A) = prod\text{-}mOp\ I\ A$
⟨*proof*⟩

lemma *prodag-zero*:$\forall\, k{\in}I.\ aGroup\ (A\ k) \Longrightarrow$
$zero\ (prodag\ I\ A) = prod\text{-}zero\ I\ A$
⟨*proof*⟩

lemma *prodag-sameTr0*:$[\![\forall\, k{\in}I.\ aGroup\ (A\ k);\ \forall\, k{\in}I.\ A\ k = B\ k]\!]$
$\Longrightarrow Un\text{-}carrier\ I\ A = Un\text{-}carrier\ I\ B$
⟨*proof*⟩

lemma *prodag-sameTr1*:$[\![\forall\, k{\in}I.\ aGroup\ (A\ k);\ \forall\, k{\in}I.\ A\ k = B\ k]\!]$
$\Longrightarrow carr\text{-}prodag\ I\ A = carr\text{-}prodag\ I\ B$
⟨*proof*⟩

lemma *prodag-sameTr2*:$[\![\forall\, k{\in}I.\ aGroup\ (A\ k);\ \forall\, k{\in}I.\ A\ k = B\ k]\!]$
$\Longrightarrow prod\text{-}pOp\ I\ A = prod\text{-}pOp\ I\ B$
⟨*proof*⟩

lemma *prodag-sameTr3*:$[\![\forall\, k{\in}I.\ aGroup\ (A\ k);\ \forall\, k{\in}I.\ A\ k = B\ k]\!]$
$\Longrightarrow prod\text{-}mOp\ I\ A = prod\text{-}mOp\ I\ B$
⟨*proof*⟩

lemma *prodag-sameTr4*:$[\![\forall\, k{\in}I.\ aGroup\ (A\ k);\ \forall\, k{\in}I.\ A\ k = B\ k]\!]$
$\Longrightarrow$ *prod-zero I A* = *prod-zero I B*
⟨*proof*⟩

lemma *prodag-same*:$[\![\forall\, k{\in}I.\ aGroup\ (A\ k);\ \forall\, k{\in}I.\ A\ k = B\ k]\!]$
$\Longrightarrow$ *prodag I A* = *prodag I B*
⟨*proof*⟩

lemma *project-mem*:$[\![\forall\, k{\in}I.\ aGroup\ (A\ k);\ j \in I;\ x \in carrier\ (prodag\ I\ A)]\!] \Longrightarrow$
$(PRoject\ I\ A\ j)\ x\ \in carrier\ (A\ j)$
⟨*proof*⟩

lemma *project-aHom*:$[\![\forall\, k{\in}I.\ aGroup\ (A\ k);\ j \in I]\!] \Longrightarrow$
$PRoject\ I\ A\ j \in aHom\ (prodag\ I\ A)\ (A\ j)$
⟨*proof*⟩

lemma *project-aHom1*:$\forall\, k{\in}I.\ aGroup\ (A\ k) \Longrightarrow$
$\forall\, j \in I.\ PRoject\ I\ A\ j \in aHom\ (prodag\ I\ A)\ (A\ j)$
⟨*proof*⟩

definition
A-to-prodag :: [('*a*, '*m*) *aGroup-scheme*, '*i set*, '*i* ⇒('*a* ⇒ '*b*),
'*i* ⇒ ('*b*, '*m1*) *aGroup-scheme*] ⇒ ('*a* ⇒ ('*i* ⇒'*b*)) **where**
A-to-prodag A I S B = (λ*a*∈*carrier A*. λ*k*∈*I*. *S k a*)

lemma *A-to-prodag-mem*:$[\![aGroup\ A;\ \forall\, k{\in}I.\ aGroup\ (B\ k);\ \forall\, k{\in}I.\ (S\ k) \in$
$aHom\ A\ (B\ k);\ x \in carrier\ A\ ]\!] \Longrightarrow A\text{-}to\text{-}prodag\ A\ I\ S\ B\ x \in carr\text{-}prodag\ I\ B$
⟨*proof*⟩

lemma *A-to-prodag-aHom*:$[\![aGroup\ A;\ \forall\, k{\in}I.\ aGroup\ (B\ k);\ \forall\, k{\in}I.\ (S\ k) \in$
$aHom\ A\ (B\ k)\ ]\!] \Longrightarrow A\text{-}to\text{-}prodag\ A\ I\ S\ B \in aHom\ A\ (a\Pi_I\ B)$
⟨*proof*⟩

definition
finiteHom :: ['*i set*, '*i* ⇒ ('*a*, '*more*) *aGroup-scheme*, '*i* ⇒ '*a*] ⇒ *bool* **where**
finiteHom I A f ⟷ $f \in carr\text{-}prodag\ I\ A \wedge (\exists\, H.\ H \subseteq I \wedge finite\ H \wedge ($
$\forall\, j \in (I - H).\ (f\ j) = \mathbf{0}_{(A\ j)}))$

definition
carr-dsumag :: ['*i set*, '*i* ⇒ ('*a*, '*more*) *aGroup-scheme*] ⇒ ('*i* ⇒ '*a*) *set* **where**
carr-dsumag I A = {*f*. *finiteHom I A f*}

definition
dsumag :: ['*i set*, '*i* ⇒ ('*a*, '*more*) *aGroup-scheme*] ⇒ ('*i* ⇒ '*a*) *aGroup* **where**
dsumag I A = (| *carrier* = *carr-dsumag I A*,
pop = *prod-pOp I A*, *mop* = *prod-mOp I A*,
zero = *prod-zero I A*|)

definition
dProj :: [*'i set*, *'i* ⇒ (*'a*, *'more*) *aGroup-scheme*, *'i*]
⇒ (*'i* ⇒ *'a*) ⇒ *'a* **where**
dProj I A x = (λ*f*∈*carr-dsumag I A*. *f x*)

abbreviation
DSUMag (‹(*a*⨁ - -)› [*72*,*73*] *72*) **where**
a⨁ *I A* == *dsumag I A*

lemma *dsum-pOp-func*:∀ *k*∈*I*. *aGroup* (*A k*) ⟹
prod-pOp I A ∈ *carr-dsumag I A* → *carr-dsumag I A* → *carr-dsumag I A*
⟨*proof*⟩

lemma *dsum-pOp-mem*:⟦∀ *k*∈*I*. *aGroup* (*A k*); *X* ∈ *carr-dsumag I A*;
Y ∈ *carr-dsumag I A*⟧ ⟹ *prod-pOp I A X Y* ∈ *carr-dsumag I A*
⟨*proof*⟩

lemma *dsum-iOp-func*:∀ *k*∈*I*. *aGroup* (*A k*) ⟹
prod-mOp I A ∈ *carr-dsumag I A* → *carr-dsumag I A*
⟨*proof*⟩

lemma *dsum-iOp-mem*:⟦∀ *j*∈*I*. *aGroup* (*A j*); *X* ∈ *carr-dsumag I A*⟧ ⟹
prod-mOp I A X ∈ *carr-dsumag I A*
⟨*proof*⟩

lemma *dsum-zero-func*:∀ *k*∈*I*. *aGroup* (*A k*) ⟹
prod-zero I A ∈ *carr-dsumag I A*
⟨*proof*⟩

lemma *dsumag-sub-prodag*:∀ *k*∈*I*. *aGroup* (*A k*) ⟹
carr-dsumag I A ⊆ *carr-prodag I A*
⟨*proof*⟩

lemma *carrier-dsumag*:∀ *k*∈*I*. *aGroup* (*A k*) ⟹
carrier (*dsumag I A*) = *carr-dsumag I A*
⟨*proof*⟩

lemma *dsumag-elemfun*:⟦∀ *k*∈*I*. *aGroup* (*A k*); *f* ∈ *carrier* (*dsumag I A*)⟧ ⟹
f ∈ *extensional I*
⟨*proof*⟩

lemma *dsumag-aGroup*:∀ *k*∈*I*. *aGroup* (*A k*) ⟹ *aGroup* (*dsumag I A*)
⟨*proof*⟩

lemma *dsumag-pOp*:∀ *k*∈*I*. *aGroup* (*A k*) ⟹
pop (*dsumag I A*) = *prod-pOp I A*
⟨*proof*⟩

lemma *dsumag-mOp*:$\forall\, k{\in}I.\ aGroup\ (A\ k) \Longrightarrow$
$mop\ (dsumag\ I\ A) = prod\text{-}mOp\ I\ A$
⟨*proof*⟩

lemma *dsumag-zero*:$\forall\, k{\in}I.\ aGroup\ (A\ k) \Longrightarrow$
$zero\ (dsumag\ I\ A) = prod\text{-}zero\ I\ A$
⟨*proof*⟩

3.19.1 Characterization of a direct product

lemma *direct-prod-mem-eq*:$[\![\forall\, j{\in}I.\ aGroup\ (A\ j);\ f \in carrier\ (a\Pi_I\ A);$
$g \in carrier\ (a\Pi_I\ A);\ \forall\, j{\in}I.\ (PRoject\ I\ A\ j)\ f = (PRoject\ I\ A\ j)\ g]\!] \Longrightarrow$
$f = g$
⟨*proof*⟩

lemma *map-family-fun*:$[\![\forall\, j{\in}I.\ aGroup\ (A\ j);\ aGroup\ S;$
$\forall\, j{\in}I.\ ((g\ j) \in aHom\ S\ (A\ j));\ x \in carrier\ S]\!] \Longrightarrow$
$(\lambda y \in carrier\ S.\ (\lambda j{\in}I.\ (g\ j)\ y))\ x \in carrier\ (a\Pi_I\ A)$
⟨*proof*⟩

lemma *map-family-aHom*:$[\![\forall\, j{\in}I.\ aGroup\ (A\ j);\ aGroup\ S;$
$\forall\, j{\in}I.\ ((g\ j) \in aHom\ S\ (A\ j))]\!] \Longrightarrow$
$(\lambda y \in carrier\ S.\ (\lambda j{\in}I.\ (g\ j)\ y)) \in aHom\ S\ (a\Pi_I\ A)$
⟨*proof*⟩

lemma *map-family-triangle*:$[\![\forall\, j{\in}I.\ aGroup\ (A\ j);\ aGroup\ S;$
$\forall\, j{\in}I.\ ((g\ j) \in aHom\ S\ (A\ j))]\!] \Longrightarrow \exists!f.\ f \in aHom\ S\ (a\Pi_I\ A) \wedge$
$(\forall\, j{\in}I.\ compos\ S\ (PRoject\ I\ A\ j)\ f = \ (g\ j))$
⟨*proof*⟩

lemma *Ag-ind-triangle*:$[\![\forall\, j{\in}I.\ aGroup\ (A\ j);\ j \in I;\ f \in carrier\ (a\Pi_I\ A) \to B;$
$bij\text{-}to\ f\ (carrier\ (a\Pi_I\ A))\ (B{::}'d\ set);\ j \in I]\!] \Longrightarrow$
$compos\ (a\Pi_I\ A)\ (compos\ (Ag\text{-}ind\ (a\Pi_I\ A)\ f)(PRoject\ I\ A\ j)\ (ainvf_{(a\Pi_I\ A)},\ (Ag\text{-}ind\ (a\Pi_I\ A)\ f)$
$(Agii\ (a\Pi_I\ A)\ f)))\ (Agii\ (a\Pi_I\ A)\ f) =$
$PRoject\ I\ A\ j$
⟨*proof*⟩

definition
$ProjInd :: ['i\ set,\ 'i \Rightarrow ('a,\ 'm)\ aGroup\text{-}scheme,\ ('i \Rightarrow 'a) \Rightarrow 'd,\ 'i] \Rightarrow$
$('d \Rightarrow 'a)$ **where**
$ProjInd\ I\ A\ f\ j = compos\ (Ag\text{-}ind\ (a\Pi_I\ A)\ f)(PRoject\ I\ A\ j)\ (ainvf_{(a\Pi_I\ A)},\ (Ag\text{-}ind\ (a\Pi_I\ A)\ f)$
$(Agii\ (a\Pi_I\ A)\ f))$

lemma *ProjInd-aHom*:$[\![\forall\, j{\in}\ I.\ aGroup\ (A\ j);\ j \in I;\ f \in carrier\ (a\Pi_I\ A) \to B;$
$bij\text{-}to\ f\ (carrier\ (a\Pi_I\ A))\ (B{::}'d\ set);\ j \in I]\!] \Longrightarrow$

$(ProjInd\ I\ A\ f\ j) \in aHom\ (Ag\text{-}ind\ (a\Pi_I\ A)\ f)\ (A\ j)$
⟨*proof*⟩

lemma *ProjInd-aHom1*:⟦$\forall j{\in} I.\ aGroup\ (A\ j);\ f \in carrier\ (a\Pi_I\ A) \to B;$
$bij\text{-}to\ f\ (carrier\ (a\Pi_I\ A))\ (B::'d\ set)$⟧ ⟹
$\forall j{\in}I.\ (ProjInd\ I\ A\ f\ j) \in aHom\ (Ag\text{-}ind\ (a\Pi_I\ A)\ f)\ (A\ j)$
⟨*proof*⟩

lemma *ProjInd-mem-eq*:⟦$\forall j{\in}I.\ aGroup\ (A\ j);\ f \in carrier\ (a\Pi_I\ A) \to B;$
$bij\text{-}to\ f\ (carrier\ (a\Pi_I\ A))\ B;\ aGroup\ S;\ x \in carrier\ (Ag\text{-}ind\ (a\Pi_I\ A)\ f);$
$y \in carrier\ (Ag\text{-}ind\ (a\Pi_I\ A)\ f);$
$\forall j{\in}I.\ (ProjInd\ I\ A\ f\ j\ x = ProjInd\ I\ A\ f\ j\ y)$⟧ ⟹ $x = y$
⟨*proof*⟩

lemma *ProjInd-mem-eq1*:⟦$\forall j{\in}I.\ aGroup\ (A\ j);\ f \in carrier\ (a\Pi_I\ A) \to B;$
$bij\text{-}to\ f\ (carrier\ (a\Pi_I\ A))\ B;\ aGroup\ S;$
$h \in aHom\ (Ag\text{-}ind\ (a\Pi_I\ A)\ f)\ (Ag\text{-}ind\ (a\Pi_I\ A)\ f);$
$\forall j{\in}I.\ compos\ (Ag\text{-}ind\ (a\Pi_I\ A)\ f)\ (ProjInd\ I\ A\ f\ j)\ h = ProjInd\ I\ A\ f\ j$⟧
⟹ $h = ag\text{-}idmap\ (Ag\text{-}ind\ (a\Pi_I\ A)\ f)$
⟨*proof*⟩

lemma *Ag-ind-triangle1*:⟦$\forall j{\in}I.\ aGroup\ (A\ j);\ f \in carrier\ (a\Pi_I\ A) \to B;$
$bij\text{-}to\ f\ (carrier\ (a\Pi_I\ A))\ (B::'d\ set);\ j \in I$⟧ ⟹
$compos\ (a\Pi_I\ A)\ (ProjInd\ I\ A\ f\ j)\ (Agii\ (a\Pi_I\ A)\ f) = PRoject\ I\ A\ j$
⟨*proof*⟩

lemma *map-family-triangle1*:⟦$\forall j{\in}I.\ aGroup\ (A\ j);\ f \in carrier\ (a\Pi_I\ A) \to B;$
$bij\text{-}to\ f\ (carrier\ (a\Pi_I\ A))\ (B::'d\ set);\ aGroup\ S;$
$\forall j{\in}I.\ ((g\ j) \in aHom\ S\ (A\ j))$⟧ ⟹ $\exists!h.\ h \in aHom\ S\ (Ag\text{-}ind\ (a\Pi_I\ A)\ f) \wedge$
$(\forall j{\in}I.\ compos\ S\ (ProjInd\ I\ A\ f\ j)\ h = (g\ j))$
⟨*proof*⟩

lemma *map-family-triangle2*:⟦$I \neq \{\};\ \forall j{\in}I.\ aGroup\ (A\ j);\ aGroup\ S;$
$\forall j{\in}I.\ g\ j \in aHom\ S\ (A\ j);\ ff \in carrier\ (a\Pi_I\ A) \to B;$
$bij\text{-}to\ ff\ (carrier\ (a\Pi_I\ A))\ B;$
$h1 \in aHom\ (Ag\text{-}ind\ (a\Pi_I\ A)\ ff)\ S;$
$\forall j{\in}I.\ compos\ (Ag\text{-}ind\ (a\Pi_I\ A)\ ff)\ (g\ j)\ h1 = ProjInd\ I\ A\ ff\ j;$
$h2 \in aHom\ S\ (Ag\text{-}ind\ (a\Pi_I\ A)\ ff);$
$\forall j{\in}I.\ compos\ S\ (ProjInd\ I\ A\ ff\ j)\ h2 = g\ j$⟧
⟹ $\forall j{\in}I.\ compos\ (Ag\text{-}ind\ (a\Pi_I\ A)\ ff)\ (ProjInd\ I\ A\ ff\ j)$
$(compos\ (Ag\text{-}ind\ (a\Pi_I\ A)\ ff)\ h2\ h1) =$
$ProjInd\ I\ A\ ff\ j$
⟨*proof*⟩

lemma *map-family-triangle3*:⟦$\forall j{\in}I.\ aGroup\ (A\ j);\ aGroup\ S;\ aGroup\ S1;$
$\forall j{\in}I.\ f\ j \in aHom\ S\ (A\ j);\ \forall j{\in}I.\ g\ j \in aHom\ S1\ (A\ j);$
$h1 \in aHom\ S1\ S;\ h2 \in aHom\ S\ S1;$
$\forall j{\in}I.\ compos\ S\ (g\ j)\ h2 = f\ j;$
$\forall j{\in}I.\ compos\ S1\ (f\ j)\ h1 = g\ j$⟧

$\Longrightarrow \forall\, j{\in}I.\ compos\ S\ (f\ j)\ (compos\ S\ h1\ h2) = f\ j$
⟨*proof*⟩

lemma *map-family-triangle4*:⟦$\forall\, j{\in}I.\ aGroup\ (A\ j)$; $aGroup\ S$;
$\forall\, j{\in}I.\ f\ j \in aHom\ S\ (A\ j)$⟧ $\Longrightarrow$
$\forall\, j{\in}I.\ compos\ S\ (f\ j)\ (ag\text{-}idmap\ S) = f\ j$
⟨*proof*⟩

lemma *prod-triangle*:⟦$I \neq \{\}$; $\forall\, j{\in}I.\ aGroup\ (A\ j)$; $aGroup\ S$;
$\forall\, j{\in}I.\ g\ j \in aHom\ S\ (A\ j)$; $ff \in carrier\ (a\Pi_I\ A) \to B$;
$bij\text{-}to\ ff\ (carrier\ (a\Pi_I\ A))\ B$;
$h1 \in aHom\ (Ag\text{-}ind\ (a\Pi_I\ A)\ ff)\ S$;
$\forall\, j{\in}I.\ compos\ (Ag\text{-}ind\ (a\Pi_I\ A)\ ff)\ (g\ j)\ h1 = ProjInd\ I\ A\ ff\ j$;
$h2 \in aHom\ S\ (Ag\text{-}ind\ (a\Pi_I\ A)\ ff)$;
$\forall\, j{\in}I.\ compos\ S\ (ProjInd\ I\ A\ ff\ j)\ h2 = g\ j$⟧
$\Longrightarrow (compos\ (Ag\text{-}ind\ (a\Pi_I\ A)\ ff)\ h2\ h1) = ag\text{-}idmap\ (Ag\text{-}ind\ (a\Pi_I\ A)\ ff)$
⟨*proof*⟩

lemma *characterization-prodag*:⟦$I \neq \{\}$; $\forall\, j{\in}(I{::}'i\ set).\ aGroup\ ((A\ j){::}$
$('a, 'm)\ aGroup\text{-}scheme)$; $aGroup\ (S{::}'d\ aGroup)$;
$\forall\, j{\in}I.\ ((g\ j) \in aHom\ S\ (A\ j))$; $\exists\, ff.\ ff \in carrier\ (a\Pi_I\ A) \to (B{::}'d\ set) \wedge$
$bij\text{-}to\ ff\ (carrier\ (a\Pi_I\ A))\ B$;
$\forall\, (S'{::}\ 'd\ aGroup).\ aGroup\ S' \longrightarrow$
$(\forall\, g'.\ (\forall\, j{\in}I.\ (g'\ j) \in aHom\ S'\ (A\ j) \longrightarrow$
$(\exists!\ f.\ f \in aHom\ S'\ S \wedge (\forall\, j{\in}I.\ compos\ S'\ (g\ j)\ f = (g'\ j)))))$⟧ $\Longrightarrow$
$\exists\, h.\ bijec_{(prodag\ I\ A),S}\ h$
⟨*proof*⟩

Chapter 4

Ring theory

4.1 Definition of a ring and an ideal

record $'a$ *Ring* = $'a$ *aGroup* +
  tp :: $['a, 'a] \Rightarrow 'a$ (**infixl** $\langle \cdot_r 1 \rangle$ *70*)
  un :: $'a$ $(\langle 1_r 1 \rangle)$

locale *Ring* =
fixes R (**structure**)

assumes
  pop-closed: $pop\ R \in carrier\ R \to carrier\ R \to carrier\ R$
and *pop-aassoc* : $[\![a \in carrier\ R;\ b \in carrier\ R;\ c \in carrier\ R]\!] \Longrightarrow (a \pm b) \pm c = a \pm (b \pm c)$
and *pop-commute*: $[\![a \in carrier\ R;\ b \in carrier\ R]\!] \Longrightarrow a \pm b = b \pm a$
and *mop-closed*: $mop\ R \in carrier\ R \to carrier\ R$
and *l-m* : $a \in carrier\ R \Longrightarrow (-_a\ a) \pm a = \mathbf{0}$
and *ex-zero*: $\mathbf{0} \in carrier\ R$
and *l-zero*: $a \in carrier\ R \Longrightarrow \mathbf{0} \pm a = a$
and *tp-closed*: $tp\ R \in carrier\ R \to carrier\ R \to carrier\ R$
and *tp-assoc* : $[\![a \in carrier\ R;\ b \in carrier\ R;\ c \in carrier\ R]\!] \Longrightarrow (a \cdot_r b) \cdot_r c = a \cdot_r (b \cdot_r c)$
and *tp-commute*: $[\![a \in carrier\ R;\ b \in carrier\ R]\!] \Longrightarrow a \cdot_r b = b \cdot_r a$
and *un-closed*: $(1_r) \in carrier\ R$
and *rg-distrib*: $[\![a \in carrier\ R;\ b \in carrier\ R;\ c \in carrier\ R]\!] \Longrightarrow a \cdot_r (b \pm c) = a \cdot_r b \pm a \cdot_r c$
and *rg-l-unit*: $a \in carrier\ R \Longrightarrow (1_r) \cdot_r a = a$

definition
  zeroring :: $('a, 'more)$ *Ring-scheme* $\Rightarrow$ *bool* **where**
  $zeroring\ R \longleftrightarrow Ring\ R \wedge carrier\ R = \{\mathbf{0}_R\}$

primrec *nscal* :: $('a, 'more)$ *Ring-scheme* => $'a$ => *nat* => $'a$
where
  nscal-0: $nscal\ R\ x\ 0 = \mathbf{0}_R$

| *nscal-suc*: $nscal\ R\ x\ (Suc\ n) = (nscal\ R\ x\ n) \pm_R x$

primrec *npow* :: $('a, 'more)\ Ring\text{-}scheme \Rightarrow 'a \Rightarrow nat \Rightarrow 'a$
where
npow-0: $npow\ R\ x\ 0 = 1_{r\,R}$
| *npow-suc*: $npow\ R\ x\ (Suc\ n) = (npow\ R\ x\ n) \cdot_{r\,R} x$

primrec *nprod* :: $('a, 'more)\ Ring\text{-}scheme \Rightarrow (nat \Rightarrow 'a) \Rightarrow nat \Rightarrow 'a$
where
nprod-0: $nprod\ R\ f\ 0 = f\ 0$
| *nprod-suc*: $nprod\ R\ f\ (Suc\ n) = (nprod\ R\ f\ n) \cdot_{r\,R} (f\ (Suc\ n))$

primrec *nsum* :: $('a, 'more)\ aGroup\text{-}scheme \Rightarrow (nat \Rightarrow 'a) \Rightarrow nat \Rightarrow 'a$
where
nsum-0: $nsum\ R\ f\ 0 = f\ 0$
| *nsum-suc*: $nsum\ R\ f\ (Suc\ n) = (nsum\ R\ f\ n) \pm_R (f\ (Suc\ n))$

abbreviation
$NSCAL :: [nat, ('a, 'more)\ Ring\text{-}scheme, 'a] \Rightarrow 'a$
$(\langle(3\ \text{-}\ \times_{\text{-}}\ \text{-})\rangle\ [75,75,76]75)$ **where**
$n \times_R x == nscal\ R\ x\ n$

abbreviation
$NPOW :: ['a, ('a, 'more)\ Ring\text{-}scheme, nat] \Rightarrow 'a$
$(\langle(3\text{-}^{\frown_R}\ \text{-})\rangle\ [77,77,78]77)$ **where**
$a^{\frown R}\ n == npow\ R\ a\ n$

abbreviation
$SUM :: ('a, 'more)\ aGroup\text{-}scheme \Rightarrow (nat \Rightarrow 'a) \Rightarrow nat \Rightarrow 'a$
$(\langle(3\Sigma_e\ \text{-}\ \text{-}\ \text{-})\rangle\ [85,85,86]85)$ **where**
$\Sigma_e\ G\ f\ n == nsum\ G\ f\ n$

abbreviation
$NPROD :: [('a, 'm)\ Ring\text{-}scheme, nat, nat \Rightarrow 'a] \Rightarrow 'a$
$(\langle(3e\Pi_{\text{-},\text{-}}\ \text{-})\rangle\ [98,98,99]98)$ **where**
$e\Pi_{R,n}\ f == nprod\ R\ f\ n$

definition
$fSum :: [\text{-}, (nat \Rightarrow 'a), nat, nat] \Rightarrow 'a$ **where**
$fSum\ A\ f\ n\ m = (if\ n \leq m\ then\ nsum\ A\ (cmp\ f\ (slide\ n))(m - n)$
$else\ \mathbf{0}_A)$

abbreviation
$FSUM :: [('a, 'more)\ aGroup\text{-}scheme, (nat \Rightarrow 'a), nat, nat] \Rightarrow 'a$
$(\langle(4\Sigma_f\ \text{-}\ \text{-}\ \text{-}\ \text{-})\rangle\ [85,85,85,86]85)$ **where**
$\Sigma_f\ G\ f\ n\ m == fSum\ G\ f\ n\ m$

lemma (**in** *aGroup*) *nsum-zeroGTr*:$(\forall j \leq n.\ f\ j = \mathbf{0}) \longrightarrow nsum\ A\ f\ n = \mathbf{0}$
⟨*proof*⟩

lemma (**in** *aGroup*) *nsum-zeroA*:$\forall j \leq n.\ f\ j = \mathbf{0} \Longrightarrow \quad nsum\ A\ f\ n = \mathbf{0}$
⟨*proof*⟩

definition
sr :: [- , *'a set*] ⇒ *bool* **where**
sr R S == $S \subseteq carrier\ R \wedge 1_{r}R \in S \wedge (\forall x{\in}S.\ \forall y \in S.\ x\ \pm_{R} (-_{a}R\ y) \in S \wedge$
$x \cdot_{r}R\ y \in S)$

definition
Sr :: [- , *'a set*] ⇒ - **where**
Sr R S = *R* ⦇*carrier* := *S*, *pop* := $\lambda x{\in}S.\ \lambda y{\in}S.\ x \pm_{R} y$, *mop* := $\lambda x{\in}S.\ (-_{a}R\ x)$,
zero := $\mathbf{0}_{R}$, *tp* := $\lambda x{\in}S.\ \lambda y{\in}S.\ x \cdot_{r}R\ y$, *un* := $1_{r}R$ ⦈

lemma (**in** *Ring*) *Ring*: *Ring R* ⟨*proof*⟩

lemma (**in** *Ring*) *ring-is-ag*:*aGroup R*
⟨*proof*⟩

lemma (**in** *Ring*) *ring-zero*:$\mathbf{0} \in carrier\ R$
⟨*proof*⟩

lemma (**in** *Ring*) *ring-one*:$1_{r} \in carrier\ R$
⟨*proof*⟩

lemma (**in** *Ring*) *ring-tOp-closed*:$[\![\ x \in carrier\ R;\ y \in carrier\ R]\!] \Longrightarrow$
$x \cdot_{r} y \in carrier\ R$
⟨*proof*⟩

lemma (**in** *Ring*) *ring-tOp-commute*:$[\![x \in carrier\ R;\ y \in carrier\ R]\!] \Longrightarrow$
$x \cdot_{r} y = y \cdot_{r} x$
⟨*proof*⟩

lemma (**in** *Ring*) *ring-distrib1*:$[\![x \in carrier\ R;\ y \in carrier\ R;\ z \in carrier\ R\ ]\!]$
$\Longrightarrow x \cdot_{r} (y \pm z) = x \cdot_{r} y \pm x \cdot_{r} z$
⟨*proof*⟩

lemma (**in** *Ring*) *ring-distrib2*:$[\![x \in carrier\ R;\ y \in carrier\ R;\ z \in carrier\ R\ ]\!]$
$\Longrightarrow (y \pm z) \cdot_{r} x = y \cdot_{r} x \pm\ \ z \cdot_{r} x$
⟨*proof*⟩

lemma (**in** *Ring*) *ring-distrib3*:$[\![a \in carrier\ R;\ b \in carrier\ R;\ x \in carrier\ R;$
$y \in carrier\ R\ ]\!] \Longrightarrow (a \pm b) \cdot_{r} (x \pm y) =$
$a \cdot_{r} x \pm a \cdot_{r} y \pm b \cdot_{r} x \pm b \cdot_{r} y$
⟨*proof*⟩

lemma (**in** *Ring*) *rEQMulR*:
$[\![x \in carrier\ R;\ y \in carrier\ R;\ z \in carrier\ R;\ x = y\ ]\!]$
$\Longrightarrow x \cdot_r z = y \cdot_r z$
⟨*proof*⟩

lemma (**in** *Ring*) *ring-tOp-assoc*:$[\![x \in carrier\ R;\ y \in carrier\ R;\ z \in carrier\ R\ ]\!]$
$\Longrightarrow (x \cdot_r y) \cdot_r z = x \cdot_r (y \cdot_r z)$
⟨*proof*⟩

lemma (**in** *Ring*) *ring-l-one*:$x \in carrier\ R \Longrightarrow 1_r \cdot_r x = x$
⟨*proof*⟩

lemma (**in** *Ring*) *ring-r-one*:$x \in carrier\ R \Longrightarrow x \cdot_r 1_r = x$
⟨*proof*⟩

lemma (**in** *Ring*) *ring-times-0-x*:$x \in carrier\ R \Longrightarrow \mathbf{0} \cdot_r x = \mathbf{0}$
⟨*proof*⟩

lemma (**in** *Ring*) *ring-times-x-0*:$x \in carrier\ R \Longrightarrow x \cdot_r \mathbf{0} = \mathbf{0}$
⟨*proof*⟩

lemma (**in** *Ring*) *rMulZeroDiv*:
$[\![\ x \in carrier\ R;\ y \in carrier\ R;\ x = \mathbf{0} \vee y = \mathbf{0}\ ]\!] \Longrightarrow x \cdot_r y = \mathbf{0}$
⟨*proof*⟩

lemma (**in** *Ring*) *ring-inv1*:$[\![\ a \in carrier\ R;\ b \in carrier\ R\ ]\!] \Longrightarrow$
$-_a (a \cdot_r b) = (-_a a) \cdot_r b \wedge -_a (a \cdot_r b) = a \cdot_r (-_a b)$
⟨*proof*⟩

lemma (**in** *Ring*) *ring-inv1-1*:$[\![a \in carrier\ R;\ b \in carrier\ R\ ]\!] \Longrightarrow$
$-_a (a \cdot_r b) = (-_a a) \cdot_r b$
⟨*proof*⟩

lemma (**in** *Ring*) *ring-inv1-2*:$[\![\ a \in carrier\ R;\ b \in carrier\ R\ ]\!] \Longrightarrow$
$-_a (a \cdot_r b) = a \cdot_r (-_a b)$
⟨*proof*⟩

lemma (**in** *Ring*) *ring-times-minusl*:$a \in carrier\ R \Longrightarrow -_a a = (-_a 1_r) \cdot_r a$
⟨*proof*⟩

lemma (**in** *Ring*) *ring-times-minusr*:$a \in carrier\ R \Longrightarrow -_a a = a \cdot_r (-_a 1_r)$
⟨*proof*⟩

lemma (**in** *Ring*) *ring-inv1-3*:$[\![a \in carrier\ R;\ b \in carrier\ R]\!] \Longrightarrow$
$a \cdot_r b = (-_a a) \cdot_r (-_a b)$
⟨*proof*⟩

lemma (**in** *Ring*) *ring-distrib4*:$[\![a \in carrier\ R;\ b \in carrier\ R;$

$x \in carrier\ R;\ y \in carrier\ R\ \rrbracket \Longrightarrow$
$a \cdot_r b \pm (-_a\ x \cdot_r y) = a \cdot_r (b \pm (-_a\ y)) \pm (a \pm (-_a\ x)) \cdot_r y$
⟨*proof*⟩

lemma (**in** *Ring*) *rMulLC*:
$\llbracket x \in carrier\ R;\ y \in carrier\ R;\ z \in carrier\ R \rrbracket$
$\Longrightarrow x \cdot_r (y \cdot_r z) = y \cdot_r (x \cdot_r z)$
⟨*proof*⟩

lemma (**in** *Ring*) *Zero-ring*:$1_r = \mathbf{0} \Longrightarrow zeroring\ R$
⟨*proof*⟩

lemma (**in** *Ring*) *Zero-ring1*:$\neg\ (zeroring\ R) \Longrightarrow\ 1_r \neq \mathbf{0}$
⟨*proof*⟩

lemma (**in** *Ring*) *Sr-one*:$sr\ R\ S \Longrightarrow 1_r \in S$
⟨*proof*⟩

lemma (**in** *Ring*) *Sr-zero*:$sr\ R\ S \Longrightarrow \mathbf{0} \in S$
⟨*proof*⟩

lemma (**in** *Ring*) *Sr-mOp-closed*:$\llbracket sr\ R\ S;\ x \in S \rrbracket \Longrightarrow -_a\ x \in S$
⟨*proof*⟩

lemma (**in** *Ring*) *Sr-pOp-closed*:$\llbracket sr\ R\ S;\ x \in S;\ y \in S \rrbracket \Longrightarrow x \pm y \in S$
⟨*proof*⟩

lemma (**in** *Ring*) *Sr-tOp-closed*:$\llbracket sr\ R\ S;\ x \in S;\ y \in S \rrbracket \Longrightarrow x \cdot_r y \in S$
⟨*proof*⟩

lemma (**in** *Ring*) *Sr-ring*:$sr\ R\ S \Longrightarrow Ring\ (Sr\ R\ S)$
⟨*proof*⟩

4.2 Calculation of elements

4.2.1 nscale

lemma (**in** *Ring*) *ring-tOp-rel*:$\llbracket x \in carrier\ R;\ xa \in carrier\ R;\ y \in carrier\ R;$
$ya \in carrier\ R\ \rrbracket \Longrightarrow (x \cdot_r xa) \cdot_r (y \cdot_r ya) = (x \cdot_r y) \cdot_r (xa \cdot_r ya)$
⟨*proof*⟩

lemma (**in** *Ring*) *nsClose*:
$\bigwedge n.\ \llbracket\ x \in carrier\ R\ \rrbracket\ \Longrightarrow nscal\ R\ x\ n \in carrier\ R$
⟨*proof*⟩

lemma (**in** *Ring*) *nsZero*:
$nscal\ R\ \mathbf{0}\ n = \mathbf{0}$
⟨*proof*⟩

lemma (**in** *Ring*) *nsZeroI*: $\bigwedge n.\ \ x = \mathbf{0} \implies nscal\ R\ x\ n = \mathbf{0}$
⟨*proof*⟩

lemma (**in** *Ring*) *nsEqElm*: $[\![\ x \in carrier\ R;\ y \in carrier\ R;\ x = y\ ]\!]$
$\implies (nscal\ R\ x\ n) = (nscal\ R\ y\ n)$
⟨*proof*⟩

lemma (**in** *Ring*) *nsDistr*: $x \in carrier\ R$
$\implies (nscal\ R\ x\ n) \pm (nscal\ R\ x\ m) = nscal\ R\ x\ (n + m)$
⟨*proof*⟩

lemma (**in** *Ring*) *nsDistrL*: $[\![x \in carrier\ R;\ y \in carrier\ R\ ]\!]$
$\implies (nscal\ R\ x\ n) \pm (nscal\ R\ y\ n) = nscal\ R\ (x \pm y)\ n$
⟨*proof*⟩

lemma (**in** *Ring*) *nsMulDistrL*:$[\![\ x \in carrier\ R;\ y \in carrier\ R\ ]\!]$
$\implies x \cdot_r (nscal\ R\ y\ n) = nscal\ R\ (x \cdot_r y)\ n$
⟨*proof*⟩

lemma (**in** *Ring*) *nsMulDistrR*:$[\![\ x \in carrier\ R;\ y \in carrier\ R]\!]$
$\implies (nscal\ R\ y\ n) \cdot_r x = nscal\ R\ (y \cdot_r x)\ n$
⟨*proof*⟩

4.2.2 npow

lemma (**in** *Ring*) *npClose*:$x \in carrier\ R \implies npow\ R\ x\ n \in carrier\ R$
⟨*proof*⟩

lemma (**in** *Ring*) *npMulDistr*:$\bigwedge n\ m.\ x \in carrier\ R\ \implies$
$(npow\ R\ x\ n) \cdot_r (npow\ R\ x\ m) = npow\ R\ x\ (n + m)$
⟨*proof*⟩

lemma (**in** *Ring*) *npMulExp*:$\bigwedge n\ m.\ x \in carrier\ R$
$\implies\ npow\ R\ (npow\ R\ x\ n)\ m = npow\ R\ x\ (n * m)$
⟨*proof*⟩

lemma (**in** *Ring*) *npGTPowZero-sub*:
$\bigwedge n.\ [\![\ x \in carrier\ R;\ npow\ R\ x\ m = \mathbf{0}\ ]\!]$
$\implies (m \leq n) \longrightarrow (npow\ R\ x\ n = \mathbf{0}\ )$
⟨*proof*⟩

lemma (**in** *Ring*) *npGTPowZero*:
$\bigwedge n.\ [\![\ x \in carrier\ R;\ npow\ R\ x\ m = \mathbf{0};\ m \leq n\ ]\!]$
$\implies npow\ R\ x\ n = \mathbf{0}$
⟨*proof*⟩

lemma (**in** *Ring*) *npOne*: $npow\ R\ (1_r)\ n = 1_r$

⟨*proof*⟩

lemma (**in** *Ring*) *npZero-sub*: $0 < n \longrightarrow npow\ R\ \mathbf{0}\ n = \mathbf{0}$
⟨*proof*⟩

lemma (**in** *Ring*) *npZero*: $0 < n \implies npow\ R\ \mathbf{0}\ n = \mathbf{0}$
⟨*proof*⟩

lemma (**in** *Ring*) *npMulElmL*: $\bigwedge n.\ [\![\ x \in carrier\ R;\ 0 \leq n]\!]$
$\implies x \cdot_r (npow\ R\ x\ n) = npow\ R\ x\ (Suc\ n)$
⟨*proof*⟩

lemma (**in** *Ring*) *npMulEleL*: $\bigwedge n.\ x \in carrier\ R$
$\implies (npow\ R\ x\ n) \cdot_r x = npow\ R\ x\ (Suc\ n)$
⟨*proof*⟩

lemma (**in** *Ring*) *npMulElmR*: $\bigwedge n.\ x \in carrier\ R$
$\implies (npow\ R\ x\ n) \cdot_r x = npow\ R\ x\ (Suc\ n)$
⟨*proof*⟩

lemma (**in** *Ring*) *np-1*:$a \in carrier\ R \implies npow\ R\ a\ (Suc\ 0) = a$
⟨*proof*⟩

4.2.3 nsum and fSum

lemma (**in** *aGroup*) *nsum-memTr*: $(\forall j \leq n.\ f\ j \in carrier\ A) \longrightarrow$
$nsum\ A\ f\ n \in carrier\ A$
⟨*proof*⟩

lemma (**in** *aGroup*) *nsum-mem*:$\forall j \leq n.\ f\ j \in carrier\ A \implies$
$nsum\ A\ f\ n \in carrier\ A$
⟨*proof*⟩

lemma (**in** *aGroup*) *nsum-eqTr*:$(\forall j \leq n.\ f\ j \in carrier\ A \wedge$
$g\ j \in carrier\ A \wedge$
$f\ j = g\ j)$
$\longrightarrow\ nsum\ A\ f\ n = nsum\ A\ g\ n$
⟨*proof*⟩

lemma (**in** *aGroup*) *nsum-eq*:$[\![\forall j \leq n.\ f\ j \in carrier\ A;\ \forall j \leq n.\ g\ j \in carrier\ A;$
$\forall j \leq n.\ f\ j = g\ j]\!] \implies\ nsum\ A\ f\ n = nsum\ A\ g\ n$
⟨*proof*⟩

lemma (**in** *aGroup*) *nsum-cmp-assoc*:$[\![\forall j \leq n.\ f\ j \in carrier\ A;$
$g \in \{j.\ j \leq n\} \to \{j.\ j \leq n\};\ h \in \{j.\ j \leq n\} \to \{j.\ j \leq n\}]\!] \implies$
$nsum\ A\ (cmp\ (cmp\ f\ h)\ g)\ n = nsum\ A\ (cmp\ f\ (cmp\ h\ g))\ n$
⟨*proof*⟩

lemma (**in** *aGroup*) *fSum-Suc*:$\forall j \in nset\ n\ (n + Suc\ m).\ f\ j \in carrier\ A \implies$

$$fSum\ A\ f\ n\ (n + Suc\ m) = fSum\ A\ f\ n\ (n + m) \pm f\ (n + Suc\ m)$$

⟨*proof*⟩

lemma (**in** *aGroup*) *fSum-eqTr*:$(\forall j \in nset\ n\ (n + m).\ f\ j \in carrier\ A \wedge$
$g\ j \in carrier\ A \wedge\ f\ j = g\ j) \longrightarrow$
$fSum\ A\ f\ \ n\ (n + m) = fSum\ A\ g\ n\ (n + m)$

⟨*proof*⟩

lemma (**in** *aGroup*) *fSum-eq*:$[\![\forall j \in nset\ n\ (n + m).\ f\ j \in carrier\ A;$
$\forall j \in nset\ n\ (n + m).\ g\ j \in carrier\ A;\ (\forall j \in nset\ n\ (n + m).\ f\ j = g\ j)]\!]$
$\Longrightarrow$
$fSum\ A\ f\ n\ (n + m) = fSum\ A\ g\ n\ (n + m)$

⟨*proof*⟩

lemma (**in** *aGroup*) *fSum-eq1*:$[\![n \leq m;\ \forall j \in nset\ n\ m.\ f\ j \in carrier\ A;$
$\forall j \in nset\ n\ m.\ g\ j \in carrier\ A;\ \forall j \in nset\ n\ m.\ f\ j = g\ j]\!] \Longrightarrow$
$fSum\ A\ f\ n\ m = fSum\ A\ g\ n\ m$

⟨*proof*⟩

lemma (**in** *aGroup*) *fSum-zeroTr*:$(\forall j \in nset\ n\ (n + m).\ f\ j = \mathbf{0}) \longrightarrow$
$fSum\ A\ f\ \ n\ (n + m) = \mathbf{0}$

⟨*proof*⟩

lemma (**in** *aGroup*) *fSum-zero*:$\forall j \in nset\ n\ (n + m).\ f\ j = \mathbf{0} \Longrightarrow$
$fSum\ A\ f\ \ n\ (n + m) = \mathbf{0}$

⟨*proof*⟩

lemma (**in** *aGroup*) *fSum-zero1*:$[\![n < m;\ \forall j \in nset\ (Suc\ n)\ m.\ f\ j = \mathbf{0}]\!] \Longrightarrow$
$fSum\ A\ f\ \ (Suc\ n)\ m = \mathbf{0}$

⟨*proof*⟩

lemma (**in** *Ring*) *nsumMulEleL*: $\bigwedge n.\ [\![\forall\ i.\ f\ i \in carrier\ R;\ x \in carrier\ R\ ]\!]$
$\Longrightarrow x \cdot_r (nsum\ R\ f\ n) = nsum\ R\ (\lambda\ i.\ x \cdot_r (f\ i))\ n$

⟨*proof*⟩

lemma (**in** *Ring*) *nsumMulElmL*:
$\bigwedge n.\ [\![\forall\ i.\ f\ i \in carrier\ R;\ x \in carrier\ R\ ]\!]$
$\Longrightarrow x \cdot_r (nsum\ R\ f\ n) = nsum\ R\ (\lambda\ i.\ x \cdot_r (f\ i))\ n$

⟨*proof*⟩

lemma (**in** *aGroup*) *nsumTailTr*:
$(\forall j \leq (Suc\ n).\ f\ j \in carrier\ A) \longrightarrow$
$nsum\ A\ f\ (Suc\ n) = (nsum\ A\ (\lambda\ i.\ (f\ (Suc\ i)))\ n) \pm (f\ 0)$

⟨*proof*⟩

lemma (**in** *aGroup*) *nsumTail*:
$\forall j \leq (Suc\ n).\ f\ j \in carrier\ A \Longrightarrow$
$nsum\ A\ f\ (Suc\ n) = (nsum\ A\ (\lambda\ i.\ (f\ (Suc\ i)))\ n) \pm (f\ 0)$

⟨*proof*⟩

lemma (**in** *aGroup*) *nsumElmTail*:
$\forall i.\ f\ i \in carrier\ A$
$\Longrightarrow nsum\ A\ f\ (Suc\ n) = (nsum\ A\ (\lambda\ i.\ (f\ (Suc\ i)))\ n) \pm (f\ 0)$
⟨*proof*⟩

lemma (**in** *aGroup*) *nsum-addTr*:
$(\forall j \leq n.\ f\ j \in carrier\ A \wedge g\ j \in carrier\ A) \longrightarrow$
$nsum\ A\ (\lambda\ i.\ (f\ i) \pm (g\ i))\ n = (nsum\ A\ f\ n) \pm (nsum\ A\ g\ n)$
⟨*proof*⟩

lemma (**in** *aGroup*) *nsum-add*:
$[\![\ \forall j \leq n.\ f\ j \in carrier\ A;\ \forall j \leq n.\ g\ j \in carrier\ A]\!] \Longrightarrow$
$nsum\ A\ (\lambda\ i.\ (f\ i) \pm (g\ i))\ n = (nsum\ A\ f\ n) \pm (nsum\ A\ g\ n)$
⟨*proof*⟩

lemma (**in** *aGroup*) *nsumElmAdd*:
$[\![\ \forall\ i.\ f\ i \in carrier\ A;\ \forall\ i.\ g\ i \in carrier\ A]\!]$
$\Longrightarrow nsum\ A\ (\lambda\ i.\ (f\ i) \pm (g\ i))\ n = (nsum\ A\ f\ n) \pm (nsum\ A\ g\ n)$
⟨*proof*⟩

lemma (**in** *aGroup*) *nsum-add-nmTr*:
$(\forall j \leq n.\ f\ j \in carrier\ A) \wedge (\forall j \leq m.\ g\ j \in carrier\ A) \longrightarrow$
$nsum\ A\ (jointfun\ n\ f\ m\ g)\ (Suc\ (n + m)) = (nsum\ A\ f\ n) \pm (nsum\ A\ g\ m)$
⟨*proof*⟩

lemma (**in** *aGroup*) *nsum-add-nm*:
$[\![\forall j \leq n.\ f\ j \in carrier\ A;\ \forall j \leq m.\ g\ j \in carrier\ A]\!] \Longrightarrow$
$nsum\ A\ (jointfun\ n\ f\ m\ g)\ (Suc\ (n + m)) = (nsum\ A\ f\ n) \pm (nsum\ A\ g\ m)$
⟨*proof*⟩

lemma (**in** *Ring*) *npeSum2-sub-muly*:
$[\![\ x \in carrier\ R;\ y \in carrier\ R\ ]\!] \Longrightarrow$
$y \cdot_r (nsum\ R\ (\lambda i.\ nscal\ R\ ((npow\ R\ x\ (n-i)) \cdot_r (npow\ R\ y\ i))$
$(n\ choose\ i))\ n)$
$= nsum\ R\ (\lambda i.\ nscal\ R\ ((npow\ R\ x\ (n-i)) \cdot_r (npow\ R\ y\ (i+1)))$
$(n\ choose\ i))\ n$
⟨*proof*⟩

lemma *binomial-n0*: $(Suc\ n\ choose\ 0) = (n\ choose\ 0)$
⟨*proof*⟩

lemma *binomial-ngt-diff*:
$(n\ choose\ Suc\ n) = (Suc\ n\ choose\ Suc\ n) - (n\ choose\ n)$
⟨*proof*⟩

lemma *binomial-ngt-0*: $(n\ choose\ Suc\ n) = 0$

⟨*proof*⟩

lemma *diffLessSuc*: $m \leq n \Longrightarrow Suc\ (n-m) = Suc\ n - m$
⟨*proof*⟩

lemma (**in** *Ring*) *npow-suc-i*:
$[\![\ x \in carrier\ R;\ i \leq n\ ]\!]$
$\Longrightarrow npow\ R\ x\ (Suc\ n - i) = x \cdot_r (npow\ R\ x\ (n-i))$
⟨*proof*⟩

lemma (**in** *Ring*) *npeSum2-sub-mulx*: $[\![\ x \in carrier\ R;\ y \in carrier\ R\ ]\!] \Longrightarrow$
$x \cdot_r (nsum\ R\ (\lambda\ i.\ nscal\ R\ ((npow\ R\ x\ (n-i)) \cdot_r (npow\ R\ y\ i))$
$(n\ choose\ i))\ n)$
$= (nsum\ R\ (\lambda i.\ nscal\ R$
$((npow\ R\ x\ (Suc\ n - Suc\ i)) \cdot_r (npow\ R\ y\ (Suc\ i)))$
$(n\ choose\ Suc\ i))\ n) \pm$
$(nscal\ R\ ((npow\ R\ x\ (Suc\ n - 0)) \cdot_r (npow\ R\ y\ 0))$
$(Suc\ n\ choose\ 0))$
⟨*proof*⟩

lemma (**in** *Ring*) *npeSum2-sub-mulx2*:
$[\![\ x \in carrier\ R;\ y \in carrier\ R\ ]\!] \Longrightarrow$
$x \cdot_r (nsum\ R\ (\lambda\ i.\ nscal\ R\ ((npow\ R\ x\ (n-i)) \cdot_r (npow\ R\ y\ i))$
$(n\ choose\ i))\ n)$
$= (nsum\ R\ \ (\lambda i.\ nscal\ R$
$((npow\ R\ x\ (n - i)) \cdot_r ((npow\ R\ y\ i) \cdot_r y\ ))$
$(n\ choose\ Suc\ i))\ n) \pm$
$(\mathbf{0} \pm ((x \cdot_r (npow\ R\ x\ n)) \cdot_r (1_r)))$
⟨*proof*⟩

lemma (**in** *Ring*) *npeSum2*:
$\bigwedge\ n.\ [\![\ x \in carrier\ R;\ y \in carrier\ R\ ]\!]$
$\Longrightarrow npow\ R\ (x \pm y)\ n =$
$nsum\ R\ (\lambda\ i.\ nscal\ R\ ((npow\ R\ x\ (n-i)) \cdot_r (npow\ R\ y\ i))$
$(\ n\ choose\ i)\ )\ n$
⟨*proof*⟩

lemma (**in** *aGroup*) *nsum-zeroTr*:
$\bigwedge\ n.\ (\forall\ i.\ i \leq n \longrightarrow\ f\ i = \mathbf{0}) \longrightarrow (nsum\ A\ f\ n = \mathbf{0})$
⟨*proof*⟩

lemma (**in** *Ring*) *npAdd*:
$[\![\ x \in carrier\ R;\ y \in carrier\ R;$
$npow\ R\ x\ m = \mathbf{0};\ npow\ R\ y\ n = \mathbf{0}\ ]\!]$
$\Longrightarrow npow\ R\ (x \pm y)\ (m + n) = \mathbf{0}$
⟨*proof*⟩

lemma (**in** *Ring*) *npInverse*:
$\bigwedge n.\ x \in carrier\ R$
$\quad \Longrightarrow npow\ R\ (-_a\ x)\ n = npow\ R\ x\ n$
$\quad\quad \vee\ npow\ R\ (-_a\ x)\ n = -_a\ (npow\ R\ x\ n)$
⟨*proof*⟩

lemma (**in** *Ring*) *npMul*:
$\bigwedge\ n.\ [\![\ x \in carrier\ R;\ y \in carrier\ R\ ]\!]$
$\quad \Longrightarrow npow\ R\ (x \cdot_r y)\ n = (npow\ R\ x\ n) \cdot_r (npow\ R\ y\ n)$
⟨*proof*⟩

4.3 Ring homomorphisms

definition
rHom :: [('*a*, '*m*) *Ring-scheme*, ('*b*, '*m1*) *Ring-scheme*]
$\quad \Rightarrow$ ('*a* $\Rightarrow$ '*b*) *set* **where**
rHom A R = {*f*. *f* ∈ *aHom A R* ∧
(∀ *x*∈*carrier A*. ∀ *y*∈*carrier A*. *f* (*x* $\cdot_{r\,A}$ *y*) = (*f x*) $\cdot_{r\,R}$ (*f y*))
∧ *f* (*1*$_{r\,A}$) = (*1*$_{r\,R}$)}

definition
rInvim :: [('*a*, '*m*) *Ring-scheme*, ('*b*, '*m1*) *Ring-scheme*, '*a* $\Rightarrow$ '*b*, '*b set*]
$\quad \Rightarrow$ '*a set* **where**
rInvim A R f K = {*a*. *a* ∈ *carrier A* ∧ *f a* ∈ *K*}

definition
rimg :: [('*a*, '*m*) *Ring-scheme*, ('*b*, '*m1*) *Ring-scheme*, '*a* $\Rightarrow$ '*b*] $\Rightarrow$
'*b Ring* **where**
rimg A R f = (|*carrier*= *f* ‘(*carrier A*), *pop* = *pop R*, *mop* = *mop R*,
zero = *zero R*, *tp* = *tp R*, *un* = *un R* |)

definition
ridmap :: ('*a*, '*m*) *Ring-scheme* $\Rightarrow$ ('*a* $\Rightarrow$ '*a*) **where**
ridmap R = (λ*x*∈*carrier R*. *x*)

definition
r-isom :: [('*a*, '*m*) *Ring-scheme*, ('*b*, '*m1*) *Ring-scheme*] $\Rightarrow$ *bool*
(**infixr** ‹≅$_r$› *100*) **where**
r-isom R R' ⟷ (∃ *f*∈*rHom R R'*. *bijec*$_{R,R'}$ *f*)

definition
Subring :: [('*a*, '*m*) *Ring-scheme*, ('*a*, '*m1*) *Ring-scheme*] $\Rightarrow$ *bool* **where**
Subring R S == *Ring S* ∧ (*carrier S* ⊆ *carrier R*) ∧ (*ridmap S*) ∈ *rHom S R*

lemma *ridmap-surjec*:*Ring A* $\Longrightarrow$ *surjec*$_{A,A}$ (*ridmap A*)
⟨*proof*⟩

lemma *rHom-aHom*:*f* ∈ *rHom A R* $\Longrightarrow$ *f* ∈ *aHom A R*

⟨*proof*⟩

lemma *rimg-carrier*:$f \in rHom\ A\ R \Longrightarrow carrier\ (rimg\ A\ R\ f) = f\ `\ (carrier\ A)$
⟨*proof*⟩

lemma *rHom-mem*:$[\![\ f \in rHom\ A\ R;\ a \in carrier\ A\ ]\!] \Longrightarrow f\ a \in carrier\ R$
⟨*proof*⟩

lemma *rHom-func*:$f \in rHom\ A\ R \Longrightarrow f \in carrier\ A \to carrier\ R$
⟨*proof*⟩

lemma *ringhom1*:$[\![\ Ring\ A;\ Ring\ R;\ x \in carrier\ A;\ y \in carrier\ A;$
$f \in rHom\ A\ R\ ]\!] \Longrightarrow f\ (x \pm_A\ y) = (f\ x) \pm_R\ (f\ y)$
⟨*proof*⟩

lemma *rHom-inv-inv*:$[\![\ Ring\ A;\ Ring\ R;\ x \in carrier\ A;\ f \in rHom\ A\ R\ ]\!]$
$\Longrightarrow f\ (-_{a}A\ x) = -_{a}R\ (f\ x)$
⟨*proof*⟩

lemma *rHom-0-0*:$[\![\ Ring\ A;\ Ring\ R;\ f \in rHom\ A\ R\ ]\!] \Longrightarrow f\ (\mathbf{0}_A) = \mathbf{0}_R$
⟨*proof*⟩

lemma *rHom-tOp*:$[\![\ Ring\ A;\ Ring\ R;\ x \in carrier\ A;\ y \in carrier\ A;$
$f \in rHom\ A\ R\ ]\!] \Longrightarrow f\ (x \cdot_{r}A\ y) = (f\ x) \cdot_{r}R\ (f\ y)$
⟨*proof*⟩

lemma *rHom-add*:$[\![f \in rHom\ A\ R;\ x \in carrier\ A;\ y \in carrier\ A]\!] \Longrightarrow$
$f\ (x \pm_A\ y) = (f\ x) \pm_R\ (f\ y)$
⟨*proof*⟩

lemma *rHom-one*:$[\![\ Ring\ A;\ Ring\ R;f \in rHom\ A\ R\ ]\!] \Longrightarrow f\ (1_{r}A) = (1_{r}R)$
⟨*proof*⟩

lemma *rHom-npow*:$[\![\ Ring\ A;\ Ring\ R;\ x \in carrier\ A;\ f \in rHom\ A\ R\ ]\!] \Longrightarrow$
$f\ (x^{\frown}A\ n) = (f\ x)^{\frown}R\ n$
⟨*proof*⟩

lemma *rHom-compos*:$[\![Ring\ A;\ Ring\ B;\ Ring\ C;\ f \in rHom\ A\ B;\ g \in rHom\ B\ C]\!]$
$\Longrightarrow$
$compos\ A\ g\ f \in rHom\ A\ C$
⟨*proof*⟩

lemma *rimg-ag*:$[\![Ring\ A;\ Ring\ R;\ f \in rHom\ A\ R]\!] \Longrightarrow aGroup\ (rimg\ A\ R\ f)$
⟨*proof*⟩

lemma *rimg-ring*:$[\![Ring\ A;\ Ring\ R;\ f \in rHom\ A\ R\ ]\!] \Longrightarrow Ring\ (rimg\ A\ R\ f)$
⟨*proof*⟩

definition

ideal :: [- , *'a set*] ⇒ *bool* **where**
ideal R I ⟷ (*R* +> *I*) ∧ (∀ *r*∈*carrier R*. ∀ *x*∈*I*. (*r* ·$_{r\,R}$ *x* ∈ *I*))

lemma (**in** *Ring*) *ideal-asubg*:*ideal R I* ⟹ *R* +> *I*
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-pOp-closed*:⟦*ideal R I*; *x* ∈ *I*; *y* ∈ *I* ⟧
⟹ *x* ± *y* ∈ *I*
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-nsum-closedTr*:*ideal R I* ⟹
(∀ *j* ≤ *n*. *f j* ∈ *I*) ⟶ *nsum R f n* ∈ *I*
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-nsum-closed*:⟦*ideal R I*; ∀ *j* ≤ *n*. *f j* ∈ *I*⟧ ⟹
nsum R f n ∈ *I*
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-subset1*:*ideal R I* ⟹ *I* ⊆ *carrier R*
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-subset*:⟦*ideal R I*; *h* ∈ *I*⟧ ⟹ *h* ∈ *carrier R*
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-ring-multiple*:⟦*ideal R I*; *x* ∈ *I*; *r* ∈ *carrier R*⟧ ⟹
r ·$_r$ *x* ∈ *I*
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-ring-multiple1*:⟦*ideal R I*; *x* ∈ *I*; *r* ∈ *carrier R* ⟧ ⟹
x ·$_r$ *r* ∈ *I*
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-npow-closedTr*:⟦*ideal R I*; *x* ∈ *I*⟧ ⟹
0 < *n* ⟶ *x*^^{R} *n* ∈ *I*
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-npow-closed*:⟦*ideal R I*; *x* ∈ *I*; *0* < *n*⟧ ⟹ *x*^^{R} *n* ∈ *I*
⟨*proof*⟩

lemma (**in** *Ring*) *times-modTr*:⟦*a* ∈ *carrier R*; *a'* ∈ *carrier R*; *b* ∈ *carrier R*;
b' ∈ *carrier R*; *ideal R I*; *a* ± (−$_a$ *b*) ∈ *I*; *a'* ± (−$_a$ *b'*) ∈ *I*⟧ ⟹
a ·$_r$ *a'* ± (−$_a$ (*b* ·$_r$ *b'*)) ∈ *I*
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-inv1-closed*:⟦ *ideal R I*; *x* ∈ *I* ⟧ ⟹ −$_a$ *x* ∈ *I*
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-zero*:*ideal R I* ⟹ **0** ∈ *I*

⟨*proof*⟩

lemma (**in** *Ring*) *ideal-zero-forall*:$\forall\, I.\ ideal\ R\ I \longrightarrow\ \mathbf{0} \in I$
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-ele-sumTr1*:$[\![\ ideal\ R\ I;\ a \in carrier\ R;\ b \in carrier\ R;$
$a \pm b \in I;\ a \in I\ ]\!] \Longrightarrow b \in I$
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-ele-sumTr2*:$[\![ideal\ R\ I;\ a \in carrier\ R;\ b \in carrier\ R;$
$a \pm b \in I;\ b \in I]\!] \Longrightarrow a \in I$
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-condition*:$[\![I \subseteq carrier\ R;\ I \neq \{\};$
$\forall\, x{\in}I.\ \forall\, y{\in}I.\ x \pm (-_a\ y) \in I;\ \forall\, r{\in}carrier\ R.\ \forall\, x{\in}I.\ r \cdot_r x \in I\ ]\!] \Longrightarrow$
$ideal\ R\ I$
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-condition1*:$[\![I \subseteq carrier\ R;\ I \neq \{\};$
$\forall\, x{\in}I.\ \forall\, y{\in}I.\ x \pm y \in I;\ \forall\, r{\in}carrier\ R.\ \forall\, x{\in}I.\ r \cdot_r x \in I\ ]\!] \Longrightarrow ideal\ R\ I$
⟨*proof*⟩

lemma (**in** *Ring*) *zero-ideal*:$ideal\ R\ \{\mathbf{0}\}$
⟨*proof*⟩

lemma (**in** *Ring*) *whole-ideal*:$ideal\ R\ (carrier\ R)$
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-inc-one*:$[\![ideal\ R\ I;\ 1_r \in I\ ]\!] \Longrightarrow I = carrier\ R$
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-inc-one1*:$ideal\ R\ I \Longrightarrow$
$(1_r \in I) = (I = carrier\ R)$
⟨*proof*⟩

definition
Unit :: - ⇒ *'a* ⇒ *bool* **where**
Unit R a ⟷ $a \in carrier\ R \wedge (\exists\, b{\in}carrier\ R.\ a \cdot_{r\,R}\ b = 1_{r\,R})$

lemma (**in** *Ring*) *ideal-inc-unit*:$[\![ideal\ R\ I;\ a \in I;\ Unit\ R\ a]\!] \Longrightarrow 1_r \in I$
⟨*proof*⟩

lemma (**in** *Ring*) *proper-ideal*:$[\![ideal\ R\ I;\ 1_r \notin I]\!] \Longrightarrow I \neq carrier\ R$
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-inc-unit1*:$[\![a \in carrier\ R;\ Unit\ R\ a;\ ideal\ R\ I;\ a \in I]\!]$
$\Longrightarrow I = carrier\ R$
⟨*proof*⟩

lemma (**in** *Ring*) *int-ideal*:⟦*ideal R I*; *ideal R J*⟧ ⟹ *ideal R* (*I* ∩ *J*)
⟨*proof*⟩

definition
ideal-prod::[-, *'a set*, *'a set*] ⇒ *'a set* (**infix** ‹⋄$_{r1}$› *90*) **where**
ideal-prod R I J == ⋂ {*L*. *ideal R L* ∧
{*x*.(∃ *i*∈*I*. ∃ *j*∈*J*. *x* = *i* ·$_{r\,R}$ *j*)} ⊆ *L*}

lemma (**in** *Ring*) *set-sum-mem*:⟦*a* ∈ *I*; *b* ∈ *J*; *I* ⊆ *carrier R*; *J* ⊆ *carrier R*⟧ ⟹
a ± *b* ∈ *I* ∓ *J*
⟨*proof*⟩

lemma (**in** *Ring*) *sum-ideals*:⟦*ideal R I1*; *ideal R I2*⟧ ⟹ *ideal R* (*I1* ∓ *I2*)
⟨*proof*⟩

lemma (**in** *Ring*) *sum-ideals-la1*:⟦*ideal R I1*; *ideal R I2*⟧ ⟹ *I1* ⊆ (*I1* ∓ *I2*)
⟨*proof*⟩

lemma (**in** *Ring*) *sum-ideals-la2*:⟦*ideal R I1*; *ideal R I2* ⟧ ⟹ *I2* ⊆ (*I1* ∓ *I2*)
⟨*proof*⟩

lemma (**in** *Ring*) *sum-ideals-cont*:⟦*ideal R I*; *A* ⊆ *I*; *B* ⊆ *I* ⟧ ⟹ *A* ∓ *B* ⊆ *I*
⟨*proof*⟩

lemma (**in** *Ring*) *ideals-set-sum*:⟦*ideal R A*; *ideal R B*; *x* ∈ *A* ∓ *B*⟧ ⟹
∃ *h*∈*A*. ∃ *k*∈*B*. *x* = *h* ± *k*
⟨*proof*⟩

definition
Rxa :: [-, *'a*] ⇒ *'a set* (**infixl** ‹⋄$_p$› *200*) **where**
Rxa R a = {*x*. ∃ *r*∈*carrier R*. *x* = (*r* ·$_{r\,R}$ *a*)}

lemma (**in** *Ring*) *a-in-principal*:*a* ∈ *carrier R* ⟹ *a* ∈ *Rxa R a*
⟨*proof*⟩

lemma (**in** *Ring*) *principal-ideal*:*a* ∈ *carrier R* ⟹ *ideal R* (*Rxa R a*)
⟨*proof*⟩

lemma (**in** *Ring*) *rxa-in-Rxa*:⟦*a* ∈ *carrier R*; *r* ∈ *carrier R*⟧ ⟹
r ·$_r$ *a* ∈ *Rxa R a*
⟨*proof*⟩

lemma (**in** *Ring*) *Rxa-one*:*Rxa R* 1$_r$ = *carrier R*
⟨*proof*⟩

lemma (**in** *Ring*) *Rxa-zero*:*Rxa R* **0** = {**0**}
⟨*proof*⟩

lemma (**in** *Ring*) *Rxa-nonzero*:$[\![a \in carrier\ R;\ a \neq \mathbf{0}]\!] \Longrightarrow Rxa\ R\ a \neq \{\mathbf{0}\}$
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-cont-Rxa*:$[\![ideal\ R\ I;\ a \in I]\!] \Longrightarrow Rxa\ R\ a \subseteq I$
⟨*proof*⟩

lemma (**in** *Ring*) *Rxa-mult-smaller*:$[\![\ a \in carrier\ R;\ b \in carrier\ R]\!] \Longrightarrow$
$Rxa\ R\ (a \cdot_r b) \subseteq Rxa\ R\ b$
⟨*proof*⟩

lemma (**in** *Ring*) *id-ideal-psub-sum*:$[\![ideal\ R\ I;\ a \in carrier\ R;\ a \notin I]\!] \Longrightarrow$
$I \subset I \mp Rxa\ R\ a$
⟨*proof*⟩

lemma (**in** *Ring*) *mul-two-principal-idealsTr*:$[\![a \in carrier\ R;\ b \in carrier\ R;$
$x \in Rxa\ R\ a;\ y \in Rxa\ R\ b]\!] \Longrightarrow \exists\, r{\in}carrier\ R.\ x \cdot_r y = r \cdot_r (a \cdot_r b)$
⟨*proof*⟩

primrec *sum-pr-ideals*::$[('a, 'm)\ Ring\text{-}scheme, nat \Rightarrow 'a, nat] \Rightarrow 'a\ set$
where
sum-pr0: $sum\text{-}pr\text{-}ideals\ R\ f\ 0 = Rxa\ R\ (f\ 0)$
| *sum-prn*: $sum\text{-}pr\text{-}ideals\ R\ f\ (Suc\ n) =$
$(Rxa\ R\ (f\ (Suc\ n))) \mp_R (sum\text{-}pr\text{-}ideals\ R\ f\ n)$

lemma (**in** *Ring*) *sum-of-prideals0*:
$\forall f.\ (\forall\, l \leq n.\ f\ l \in carrier\ R) \longrightarrow ideal\ R\ (sum\text{-}pr\text{-}ideals\ R\ f\ n)$
⟨*proof*⟩

lemma (**in** *Ring*) *sum-of-prideals*:$[\![\forall\, l \leq n.\ f\ l \in carrier\ R]\!] \Longrightarrow$
$ideal\ R\ (sum\text{-}pr\text{-}ideals\ R\ f\ n)$
⟨*proof*⟩

later, we show *sum-pr-ideals* is the least ideal containing $\{f\ 0, f\ 1, \ldots, f\ n\}$

lemma (**in** *Ring*) *sum-of-prideals1*:$\forall f.\ (\forall\, l \leq n.\ f\ l \in carrier\ R) \longrightarrow$
$f\ `\ \{i.\ i \leq n\} \subseteq (sum\text{-}pr\text{-}ideals\ R\ f\ n)$
⟨*proof*⟩

lemma (**in** *Ring*) *sum-of-prideals2*:$\forall\, l \leq n.\ f\ l \in carrier\ R$
$\Longrightarrow\ f\ `\ \{i.\ i \leq n\} \subseteq (sum\text{-}pr\text{-}ideals\ R\ f\ n)$
⟨*proof*⟩

lemma (**in** *Ring*) *sum-of-prideals3*:$ideal\ R\ I \Longrightarrow$
$\forall f.\ (\forall\, l \leq n.\ f\ l \in carrier\ R) \wedge (f\ `\ \{i.\ i \leq n\} \subseteq I) \longrightarrow$
$(sum\text{-}pr\text{-}ideals\ R\ f\ n \subseteq I)$
⟨*proof*⟩

lemma (**in** *Ring*) *sum-of-prideals4*:$[\![ideal\ R\ I;\ \forall\, l \leq n.\ f\ l \in carrier\ R;$

$(f\ `\ \{i.\ i \leq n\} \subseteq I)\rrbracket \Longrightarrow sum\text{-}pr\text{-}ideals\ R\ f\ n \subseteq I$
⟨*proof*⟩

lemma *ker-ideal*:⟦*Ring A*; *Ring R*; $f \in rHom\ A\ R\rrbracket \Longrightarrow ideal\ A\ (ker_{A,R}\ f)$
⟨*proof*⟩

4.3.1 Ring of integers

definition
Zr :: *int Ring* **where**
Zr = (| *carrier* = *Zset*, *pop* = λ*n*∈*Zset*. λ*m*∈*Zset*. (*m* + *n*),
mop = λ*l*∈*Zset*. −*l*, *zero* = *0*, *tp* = λ*m*∈*Zset*. λ*n*∈*Zset*. *m* ∗ *n*, *un* = *1* |)

lemma *ring-of-integers*:*Ring Zr*
⟨*proof*⟩

lemma *Zr-zero*:$\mathbf{0}_{Zr} = 0$
⟨*proof*⟩

lemma *Zr-one*:$1_{r\,Zr} = 1$
⟨*proof*⟩

lemma *Zr-minus*:$-_{a\,Zr}\ n = -\ n$
⟨*proof*⟩

lemma *Zr-add*:$n \pm_{Zr} m = n + m$
⟨*proof*⟩

lemma *Zr-times*:$n \cdot_{r\,Zr} m = n * m$
⟨*proof*⟩

definition
lev :: *int set* ⇒ *int* **where**
lev I = *Zleast* {*n*. *n* ∈ *I* ∧ *0* < *n*}

lemma *Zr-gen-Zleast*:⟦*ideal Zr I*; $I \neq \{0::int\}\rrbracket \Longrightarrow$
Rxa Zr (*lev I*) = *I*
⟨*proof*⟩

lemma *Zr-pir*:*ideal Zr I* ⟹ ∃ *n*. *Rxa Zr n* = *I*
⟨*proof*⟩

4.4 Quotient rings

lemma (**in** *Ring*) *mem-set-ar-cos*:⟦*ideal R I*; *a* ∈ *carrier R*⟧ ⟹
$a \uplus_R I \in set\text{-}ar\text{-}cos\ R\ I$
⟨*proof*⟩

lemma (**in** *Ring*) *I-in-set-ar-cos*:*ideal R I* ⟹ *I* ∈ *set-ar-cos R I*

⟨*proof*⟩

lemma (**in** *Ring*) *ar-coset-same1*:⟦*ideal R I*; $a \in$ *carrier R*; $b \in$ *carrier R*; $b \pm (-_a\ a) \in I$ ⟧ $\Longrightarrow a \uplus_R I = b \uplus_R I$
⟨*proof*⟩

lemma (**in** *Ring*) *ar-coset-same2*:⟦*ideal R I*; $a \in$ *carrier R*; $b \in$ *carrier R*; $a \uplus_R I = b \uplus_R I$⟧ $\Longrightarrow$ $b \pm (-_a\ a) \in I$
⟨*proof*⟩

lemma (**in** *Ring*) *ar-coset-same3*:⟦*ideal R I*; $a \in$ *carrier R*; $a \uplus_R I = I$⟧ $\Longrightarrow$ $a \in I$
⟨*proof*⟩

lemma (**in** *Ring*) *ar-coset-same3-1*:⟦*ideal R I*; $a \in$ *carrier R*; $a \notin I$⟧ $\Longrightarrow$ $a \uplus_R I \neq I$
⟨*proof*⟩

lemma (**in** *Ring*) *ar-coset-same4*:⟦*ideal R I*; $a \in I$⟧ $\Longrightarrow$ $a \uplus_R I = I$
⟨*proof*⟩

lemma (**in** *Ring*) *ar-coset-same4-1*:⟦*ideal R I*; $a \uplus_R I \neq I$⟧ $\Longrightarrow a \notin I$
⟨*proof*⟩

lemma (**in** *Ring*) *belong-ar-coset1*:⟦*ideal R I*; $a \in$ *carrier R*; $x \in$ *carrier R*; $x \pm (-_a\ a) \in I$⟧ $\Longrightarrow$ $x \in a \uplus_R I$
⟨*proof*⟩

lemma (**in** *Ring*) *a-in-ar-coset*:⟦*ideal R I*; $a \in$ *carrier R*⟧ $\Longrightarrow a \in a \uplus_R I$
⟨*proof*⟩

lemma (**in** *Ring*) *ar-coset-subsetD*:⟦*ideal R I*; $a \in$ *carrier R*; $x \in a \uplus_R I$ ⟧ $\Longrightarrow$ $x \in$ *carrier R*
⟨*proof*⟩

lemma (**in** *Ring*) *ar-cos-mem*:⟦*ideal R I*; $a \in$ *carrier R*⟧ $\Longrightarrow$ $a \uplus_R I \in$ *set-rcs* (*b-ag R*) *I*
⟨*proof*⟩

lemma (**in** *Ring*) *mem-ar-coset1*:⟦*ideal R I*; $a \in$ *carrier R*; $x \in a \uplus_R I$⟧ $\Longrightarrow$ $\exists h{\in}I.\ h \pm a = x$
⟨*proof*⟩

lemma (**in** *Ring*) *ar-coset-mem2*:⟦*ideal R I*; $a \in$ *carrier R*; $x \in a \uplus_R I$⟧ $\Longrightarrow$ $\exists h{\in}I.\ x = a \pm h$
⟨*proof*⟩

lemma (**in** *Ring*) *belong-ar-coset2*:⟦*ideal R I*; $a \in$ *carrier R*; $x \in a \uplus_R I$ ⟧

$$\Longrightarrow x \pm (-_a\ a) \in I$$

⟨*proof*⟩

lemma (**in** *Ring*) *ar-c-top*: ⟦*ideal R I*; *a* ∈ *carrier R*; *b* ∈ *carrier R*⟧
$\Longrightarrow$ (*c-top* (*b-ag R*) *I* (*a* $\uplus_R$ *I*) (*b* $\uplus_R$ *I*)) = (*a* ± *b*) $\uplus_R$ *I*
⟨*proof*⟩

Following lemma is not necessary to define a quotient ring. But it makes clear that the binary operation2 of the quotient ring is well defined.

lemma (**in** *Ring*) *quotient-ring-tr1*:⟦*ideal R I*; *a1* ∈ *carrier R*; *a2* ∈ *carrier R*;
b1 ∈ *carrier R*; *b2* ∈ *carrier R*;
a1 $\uplus_R$ *I* = *a2* $\uplus_R$ *I*; *b1* $\uplus_R$ *I* = *b2* $\uplus_R$ *I*⟧ $\Longrightarrow$
(*a1* $\cdot_r$ *b1*) $\uplus_R$ *I* = (*a2* $\cdot_r$ *b2*) $\uplus_R$ *I*
⟨*proof*⟩

definition
rcostOp :: [-, *'a set*] ⇒ (['*a set*, *'a set*] ⇒ *'a set*) **where**
rcostOp R I = (λ*X*∈(*set-rcs* (*b-ag R*) *I*). λ*Y*∈(*set-rcs* (*b-ag R*) *I*).
{*z*. ∃ *x* ∈ *X*. ∃ *y* ∈ *Y*. ∃*h*∈*I*. (*x* $\cdot_{r R}$ *y*) $\pm_R$ *h* = *z*})

lemma (**in** *Ring*) *rcostOp*:⟦*ideal R I*; *a* ∈ *carrier R*; *b* ∈ *carrier R*⟧ $\Longrightarrow$
rcostOp R I (*a* $\uplus_R$ *I*) (*b* $\uplus_R$ *I*) = (*a* $\cdot_r$ *b*) $\uplus_R$ *I*
⟨*proof*⟩

definition
qring :: [('*a*, *'m*) *Ring-scheme*, *'a set*] ⇒ (| *carrier* :: *'a set set*,
pop :: ['*a set*, *'a set*] ⇒ *'a set*, *mop* :: *'a set* ⇒ *'a set*,
zero :: *'a set*, *tp* :: ['*a set*, *'a set*] ⇒ *'a set*, *un* :: *'a set* |) **where**
qring R I = (| *carrier* = *set-rcs* (*b-ag R*) *I*,
pop = *c-top* (*b-ag R*) *I*,
mop = *c-iop* (*b-ag R*) *I*,
zero = *I*,
tp = *rcostOp R I*,
un = $1_{r R}$ $\uplus_R$ *I*|)

abbreviation
QRING (**infixl** ‹'/$_r$› *200*) **where**
R /$_r$ *I* == *qring R I*

lemma (**in** *Ring*) *carrier-qring*:*ideal R I* $\Longrightarrow$
carrier (*qring R I*) = *set-rcs* (*b-ag R*) *I*
⟨*proof*⟩

lemma (**in** *Ring*) *carrier-qring1*:*ideal R I* $\Longrightarrow$
carrier (*qring R I*) = *set-ar-cos R I*
⟨*proof*⟩

lemma (**in** *Ring*) *qring-ring*:*ideal R I* $\Longrightarrow$ *Ring* (*qring R I*)
⟨*proof*⟩

lemma (**in** *Ring*) *qring-carrier*:*ideal R I* $\Longrightarrow$
carrier (*qring R I*) $= \{X.\ \exists\, a \in$ *carrier R*. $a \uplus_R I = X\}$
⟨*proof*⟩

lemma (**in** *Ring*) *qring-mem*:$[\![$*ideal R I*; $a \in$ *carrier R*$]\!] \Longrightarrow$
$a \uplus_R I \in$ *carrier* (*qring R I*)
⟨*proof*⟩

lemma (**in** *Ring*) *qring-pOp*:$[\![$*ideal R I*; $a \in$ *carrier R*; $b \in$ *carrier R* $]\!]$
$\Longrightarrow$ *pop* (*qring R I*) ($a \uplus_R I$) ($b \uplus_R I$) $= (a \pm b) \uplus_R I$
⟨*proof*⟩

lemma (**in** *Ring*) *qring-zero*:*ideal R I* $\Longrightarrow$ *zero* (*qring R I*) $= I$
⟨*proof*⟩

lemma (**in** *Ring*) *qring-zero-1*:$[\![a \in$ *carrier R*; *ideal R I*; $a \uplus_R I = I]\!] \Longrightarrow$
$a \in I$
⟨*proof*⟩

lemma (**in** *Ring*) *Qring-fix1*:$[\![a \in$ *carrier R*; *ideal R I*; $a \in I]\!] \Longrightarrow a \uplus_R I = I$
⟨*proof*⟩

lemma (**in** *Ring*) *ar-cos-same*:$[\![a \in$ *carrier R*; *ideal R I*; $x \in a \uplus_R I]\!] \Longrightarrow$
$x \uplus_R I = a \uplus_R I$
⟨*proof*⟩

lemma (**in** *Ring*) *qring-tOp*:$[\![$*ideal R I*; $a \in$ *carrier R*; $b \in$ *carrier R*$]\!] \Longrightarrow$
tp (*qring R I*) ($a \uplus_R I$) ($b \uplus_R I$) $= (a \cdot_r b) \uplus_R I$
⟨*proof*⟩

lemma *rind-hom-well-def*:$[\![$*Ring A*; *Ring R*; $f \in$ *rHom A R*; $a \in$ *carrier A* $]\!] \Longrightarrow$
$f\ a = (f^\circ{}_{A,R})\ (a \uplus_A (ker_{A,R}\ f))$
⟨*proof*⟩

lemma (**in** *Ring*) *set-r-ar-cos*:*ideal R I* $\Longrightarrow$
set-rcs (*b-ag R*) $I =$ *set-ar-cos R I*
⟨*proof*⟩

lemma *set-r-ar-cos-ker*:$[\![$*Ring A*; *Ring R*; $f \in$ *rHom A R* $]\!] \Longrightarrow$
set-rcs (*b-ag A*) ($ker_{A,R}\ f$) $=$ *set-ar-cos A* ($ker_{A,R}\ f$)
⟨*proof*⟩

lemma *ind-hom-rhom*:$[\![$*Ring A*; *Ring R*; $f \in$ *rHom A R*$]\!] \Longrightarrow$
$(f^\circ{}_{A,R}) \in$ *rHom* (*qring A* ($ker_{A,R}\ f$)) *R*
⟨*proof*⟩

lemma *ind-hom-injec*:$[\![$*Ring A*; *Ring R*; $f \in$ *rHom A R*$]\!] \Longrightarrow$
$inj_{ec(qring\ A\ (ker_{A,R}\ f)),R}\ (f^\circ{}_{A,R})$

⟨*proof*⟩

lemma *rhom-to-rimg*:⟦*Ring A*; *Ring R*; $f \in rHom\ A\ R$⟧ ⟹
$f \in rHom\ A\ (rimg\ A\ R\ f)$
⟨*proof*⟩

lemma *ker-to-rimg*:⟦*Ring A*; *Ring R*; $f \in rHom\ A\ R$ ⟧ ⟹
$ker_{A,R}\ f = ker_{A,(rimg\ A\ R\ f)}\ f$
⟨*proof*⟩

lemma *indhom-eq*:⟦*Ring A*; *Ring R*; $f \in rHom\ A\ R$⟧ ⟹ $f^\circ{}_{A,(rimg\ A\ R\ f)} = f^\circ{}_{A,R}$
⟨*proof*⟩

lemma *indhom-bijec2-rimg*:⟦*Ring A*; *Ring R*; $f \in rHom\ A\ R$⟧ ⟹
$bijec_{(qring\ A\ (ker_{A,R}\ f)),(rimg\ A\ R\ f)}\ (f^\circ{}_{A,R})$
⟨*proof*⟩

lemma *surjec-ind-bijec*:⟦*Ring A*; *Ring R*; $f \in rHom\ A\ R$; $surjec_{A,R}\ f$⟧ ⟹
$bijec_{(qring\ A\ (ker_{A,R}\ f)),R}\ (f^\circ{}_{A,R})$
⟨*proof*⟩

lemma *ridmap-ind-bijec*:*Ring A* ⟹
$bijec_{(qring\ A\ (ker_{A,A}\ (ridmap\ A))),A}\ ((ridmap\ A)^\circ{}_{A,A})$
⟨*proof*⟩

lemma *ker-of-idmap*:*Ring A* ⟹ $ker_{A,A}\ (ridmap\ A) = \{\mathbf{0}_A\}$
⟨*proof*⟩

lemma *ring-natural-isom*:*Ring A* ⟹
$bijec_{(qring\ A\ \{\mathbf{0}_A\}),A}\ ((ridmap\ A)^\circ{}_{A,A})$
⟨*proof*⟩

definition
pj :: [('*a*, '*m*) *Ring-scheme*, '*a set*] ⇒ ('*a* => '*a set*) **where**
pj R I = (λ*x*. *Pj* (*b-ag R*) *I x*)

lemma *pj-Hom*:⟦*Ring R*; *ideal R I*⟧ ⟹ $(pj\ R\ I) \in rHom\ R\ (qring\ R\ I)$
⟨*proof*⟩

lemma *pj-mem*:⟦*Ring R*; *ideal R I*; $x \in carrier\ R$⟧ ⟹ $pj\ R\ I\ x = x \uplus_R I$
⟨*proof*⟩

lemma *pj-zero*:⟦*Ring R*; *ideal R I*; $x \in carrier\ R$⟧ ⟹
$(pj\ R\ I\ x = \mathbf{0}_{(R\ /_r\ I)}) = (x \in I)$
⟨*proof*⟩

lemma *pj-surj-to*:⟦*Ring R*; *ideal R J*; $X \in$ *carrier* $(R /_r J)$⟧ ⟹
$\exists\, r \in$ *carrier R*. *pj R J r* $= X$
⟨*proof*⟩

lemma *invim-of-ideal*:⟦*Ring R*; *ideal R I*; *ideal* (*qring R I*) *J* ⟧ ⟹
ideal R (*rInvim R* (*qring R I*) (*pj R I*) *J*)
⟨*proof*⟩

lemma *pj-invim-cont-I*:⟦*Ring R*; *ideal R I*; *ideal* (*qring R I*) *J*⟧ ⟹
I ⊆ (*rInvim R* (*qring R I*) (*pj R I*) *J*)
⟨*proof*⟩

lemma *pj-invim-mono1*:⟦*Ring R*; *ideal R I*; *ideal* (*qring R I*) *J1*;
ideal (*qring R I*) *J2*; *J1* ⊆ *J2* ⟧ ⟹
(*rInvim R* (*qring R I*) (*pj R I*) *J1*) ⊆ (*rInvim R* (*qring R I*) (*pj R I*) *J2*)
⟨*proof*⟩

lemma *pj-img-ideal*:⟦*Ring R*; *ideal R I*; *ideal R J*; *I* ⊆ *J*⟧ ⟹
ideal (*qring R I*) ((*pj R I*)‘*J*)
⟨*proof*⟩

lemma *npQring*:⟦*Ring R*; *ideal R I*; *a* ∈ *carrier R*⟧ ⟹
npow (*qring R I*) (*a* $\uplus_R$ *I*) *n* = (*npow R a n*) $\uplus_R$ *I*
⟨*proof*⟩

4.5 Primary ideals, Prime ideals

definition
maximal-set :: [*'a set set*, *'a set*] ⇒ *bool* **where**
maximal-set S mx ⟷ *mx* ∈ *S* ∧ (∀ *s*∈*S*. *mx* ⊆ *s* ⟶ *s* = *mx*)

definition
nilpotent :: [-, *'a*] ⇒ *bool* **where**
nilpotent R a ⟷ (∃ (*n*::*nat*). $a^{\frown_R}\, ^n = \mathbf{0}_R$)

definition
zero-divisor :: [-, *'a*] ⇒ *bool* **where**
zero-divisor R a ⟷ (∃ *x*∈ *carrier R*. *x* ≠ $\mathbf{0}_R$ ∧ *x* $\cdot_{r\,R}$ *a* = $\mathbf{0}_R$)

definition
primary-ideal :: [-, *'a set*] ⇒ *bool* **where**
primary-ideal R q ⟷ *ideal R q* ∧ ($1_{r\,R}$) ∉ *q* ∧
(∀ *x*∈ *carrier R*. ∀ *y*∈ *carrier R*.
x $\cdot_{r\,R}$ *y* ∈ *q* ⟶ (∃ *n*. (*npow R x n*) ∈ *q* ∨ *y* ∈ *q*))

definition
prime-ideal :: [-, *'a set*] ⇒ *bool* **where**
prime-ideal R p ⟷ *ideal R p* ∧ ($1_{r\,R}$) ∉ *p* ∧ (∀ *x*∈ *carrier R*. ∀ *y*∈ *carrier R*.

$(x \cdot_{r\,R}\ y \in p \longrightarrow x \in p \vee y \in p))$

definition
maximal-ideal :: [-, *'a set*] ⇒ *bool* **where**
maximal-ideal R mx ⟷ *ideal R mx* ∧ $1_{r\,R} \notin mx$ ∧
$\{J.\ (ideal\ R\ J \wedge mx \subseteq J)\} = \{mx,\ carrier\ R\}$

lemma (**in** *Ring*) *maximal-ideal-ideal*:⟦*maximal-ideal R mx*⟧ ⟹ *ideal R mx*
⟨*proof*⟩

lemma (**in** *Ring*) *maximal-ideal-proper*:*maximal-ideal R mx* ⟹ $1_r \notin mx$
⟨*proof*⟩

lemma (**in** *Ring*) *prime-ideal-ideal*:*prime-ideal R I* ⟹ *ideal R I*
⟨*proof*⟩

lemma (**in** *Ring*) *prime-ideal-proper*:*prime-ideal R I* ⟹ $I \neq carrier\ R$
⟨*proof*⟩

lemma (**in** *Ring*) *prime-ideal-proper1*:*prime-ideal R p* ⟹ $1_r \notin p$
⟨*proof*⟩

lemma (**in** *Ring*) *primary-ideal-ideal*:*primary-ideal R q* ⟹ *ideal R q*
⟨*proof*⟩

lemma (**in** *Ring*) *primary-ideal-proper1*:*primary-ideal R q* ⟹ $1_r \notin q$
⟨*proof*⟩

lemma (**in** *Ring*) *prime-elems-mult-not*:⟦*prime-ideal R P*; *x* ∈ *carrier R*;
y ∈ *carrier R*; *x* ∉ *P*; *y* ∉ *P* ⟧ ⟹ $x \cdot_r y \notin P$
⟨*proof*⟩

lemma (**in** *Ring*) *prime-is-primary*:*prime-ideal R p* ⟹ *primary-ideal R p*
⟨*proof*⟩

lemma (**in** *Ring*) *maximal-prime-Tr0*:⟦*maximal-ideal R mx*; *x* ∈ *carrier R*; *x* ∉ *mx*⟧
⟹ $mx \mp (Rxa\ R\ x) = carrier\ R$
⟨*proof*⟩

lemma (**in** *Ring*) *maximal-prime*:*maximal-ideal R mx* ⟹ *prime-ideal R mx*
⟨*proof*⟩

lemma (**in** *Ring*) *chains-un*:⟦*c* ∈ *chains* $\{I.\ ideal\ R\ I \wedge I \subset carrier\ R\}$; *c* ≠ {}⟧
⟹ *ideal R* $(\bigcup c)$
⟨*proof*⟩

lemma (**in** *Ring*) *zeroring-no-maximal*:*zeroring R* ⟹ ¬ (∃ *I*. *maximal-ideal R I*)

⟨*proof*⟩

lemma (**in** *Ring*) *id-maximal-Exist*:¬(*zeroring R*) ⟹ ∃ *I*. *maximal-ideal R I*
⟨*proof*⟩

definition
ideal-Int :: [-, *'a set set*] ⇒ *'a set* **where**
ideal-Int R S == ⋂ *S*

lemma (**in** *Ring*) *ideal-Int-ideal*:⟦*S* ⊆ {*I*. *ideal R I*}; *S*≠{}⟧ ⟹
ideal R (⋂ *S*)
⟨*proof*⟩

lemma (**in** *Ring*) *sum-prideals-Int*:⟦∀ *l* ≤ *n*. *f l* ∈ *carrier R*;
S = {*I*. *ideal R I* ∧ *f* ‘ {*i*. *i* ≤ *n*} ⊆ *I*}⟧ ⟹
(*sum-pr-ideals R f n*) = ⋂ *S*
⟨*proof*⟩

This proves that (*sum-pr-ideals R f n*) is the smallest ideal containing *f* ‘ (*Nset n*)

primrec *ideal-n-prod*::[(*'a*, *'m*) *Ring-scheme*, *nat*, *nat* ⇒ *'a set*] ⇒ *'a set*
where
ideal-n-prod0: *ideal-n-prod R 0 J* = *J 0*
| *ideal-n-prodSn*: *ideal-n-prod R* (*Suc n*) *J* =
(*ideal-n-prod R n J*) ◇$_r$ *R* (*J* (*Suc n*))

abbreviation
IDNPROD (‹(*3i*Π_,_ -)› [*98*,*98*,*99*]*98*) **where**
*i*Π$_{R,n}$ *J* == *ideal-n-prod R n J*

primrec
ideal-pow :: [*'a set*, (*'a*, *'more*) *Ring-scheme*, *nat*] ⇒ *'a set*
(‹(*3-*/ ◇_ _)› [*120*,*120*,*121*]*120*)
where
ip0: *I* ◇*R 0* = *carrier R*
| *ipSuc*: *I* ◇*R* (*Suc n*) = *I* ◇$_r$ *R* (*I* ◇*R n*)

lemma (**in** *Ring*) *prod-mem-prod-ideals*:⟦*ideal R I*; *ideal R J*; *i* ∈ *I*; *j* ∈ *J*⟧ ⟹
i ·$_r$ *j* ∈ (*I* ◇$_r$ *J*)
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-prod-ideal*:⟦*ideal R I*; *ideal R J* ⟧ ⟹
ideal R (*I* ◇$_r$ *J*)
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-prod-commute*:⟦*ideal R I*; *ideal R J*⟧ ⟹
I ◇$_r$ *J* = *J* ◇$_r$ *I*
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-prod-subTr*:⟦*ideal R I*; *ideal R J*; *ideal R C*;
$\forall\, i{\in}I.\ \forall\, j{\in}J.\ i \cdot_r j \in C⟧ \Longrightarrow I \diamondsuit_r J \subseteq C$
⟨*proof*⟩

lemma (**in** *Ring*) *n-prod-idealTr*:
$(\forall\, k \leq n.\ ideal\ R\ (J\ k)) \longrightarrow ideal\ R\ (ideal\text{-}n\text{-}prod\ R\ n\ J)$
⟨*proof*⟩

lemma (**in** *Ring*) *n-prod-ideal*:⟦$\forall\, k \leq n.\ ideal\ R\ (J\ k)$⟧
$\Longrightarrow\ ideal\ R\ (ideal\text{-}n\text{-}prod\ R\ n\ J)$
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-prod-la1*:⟦*ideal R I*; *ideal R J*⟧ $\Longrightarrow (I \diamondsuit_r J) \subseteq I$
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-prod-el1*:⟦*ideal R I*; *ideal R J*; $a \in (I \diamondsuit_r J)$⟧ $\Longrightarrow$
$a \in I$
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-prod-la2*:⟦*ideal R I*; *ideal R J* ⟧ $\Longrightarrow (I \diamondsuit_r J) \subseteq J$
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-prod-sub-Int*:⟦*ideal R I*; *ideal R J* ⟧ $\Longrightarrow$
$(I \diamondsuit_r J) \subseteq I \cap J$
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-prod-el2*:⟦*ideal R I*; *ideal R J*; $a \in (I \diamondsuit_r J)$⟧ $\Longrightarrow$
$a \in J$
⟨*proof*⟩

$i\Pi_{R,n}\ J$ is the product of ideals

lemma (**in** *Ring*) *ele-n-prodTr0*:⟦$\forall\, k \leq (Suc\ n).\ ideal\ R\ (J\ k)$;
$a \in i\Pi_{R,(Suc\ n)}\ J$ ⟧ $\Longrightarrow a \in (i\Pi_{R,n}\ J) \wedge a \in (J\ (Suc\ n))$
⟨*proof*⟩

lemma (**in** *Ring*) *ele-n-prodTr1*:
$(\forall\, k \leq n.\ ideal\ R\ (J\ k)) \wedge a \in ideal\text{-}n\text{-}prod\ R\ n\ J \longrightarrow$
$(\forall\, k \leq n.\ a \in (J\ k))$
⟨*proof*⟩

lemma (**in** *Ring*) *ele-n-prod*:⟦$\forall\, k \leq n.\ ideal\ R\ (J\ k)$;
$a \in ideal\text{-}n\text{-}prod\ R\ n\ J$ ⟧ $\Longrightarrow\ \forall\, k \leq n.\ a \in (J\ k)$
⟨*proof*⟩

lemma (**in** *Ring*) *idealprod-whole-l*:*ideal R I* $\Longrightarrow$ $(carrier\ R) \diamondsuit_{r\,R} I = I$
⟨*proof*⟩

lemma (**in** *Ring*) *idealprod-whole-r*:*ideal R I* $\Longrightarrow$ $I \diamondsuit_r (carrier\ R) = I$
⟨*proof*⟩

lemma (**in** *Ring*) *idealpow-1-self*:*ideal R I* $\Longrightarrow$ $I \diamondsuit_R (Suc\ 0) = I$
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-pow-ideal*:*ideal R I* $\Longrightarrow$ *ideal R* $(I \diamondsuit_R n)$
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-prod-prime*:⟦*ideal R I*; *ideal R J*; *prime-ideal R P*;
$I \diamondsuit_r J \subseteq P$ ⟧ $\Longrightarrow$ $I \subseteq P \vee J \subseteq P$
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-n-prod-primeTr*:*prime-ideal R P* $\Longrightarrow$
$(\forall k \leq n.\ ideal\ R\ (J\ k)) \longrightarrow (ideal\text{-}n\text{-}prod\ R\ n\ J \subseteq P) \longrightarrow$
$(\exists i \leq n.\ (J\ i) \subseteq P)$
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-n-prod-prime*:⟦*prime-ideal R P*;
$\forall k \leq n.\ ideal\ R\ (J\ k)$; *ideal-n-prod R n J* $\subseteq P$⟧ $\Longrightarrow$
$\exists i \leq n.\ (J\ i) \subseteq P$
⟨*proof*⟩

definition
ppa::[-, *nat* ⇒ *'a set*, *'a set*, *nat*] ⇒ (*nat* ⇒ *'a*) **where**
ppa R P A i l = (*SOME x*. $x \in A \wedge x \in (P\ (skip\ i\ l)) \wedge x \notin P\ i$)

lemma (**in** *Ring*) *prod-primeTr*:⟦*prime-ideal R P*; *ideal R A*; ¬ $A \subseteq P$;
ideal R B; ¬ $B \subseteq P$ ⟧ $\Longrightarrow$ $\exists x.\ x \in A \wedge x \in B \wedge x \notin P$
⟨*proof*⟩

lemma (**in** *Ring*) *prod-primeTr1*:⟦$\forall k \leq (Suc\ n).$ *prime-ideal R* $(P\ k)$;
ideal R A; $\forall l \leq (Suc\ n).$ ¬ $(A \subseteq P\ l)$;
$\forall k \leq (Suc\ n).\ \forall l \leq (Suc\ n).\ k = l \vee \neg\ (P\ k) \subseteq (P\ l)$; $i \leq (Suc\ n)$⟧ $\Longrightarrow$
$\forall l \leq n.\ ppa\ R\ P\ A\ i\ l \in A \wedge$
$ppa\ R\ P\ A\ i\ l \in (P\ (skip\ i\ l)) \wedge ppa\ R\ P\ A\ i\ l \notin (P\ i)$
⟨*proof*⟩

lemma (**in** *Ring*) *ppa-mem*:⟦$\forall k \leq (Suc\ n).$ *prime-ideal R* $(P\ k)$; *ideal R A*;
$\forall l \leq (Suc\ n).$ ¬ $(A \subseteq P\ l)$;
$\forall k \leq (Suc\ n).\ \forall l \leq (Suc\ n).\ k = l \vee \neg\ (P\ k) \subseteq (P\ l)$;
$i \leq (Suc\ n)$; $l \leq n$⟧ $\Longrightarrow$ *ppa R P A i l* ∈ *carrier R*
⟨*proof*⟩

lemma (**in** *Ring*) *nsum-memrTr*:$(\forall i \leq n.\ f\ i \in carrier\ R) \longrightarrow$
$(\forall l \leq n.\ nsum\ R\ f\ l \in carrier\ R)$
⟨*proof*⟩

lemma (**in** *Ring*) *nsum-memr*:$\forall i \leq n.\ f\ i \in carrier\ R \Longrightarrow$
$\forall l \leq n.\ nsum\ R\ f\ l \in carrier\ R$

⟨*proof*⟩

lemma (**in** *Ring*) *nsum-ideal-incTr*:*ideal R A* ⟹
(∀ *i* ≤ *n*. *f i* ∈ *A*) ⟶ *nsum R f n* ∈ *A*
⟨*proof*⟩

lemma (**in** *Ring*) *nsum-ideal-inc*:⟦*ideal R A*; ∀ *i* ≤ *n*. *f i* ∈ *A*⟧ ⟹
nsum R f n ∈ *A*
⟨*proof*⟩

lemma (**in** *Ring*) *nsum-ideal-excTr*:*ideal R A* ⟹
(∀ *i* ≤ *n*. *f i* ∈ *carrier R*) ∧ (∃ *j* ≤ *n*. (∀ *l* ∈ {*i*. *i* ≤ *n*} −{*j*}. *f l* ∈ *A*)
∧ (*f j* ∉ *A*)) ⟶ *nsum R f n* ∉ *A*
⟨*proof*⟩

lemma (**in** *Ring*) *nsum-ideal-exc*:⟦*ideal R A*; ∀ *i* ≤ *n*. *f i* ∈ *carrier R*;
∃ *j* ≤ *n*. (∀ *l*∈{*i*. *i* ≤ *n*} −{*j*}. *f l* ∈ *A*) ∧ (*f j* ∉ *A*) ⟧ ⟹ *nsum R f n* ∉ *A*
⟨*proof*⟩

lemma (**in** *Ring*) *nprod-memTr*:(∀ *i* ≤ *n*. *f i* ∈ *carrier R*) ⟶
(∀ *l*. *l* ≤ *n* ⟶ *nprod R f l* ∈ *carrier R*)
⟨*proof*⟩

lemma (**in** *Ring*) *nprod-mem*:⟦∀ *i* ≤ *n*. *f i* ∈ *carrier R*; *l* ≤ *n*⟧ ⟹
nprod R f l ∈ *carrier R*
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-nprod-incTr*:*ideal R A* ⟹
(∀ *i* ≤ *n*. *f i* ∈ *carrier R*) ∧
(∃ *l* ≤ *n*. *f l* ∈ *A*) ⟶ *nprod R f n* ∈ *A*
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-nprod-inc*:⟦*ideal R A*; ∀ *i* ≤ *n*. *f i* ∈ *carrier R*;
∃ *l* ≤ *n*. *f l* ∈ *A*⟧ ⟹ *nprod R f n* ∈ *A*
⟨*proof*⟩

lemma (**in** *Ring*) *nprod-excTr*:*prime-ideal R P* ⟹
(∀ *i* ≤ *n*. *f i* ∈ *carrier R*) ∧ (∀ *l* ≤ *n*. *f l* ∉ *P*) ⟶
nprod R f n ∉ *P*
⟨*proof*⟩

lemma (**in** *Ring*) *prime-nprod-exc*:⟦*prime-ideal R P*; ∀ *i* ≤ *n*. *f i* ∈ *carrier R*;
∀ *l* ≤ *n*. *f l* ∉ *P*⟧ ⟹ *nprod R f n* ∉ *P*
⟨*proof*⟩

definition
nilrad :: - ⇒ *'a set* **where**
nilrad R = {*x*. *x* ∈ *carrier R* ∧ *nilpotent R x*}

lemma (**in** *Ring*) *id-nilrad-ideal*:*ideal R* (*nilrad R*)
⟨*proof*⟩

definition
rad-ideal :: [-, *'a set*] ⇒ *'a set* **where**
rad-ideal R I = {*a*. *a* ∈ *carrier R* ∧ *nilpotent* (*qring R I*) ((*pj R I*) *a*)}

lemma (**in** *Ring*) *id-rad-invim*:*ideal R I* ⟹
rad-ideal R I = (*rInvim R* (*qring R I*) (*pj R I*) (*nilrad* (*qring R I*)))
⟨*proof*⟩

lemma (**in** *Ring*) *id-rad-ideal*:*ideal R I* ⟹ *ideal R* (*rad-ideal R I*)

⟨*proof*⟩

lemma (**in** *Ring*) *id-rad-cont-I*:*ideal R I* ⟹ *I* ⊆ (*rad-ideal R I*)
⟨*proof*⟩

lemma (**in** *Ring*) *id-rad-set*:*ideal R I* ⟹
rad-ideal R I = {*x*. *x* ∈ *carrier R* ∧ (∃ *n*. *npow R x n* ∈ *I*)}
⟨*proof*⟩

lemma (**in** *Ring*) *rad-primary-prime*:*primary-ideal R q* ⟹
prime-ideal R (*rad-ideal R q*)
⟨*proof*⟩

lemma (**in** *Ring*) *npow-notin-prime*:⟦*prime-ideal R P*; *x* ∈ *carrier R*; *x* ∉ *P*⟧
⟹ ∀ *n*. *npow R x n* ∉ *P*
⟨*proof*⟩

lemma (**in** *Ring*) *npow-in-prime*:⟦*prime-ideal R P*; *x* ∈ *carrier R*;
∃ *n*. *npow R x n* ∈ *P* ⟧ ⟹ *x* ∈ *P*
⟨*proof*⟩

definition
mul-closed-set::[-, *'a set*] ⇒ *bool* **where**
mul-closed-set R S ⟷ *S* ⊆ *carrier R* ∧ (∀ *s*∈*S*. ∀ *t*∈*S*. *s* ·$_{rR}$ *t* ∈ *S*)

locale *Idomain* = *Ring* +
assumes *idom*:
⟦*a* ∈ *carrier R*; *b* ∈ *carrier R*; *a* ·$_r$ *b* = **0**⟧ ⟹ *a* = **0** ∨ *b* = **0**

locale *Corps* =
fixes *K* (**structure**)
assumes *f-is-ring*: *Ring K*
and *f-inv*: ∀ *x*∈*carrier K* − {**0**}. ∃ *x'* ∈ *carrier K*. *x'* ·$_r$ *x* = *1*$_r$

lemma (**in** *Ring*) *mul-closed-set-sub*:*mul-closed-set R S* $\Longrightarrow$ $S \subseteq$ *carrier R*
⟨*proof*⟩

lemma (**in** *Ring*) *mul-closed-set-tOp-closed*:⟦*mul-closed-set R S*; $s \in S$;
$t \in S$⟧ $\Longrightarrow$ $s \cdot_r t \in S$
⟨*proof*⟩

lemma (**in** *Corps*) *f-inv-unique*:⟦ $x \in$ *carrier K* $-$ {**0**}; $x' \in$ *carrier K*;
$x'' \in$ *carrier K*; $x' \cdot_r x = 1_r$; $x'' \cdot_r x = 1_r$ ⟧ $\Longrightarrow$ $x' = x''$
⟨*proof*⟩

definition
invf :: [-, 'a] ⇒ 'a **where**
invf K x = (*THE y*. $y \in$ *carrier K* $\wedge$ $y \cdot_{r\,K} x = 1_{r\,K}$)

lemma (**in** *Corps*) *invf-inv*:$x \in$ *carrier K* $-$ {**0**} $\Longrightarrow$
(*invf K x*) $\in$ *carrier K* $\wedge$ (*invf K x*) $\cdot_r x = 1_r$
⟨*proof*⟩

definition
npowf :: - ⇒ 'a ⇒ *int* ⇒ 'a **where**
npowf K x n =
(*if* 0 $\leq$ *n then npow K x* (*nat n*) *else npow K* (*invf K x*) (*nat* (− *n*)))

abbreviation
NPOWF :: ['a, -, *int*] ⇒ 'a (‹(3-_^)› [77,77,78]77) **where**
$a_K{}^n$ == *npowf K a n*

abbreviation
IOP :: ['a, -] ⇒ 'a (‹(-^)› [87,88]87) **where**
a^{-K} == *invf K a*

lemma (**in** *Idomain*) *idom-is-ring*: *Ring R* ⟨*proof*⟩

lemma (**in** *Idomain*) *idom-tOp-nonzeros*:⟦$x \in$ *carrier R*;
$y \in$ *carrier R*; $x \neq$ **0**; $y \neq$ **0**⟧ $\Longrightarrow$ $x \cdot_r y \neq$ **0**
⟨*proof*⟩

lemma (**in** *Idomain*) *idom-potent-nonzero*:
⟦$x \in$ *carrier R*; $x \neq$ **0**⟧ $\Longrightarrow$ *npow R x n* $\neq$ **0**
⟨*proof*⟩

lemma (**in** *Idomain*) *idom-potent-unit*:⟦$a \in$ *carrier R*; 0 < *n*⟧
$\Longrightarrow$ (*Unit R a*) = (*Unit R* (*npow R a n*))
⟨*proof*⟩

lemma (**in** *Idomain*) *idom-mult-cancel-r*:⟦$a \in$ *carrier R*;

$b \in$ *carrier* R; $c \in$ *carrier* R; $c \neq \mathbf{0}$; $a \cdot_r c = b \cdot_r c\rrbracket \Longrightarrow a = b$
⟨*proof*⟩

lemma (**in** *Idomain*) *idom-mult-cancel-l*:⟦$a \in$ *carrier* R;
$b \in$ *carrier* R; $c \in$ *carrier* R; $c \neq \mathbf{0}$; $c \cdot_r a = c \cdot_r b\rrbracket \Longrightarrow a = b$
⟨*proof*⟩

lemma (**in** *Corps*) *invf-closed1*:$x \in$ *carrier* $K - \{\mathbf{0}\} \Longrightarrow$
invf K $x \in$ (*carrier* K) $- \{\mathbf{0}\}$
⟨*proof*⟩

lemma (**in** *Corps*) *linvf*:$x \in$ *carrier* $K - \{\mathbf{0}\} \Longrightarrow$ (*invf* K x) $\cdot_r$ $x = 1_r$
⟨*proof*⟩

lemma (**in** *Corps*) *field-is-ring*:*Ring* K
⟨*proof*⟩

lemma (**in** *Corps*) *invf-one*:$1_r \neq \mathbf{0} \Longrightarrow$ *invf* K $(1_r) = 1_r$
⟨*proof*⟩

lemma (**in** *Corps*) *field-tOp-assoc*:⟦$x \in$ *carrier* K; $y \in$ *carrier* K; $z \in$ *carrier* $K\rrbracket$
$\Longrightarrow x \cdot_r y \cdot_r z = x \cdot_r (y \cdot_r z)$
⟨*proof*⟩

lemma (**in** *Corps*) *field-tOp-commute*:⟦$x \in$ *carrier* K; $y \in$ *carrier* $K\rrbracket$
$\Longrightarrow x \cdot_r y = y \cdot_r x$
⟨*proof*⟩

lemma (**in** *Corps*) *field-inv-inv*:⟦$x \in$ *carrier* K; $x \neq \mathbf{0}\rrbracket \Longrightarrow (x^{-K})^{-K} = x$
⟨*proof*⟩

lemma (**in** *Corps*) *field-is-idom*:*Idomain* K
⟨*proof*⟩

lemma (**in** *Corps*) *field-potent-nonzero*:⟦$x \in$ *carrier* K; $x \neq \mathbf{0}\rrbracket \Longrightarrow$
$x^{\frown K\ n} \neq \mathbf{0}$
⟨*proof*⟩

lemma (**in** *Corps*) *field-potent-nonzero1*:⟦$x \in$ *carrier* K; $x \neq \mathbf{0}\rrbracket \Longrightarrow {x_K}^n \neq \mathbf{0}$
⟨*proof*⟩

lemma (**in** *Corps*) *field-nilp-zero*:⟦$x \in$ *carrier* K; $x^{\frown K\ n} = \mathbf{0}\rrbracket \Longrightarrow x = \mathbf{0}$
⟨*proof*⟩

lemma (**in** *Corps*) *npowf-mem*:⟦$a \in$ *carrier* K; $a \neq \mathbf{0}\rrbracket \Longrightarrow$
npowf K a $n \in$ *carrier* K
⟨*proof*⟩

lemma (**in** *Corps*) *field-npowf-exp-zero*:⟦$a \in$ *carrier* K; $a \neq \mathbf{0}\rrbracket \Longrightarrow$

$$npowf\ K\ a\ 0 = 1_r$$

⟨*proof*⟩

lemma (**in** *Corps*) *npow-exp-minusTr1*:⟦$x \in carrier\ K$; $x \neq \mathbf{0}$; $0 \leq i$⟧ ⟹
$0 \leq i - (int\ j) \longrightarrow\ {x_K}^{(i - (int\ j))} = {x^{\frown K}}^{(nat\ i)} \cdot_r {(x^{\text{-}K})^{\frown K}}^{j}$
⟨*proof*⟩

lemma (**in** *Corps*) *npow-exp-minusTr2*:⟦$x \in carrier\ K$; $x \neq \mathbf{0}$; $0 \leq i$; $0 \leq j$;
$0 \leq i - j$⟧ ⟹ ${x_K}^{(i - j)} = {x^{\frown K}}^{(nat\ i)} \cdot_r {(x^{\text{-}K})^{\frown K}}^{(nat\ j)}$
⟨*proof*⟩

lemma (**in** *Corps*) *npowf-inv*:⟦$x \in carrier\ K$; $x \neq \mathbf{0}$; $0 \leq j$⟧ ⟹ ${x_K}^{j} = {(x^{\text{-}K})_K}^{(-j)}$
⟨*proof*⟩

lemma (**in** *Corps*) *npowf-inv1*:⟦$x \in carrier\ K$; $x \neq \mathbf{0}$; $\neg\ 0 \leq j$⟧ ⟹
${x_K}^{j} = {(x^{\text{-}K})_K}^{(-j)}$
⟨*proof*⟩

lemma (**in** *Corps*) *npowf-inverse*:⟦$x \in carrier\ K$; $x \neq \mathbf{0}$⟧ ⟹ ${x_K}^{j} = {(x^{\text{-}K})_K}^{(-j)}$
⟨*proof*⟩

lemma (**in** *Corps*) *npowf-expTr1*:⟦$x \in carrier\ K$; $x \neq \mathbf{0}$; $0 \leq i$; $0 \leq j$;
$0 \leq i - j$⟧ ⟹ ${x_K}^{(i - j)} = {x_K}^{i} \cdot_r {x_K}^{(-j)}$
⟨*proof*⟩

lemma (**in** *Corps*) *npowf-expTr2*:⟦$x \in carrier\ K$; $x \neq \mathbf{0}$; $0 \leq i + j$⟧ ⟹
${x_K}^{(i + j)} = {x_K}^{i} \cdot_r {x_K}^{j}$
⟨*proof*⟩

lemma (**in** *Corps*) *npowf-exp-add*:⟦$x \in carrier\ K$; $x \neq \mathbf{0}$⟧ ⟹
${x_K}^{(i + j)} = {x_K}^{i} \cdot_r {x_K}^{j}$
⟨*proof*⟩

lemma (**in** *Corps*) *npowf-exp-1-add*:⟦$x \in carrier\ K$; $x \neq \mathbf{0}$⟧ ⟹
${x_K}^{(1 + j)} = x \cdot_r {x_K}^{j}$
⟨*proof*⟩

lemma (**in** *Corps*) *npowf-minus*:⟦$x \in carrier\ K$; $x \neq \mathbf{0}$⟧ ⟹ $({x_K}^{j})^{\text{-}K} = {x_K}^{(-\ j)}$
⟨*proof*⟩

lemma (**in** *Ring*) *residue-fieldTr*:⟦*maximal-ideal R mx*; $x \in carrier(qring\ R\ mx)$;
$x \neq \mathbf{0}_{(qring\ R\ mx)}$⟧ ⟹∃ y∈*carrier* $(qring\ R\ mx)$. $y \cdot_{r(qring\ R\ mx)} x = 1_{r(qring\ R\ mx)}$
⟨*proof*⟩

lemma (**in** *Ring*) *residue-field-cd*:*maximal-ideal R mx* $\Longrightarrow$
Corps (*qring R mx*)
⟨*proof*⟩

lemma (**in** *Ring*) *maximal-set-idealTr*:
maximal-set $\{I.\ ideal\ R\ I \wedge S \cap I = \{\}\}\ mx \Longrightarrow ideal\ R\ mx$
⟨*proof*⟩

lemma (**in** *Ring*) *maximal-setTr*:$[\![$*maximal-set* $\{I.\ ideal\ R\ I \wedge S \cap I = \{\}\}\ mx$;
ideal R J; $mx \subset J\ ]\!] \Longrightarrow S \cap J \neq \{\}$
⟨*proof*⟩

lemma (**in** *Ring*) *mulDisj*:$[\![$*mul-closed-set R S*; $1_r \in S$; $\mathbf{0} \notin S$;
$T = \{I.\ ideal\ R\ I \wedge S \cap I = \{\}\}$; *maximal-set T mx* $]\!] \Longrightarrow$ *prime-ideal R mx*
⟨*proof*⟩

lemma (**in** *Ring*) *ex-mulDisj-maximal*:$[\![$*mul-closed-set R S*; $\mathbf{0} \notin S$; $1_r \in S$;
$T = \{I.\ ideal\ R\ I \wedge S \cap I = \{\}\}]\!] \Longrightarrow\ \exists mx.\ maximal\text{-}set\ T\ mx$
⟨*proof*⟩

lemma (**in** *Ring*) *ex-mulDisj-prime*:$[\![$*mul-closed-set R S*; $\mathbf{0} \notin S$; $1_r \in S]\!] \Longrightarrow$
$\exists mx.\ prime\text{-}ideal\ R\ mx \wedge S \cap mx = \{\}$
⟨*proof*⟩

lemma (**in** *Ring*) *nilradTr1*:$\neg$ *zeroring R* $\Longrightarrow$ *nilrad R* $= \bigcap \{p.\ prime\text{-}ideal\ R\ p\}$
⟨*proof*⟩

lemma (**in** *Ring*) *nonilp-residue-nilrad*:$[\![\neg$ *zeroring R*; $x \in$ *carrier R*;
nilpotent (*qring R* (*nilrad R*)) ($x \uplus_R$ (*nilrad R*))$]\!] \Longrightarrow$
$x \uplus_R$ (*nilrad R*) $= \mathbf{0}_{(qring\ R\ (nilrad\ R))}$
⟨*proof*⟩

lemma (**in** *Ring*) *ex-contid-maximal*:$[\![$ $S = \{1_r\}$; $\mathbf{0} \notin S$; *ideal R I*; $I \cap S = \{\}$;
$T = \{J.\ ideal\ R\ J \wedge S \cap J = \{\} \wedge I \subseteq J\}]\!] \Longrightarrow \exists mx.\ maximal\text{-}set\ T\ mx$
⟨*proof*⟩

lemma (**in** *Ring*) *contid-maximal*:$[\![S = \{1_r\}$; $\mathbf{0} \notin S$; *ideal R I*; $I \cap S = \{\}$;
$T = \{J.\ ideal\ R\ J \wedge S \cap J = \{\} \wedge I \subseteq J\}$; *maximal-set T mx*$]\!] \Longrightarrow$
maximal-ideal R mx
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-contained-maxid*:$[\![\neg$(*zeroring R*); *ideal R I*; $1_r \notin I]\!] \Longrightarrow$
$\exists mx.\ maximal\text{-}ideal\ R\ mx \wedge I \subseteq mx$
⟨*proof*⟩

lemma (**in** *Ring*) *nonunit-principal-id*:$[\![a \in carrier\ R;\ \neg\ (Unit\ R\ a)]\!] \Longrightarrow$
$(R \diamondsuit_p\ a) \neq (carrier\ R)$
⟨*proof*⟩

lemma (**in** *Ring*) *nonunit-contained-maxid*:$[\![\neg(zeroring\ R);\ a \in carrier\ R;$
$\neg\ Unit\ R\ a\ ]\!] \Longrightarrow\ \exists\, mx.\ maximal\text{-}ideal\ R\ mx \wedge a \in\ mx$
⟨*proof*⟩

definition
local-ring :: - ⇒ *bool* **where**
local-ring R == *Ring R* ∧ ¬ *zeroring R* ∧ *card* {*mx*. *maximal-ideal R mx*} = *1*

lemma (**in** *Ring*) *local-ring-diff*:$[\![\neg\ zeroring\ R;\ ideal\ R\ mx;\ mx \neq carrier\ R;$
$\forall\, a{\in}\ (carrier\ R - mx).\ Unit\ R\ a\ ]\!] \Longrightarrow local\text{-}ring\ R \wedge maximal\text{-}ideal\ R\ mx$
⟨*proof*⟩

lemma (**in** *Ring*) *localring-unit*:$[\![\neg\ zeroring\ R;\ maximal\text{-}ideal\ R\ mx;$
$\forall\, x.\ x \in mx \longrightarrow Unit\ R\ (x \pm 1_r)\ ]\!] \Longrightarrow local\text{-}ring\ R$
⟨*proof*⟩

definition
J-rad ::- ⇒ *'a set* **where**
J-rad R = (*if* (*zeroring R*) *then* (*carrier R*) *else*
$\bigcap$ {*mx*. *maximal-ideal R mx*})

lemma (**in** *Ring*) *zeroring-J-rad-empty*:*zeroring R* ⟹ *J-rad R* = *carrier R*
⟨*proof*⟩

lemma (**in** *Ring*) *J-rad-mem*:*x* ∈ *J-rad R* ⟹ *x* ∈ *carrier R*
⟨*proof*⟩

lemma (**in** *Ring*) *J-rad-unit*:$[\![\neg\ zeroring\ R;\ x \in J\text{-}rad\ R]\!] \Longrightarrow$
$\forall\, y.\ (y{\in}\ carrier\ R \longrightarrow Unit\ R\ (1_r \pm (-_a\ x) \cdot_r\ y))$
⟨*proof*⟩

end

theory *Algebra5* **imports** *Algebra4* **begin**

4.6 Operation of ideals

lemma (**in** *Ring*) *ideal-sumTr1*:$[\![ideal\ R\ A;\ ideal\ R\ B]\!] \Longrightarrow$
$A \mp B = \bigcap\ \{J.\ ideal\ R\ J \wedge (A \cup B) \subseteq J\}$
⟨*proof*⟩

lemma (**in** *Ring*) *sum-ideals-commute*:$[\![ideal\ R\ A;\ ideal\ R\ B]\!] \Longrightarrow$
$A \mp B = B \mp A$

⟨*proof*⟩

lemma (**in** *Ring*) *ideal-prod-mono1*:⟦*ideal R A*; *ideal R B*; *ideal R C*;
A ⊆ *B* ⟧ ⟹ *A* ◇$_r$ *C* ⊆ *B* ◇$_r$ *C*
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-prod-mono2*:⟦*ideal R A*; *ideal R B*; *ideal R C*;
A ⊆ *B* ⟧ ⟹ *C* ◇$_r$ *A* ⊆ *C* ◇$_r$ *B*
⟨*proof*⟩

lemma (**in** *Ring*) *cont-ideal-prod*:⟦*ideal R A*; *ideal R B*; *ideal R C*;
A ⊆ *C*; *B* ⊆ *C* ⟧ ⟹ *A* ◇$_r$ *B* ⊆ *C*
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-distrib*:⟦*ideal R A*; *ideal R B*; *ideal R C*⟧ ⟹
A ◇$_r$ (*B* ∓ *C*) = *A* ◇$_r$ *B* ∓ *A* ◇$_r$ *C*
⟨*proof*⟩

definition
coprime-ideals::[-, *'a set*, *'a set*] ⇒ *bool* **where**
coprime-ideals R A B ⟷ *A* ∓$_R$ *B* = *carrier R*

lemma (**in** *Ring*) *coprimeTr*:⟦*ideal R A*; *ideal R B*⟧ ⟹
coprime-ideals R A B = (∃ *a* ∈ *A*. ∃ *b* ∈ *B*. *a* ± *b* = *1*$_r$)
⟨*proof*⟩

lemma (**in** *Ring*) *coprime-int-prod*:⟦*ideal R A*; *ideal R B*; *coprime-ideals R A B*⟧
⟹ *A* ∩ *B* = *A* ◇$_r$ *B*
⟨*proof*⟩

lemma (**in** *Ring*) *coprime-elems*:⟦*ideal R A*; *ideal R B*; *coprime-ideals R A B*⟧ ⟹
∃ *a*∈*A*. ∃ *b*∈*B*. *a* ± *b* = *1*$_r$
⟨*proof*⟩

lemma (**in** *Ring*) *coprime-elemsTr*:⟦*ideal R A*; *ideal R B*; *a*∈*A*; *b*∈*B*; *a* ± *b* = *1*$_r$⟧

⟹ *pj R A b* = *1*$_{r(qring\ R\ A)}$ ∧ *pj R B a* = *1*$_{r(qring\ R\ B)}$
⟨*proof*⟩

lemma (**in** *Ring*) *partition-of-unity*:⟦*ideal R A*; *a* ∈ *A*; *b* ∈ *carrier R*;
a ± *b* = *1*$_r$; *u* ∈ *carrier R*; *v* ∈ *carrier R*⟧ ⟹
pj R A (*a* ·$_r$ *v* ± *b* ·$_r$ *u*) = *pj R A u*
⟨*proof*⟩

lemma (**in** *Ring*) *coprimes-commute*:⟦*ideal R A*; *ideal R B*; *coprime-ideals R A B*
⟧
⟹ *coprime-ideals R B A*
⟨*proof*⟩

lemma (**in** *Ring*) *coprime-surjTr*:⟦*ideal R A*; *ideal R B*; *coprime-ideals R A B*;
X ∈ *carrier* (*qring R A*); *Y* ∈ *carrier* (*qring R B*) ⟧ ⟹
∃ *r*∈*carrier R*. *pj R A r* = *X* ∧ *pj R B r* = *Y*
⟨*proof*⟩

lemma (**in** *Ring*) *coprime-n-idealsTr0*:⟦*ideal R A*; *ideal R B*; *ideal R C*;
coprime-ideals R A C; *coprime-ideals R B C* ⟧ ⟹
coprime-ideals R (*A* ◇$_r$ *B*) *C*
⟨*proof*⟩

lemma (**in** *Ring*) *coprime-n-idealsTr1*:*ideal R C* ⟹
(∀ *k* ≤ *n*. *ideal R* (*J k*)) ∧ (∀ *i* ≤ *n*. *coprime-ideals R* (*J i*) *C*) ⟶
coprime-ideals R (*i*Π$_{R,n}$ *J*) *C*
⟨*proof*⟩

lemma (**in** *Ring*) *coprime-n-idealsTr2*:⟦*ideal R C*; (∀ *k* ≤ *n*. *ideal R* (*J k*));
(∀ *i* ≤ *n*. *coprime-ideals R* (*J i*) *C*) ⟧ ⟹
coprime-ideals R (*i*Π$_{R,n}$ *J*) *C*
⟨*proof*⟩

lemma (**in** *Ring*) *coprime-n-idealsTr3*:(∀ *k* ≤ (*Suc n*). *ideal R* (*J k*)) ∧
(∀ *i* ≤ (*Suc n*). ∀ *j* ≤ (*Suc n*). *i* ≠ *j* ⟶
coprime-ideals R (*J i*) (*J j*)) ⟶ *coprime-ideals R* (*i*Π$_{R,n}$ *J*) (*J* (*Suc n*))
⟨*proof*⟩

lemma (**in** *Ring*) *coprime-n-idealsTr4*:⟦(∀ *k* ≤ (*Suc n*). *ideal R* (*J k*)) ∧
(∀ *i* ≤ (*Suc n*). ∀ *j* ≤ (*Suc n*). *i* ≠ *j* ⟶
coprime-ideals R (*J i*) (*J j*))⟧ ⟹ *coprime-ideals R* (*i*Π$_{R,n}$ *J*) (*J* (*Suc n*))
⟨*proof*⟩

4.7 Direct product1, general case

definition
prod-tOp :: ['*i set*, '*i* ⇒ ('*a*, '*m*) *Ring-scheme*] ⇒
('*i* ⇒ '*a*) ⇒ ('*i* ⇒ '*a*) ⇒ ('*i* ⇒ '*a*) **where**
prod-tOp I A = (λ*f*∈*carr-prodag I A*. λ*g*∈*carr-prodag I A*.
λ*x*∈*I*. (*f x*) ·$_{r(A\ x)}$ (*g x*))

definition
prod-one::['*i set*, '*i* ⇒ ('*a*, '*m*) *Ring-scheme*] ⇒ ('*i* ⇒ '*a*) **where**
prod-one I A == λ*x*∈*I*. 1$_{r(A\ x)}$

definition
prodrg :: ['*i set*, '*i* ⇒ ('*a*, '*more*) *Ring-scheme*] ⇒ ('*i* ⇒ '*a*) *Ring* **where**
prodrg I A = (⦇*carrier* = *carr-prodag I A*, *pop* = *prod-pOp I A*, *mop* =
prod-mOp I A, *zero* = *prod-zero I A*, *tp* = *prod-tOp I A*,

un = prod-one I A ⟫

abbreviation
PRODRING (‹(*rΠ*_/ -)› [72,73]72) **where**
$r\Pi_I\ A == prodrg\ I\ A$

definition
augm-func :: [*nat*, *nat* ⇒ *'a*,*'a set*, *nat*, *nat* ⇒ *'a*, *'a set*] ⇒ *nat* ⇒ *'a* **where**
augm-func n f A m g B = (λ*i*∈{*j*. *j* ≤ (*n* + *m*)}. *if i* ≤ *n then f i else*
if (*Suc n*) ≤ *i* ∧ *i* ≤ *n* + *m then g* ((*sliden* (*Suc n*)) *i*) *else undefined*)

definition
ag-setfunc :: [*nat*, *nat* ⇒ (*'a*, *'more*) *Ring-scheme*, *nat*,
nat ⇒ (*'a*, *'more*) *Ring-scheme*] ⇒ (*nat* ⇒ *'a*) *set* ⇒ (*nat* ⇒ *'a*) *set*
⇒ (*nat* ⇒ *'a*) *set* **where**
ag-setfunc n B1 m B2 X Y =
{*f*. ∃ *g*. ∃ *h*. (*g*∈*X*) ∧(*h*∈*Y*) ∧(*f* = (*augm-func n g* (*Un-carrier* {*j*. *j* ≤ *n*} *B1*)
m h (*Un-carrier* {*j*. *j* ≤ (*m* − *1*)} *B2*)))}

primrec
ac-fProd-Rg :: [*nat*, *nat* ⇒ (*'a*, *'more*) *Ring-scheme*] ⇒
(*nat* ⇒ *'a*) *set*
where
fprod-0: *ac-fProd-Rg 0 B* = *carr-prodag* {*0*::*nat*} *B*
| *frpod-n*: *ac-fProd-Rg* (*Suc n*) *B* = *ag-setfunc n B* (*Suc 0*) (*compose* {*0*::*nat*}
B (*slide* (*Suc n*))) (*carr-prodag* {*j*. *j* ≤ *n*} *B*) (*carr-prodag* {*0*} (*compose* {*0*} *B*
(*slide* (*Suc n*))))

definition
prodB1 :: [(*'a*, *'m*) *Ring-scheme*, (*'a*, *'m*) *Ring-scheme*] ⇒
(*nat* ⇒ (*'a*, *'m*) *Ring-scheme*) **where**
prodB1 R S = (λ*k*. *if k=0 then R else if k=Suc 0 then S else*
undefined)

definition
Prod2Rg :: [(*'a*, *'m*) *Ring-scheme*, (*'a*, *'m*) *Ring-scheme*]
⇒ (*nat* ⇒ *'a*) *Ring* (**infixl** ‹$\bigoplus_r$› *80*) **where**
A1 $\bigoplus_r$ *A2* = *prodrg* {*0*, *Suc 0*} (*prodB1 A1 A2*)

Don't try (*Prod-ring* (*Nset n*) *B*) $\bigoplus_r$ (*B* (*Suc n*))

lemma *carr-prodrg-mem-eq*:⟦*f* ∈ *carrier* ($r\Pi_I\ A$); *g* ∈ *carrier* ($r\Pi_I\ A$);
∀ *i*∈*I*. *f i* = *g i* ⟧ ⟹ *f* = *g*
⟨*proof*⟩

lemma *prod-tOp-mem*:⟦∀ *k*∈*I*. *Ring* (*A k*); *X* ∈ *carr-prodag I A*;
Y ∈ *carr-prodag I A*⟧ ⟹ *prod-tOp I A X Y* ∈ *carr-prodag I A*

⟨*proof*⟩

lemma *prod-tOp-func*:∀ *k*∈*I*. *Ring* (*A k*) ⟹
prod-tOp I A ∈ *carr-prodag I A* → *carr-prodag I A* → *carr-prodag I A*
⟨*proof*⟩

lemma *prod-one-func*:∀ *k*∈*I*. *Ring* (*A k*) ⟹
prod-one I A ∈ *carr-prodag I A*
⟨*proof*⟩

lemma *prodrg-carrier*:∀ *k*∈*I*. *Ring* (*A k*) ⟹
carrier (*prodrg I A*) = *carrier* (*prodag I A*)
⟨*proof*⟩

lemma *prodrg-ring*:∀ *k*∈*I*. *Ring* (*A k*) ⟹ *Ring* (*prodrg I A*)
⟨*proof*⟩

lemma *prodrg-elem-extensional*:⟦∀ *k*∈*I*. *Ring* (*A k*); *f* ∈ *carrier* (*prodrg I A*)⟧
⟹ *f* ∈ *extensional I*
⟨*proof*⟩

lemma *prodrg-pOp*:∀ *k*∈*I*. *Ring* (*A k*) ⟹
pop (*prodrg I A*) = *prod-pOp I A*
⟨*proof*⟩

lemma *prodrg-mOp*:∀ *k*∈*I*. *Ring* (*A k*) ⟹
mop (*prodrg I A*) = *prod-mOp I A*
⟨*proof*⟩

lemma *prodrg-zero*:∀ *k*∈*I*. *Ring* (*A k*) ⟹
zero (*prodrg I A*) = *prod-zero I A*
⟨*proof*⟩

lemma *prodrg-tOp*:∀ *k*∈*I*. *Ring* (*A k*) ⟹
tp (*prodrg I A*) = *prod-tOp I A*
⟨*proof*⟩

lemma *prodrg-one*:∀ *k*∈*I*. *Ring* (*A k*) ⟹
un (*prodrg I A*) = *prod-one I A*
⟨*proof*⟩

lemma *prodrg-sameTr5*:⟦∀ *k*∈*I*. *Ring* (*A k*); ∀ *k*∈*I*. *A k* = *B k*⟧
⟹ *prod-tOp I A* = *prod-tOp I B*
⟨*proof*⟩

lemma *prodrg-sameTr6*:⟦∀ *k*∈*I*. *Ring* (*A k*); ∀ *k*∈*I*. *A k* = *B k*⟧
⟹ *prod-one I A* = *prod-one I B*
⟨*proof*⟩

lemma *prodrg-same*:$[\![\forall\, k{\in}I.\ Ring\ (A\ k);\ \forall\, k{\in}I.\ A\ k = B\ k]\!]$
$\Longrightarrow$ *prodrg I A = prodrg I B*
⟨*proof*⟩

lemma *prodrg-component*:$[\![f \in carrier\ (prodrg\ I\ A);\ i \in I]\!] \Longrightarrow$
f i ∈ *carrier* (*A i*)
⟨*proof*⟩

lemma *project-rhom*:$[\![\forall\, k{\in}I.\ Ring\ (A\ k);\ j \in I]\!] \Longrightarrow$
PRoject I A j ∈ *rHom* (*prodrg I A*) (*A j*)
⟨*proof*⟩

lemma *augm-funcTr*:$[\![\forall\, k \leq (Suc\ n).\ Ring\ (B\ k);$
$f \in carr\text{-}prodag\ \{i.\ i \leq (Suc\ n)\}\ B]\!] \Longrightarrow$
f = *augm-func n* (*restrict f* {*i*. *i* ≤ *n*}) (*Un-carrier* {*i*. *i* ≤ *n*} *B*) (*Suc 0*)
(λ*x*∈{*0*::*nat*}. *f* (*x* + *Suc n*))
(*Un-carrier* {*0*} (*compose* {*0*} *B* (*slide* (*Suc n*))))
⟨*proof*⟩

lemma *A-to-prodag-mem*:$[\![Ring\ A;\ \forall\, k{\in}I.\ Ring\ (B\ k);\ \forall\, k{\in}I.\ (S\ k) \in$
rHom A (*B k*); *x* ∈ *carrier A* $]\!] \Longrightarrow$ *A-to-prodag A I S B x* ∈ *carr-prodag I B*
⟨*proof*⟩

lemma *A-to-prodag-rHom*:$[\![Ring\ A;\ \forall\, k{\in}I.\ Ring\ (B\ k);\ \forall\, k{\in}I.\ (S\ k) \in$
rHom A (*B k*) $]\!] \Longrightarrow$ *A-to-prodag A I S B* ∈ *rHom A* ($r\Pi_I$ *B*)
⟨*proof*⟩

lemma *ac-fProd-ProdTr1*:∀ *k* ≤ (*Suc n*). *Ring* (*B k*) $\Longrightarrow$
ag-setfunc n B (*Suc 0*) (*compose* {*0*::*nat*} *B* (*slide* (*Suc n*)))
(*carr-prodag* {*i*. *i* ≤ *n*} *B*) (*carr-prodag* {*0*}
(*compose* {*0*} *B* (*slide* (*Suc n*)))) ⊆ *carr-prodag* {*i*. *i* ≤ (*Suc n*)} *B*
⟨*proof*⟩

lemma *ac-fProd-Prod*:∀ *k* ≤ *n*. *Ring* (*B k*) $\Longrightarrow$
ac-fProd-Rg n B = *carr-prodag* {*j*. *j* ≤ *n*} *B*
⟨*proof*⟩

A direct product of a finite number of rings defined with *ac-fProd-Rg* is equal to that defined by using *carr-prodag*.

definition
fprodrg :: [*nat*, *nat* ⇒ ('*a*, '*more*) *Ring-scheme*] ⇒
(|*carrier*:: (*nat* ⇒ '*a*) *set*, *pop*::[(*nat* ⇒ '*a*), (*nat* ⇒ '*a*)]
⇒ (*nat* ⇒ '*a*), *mop*:: (*nat* ⇒ '*a*) ⇒ (*nat* ⇒ '*a*), *zero*::(*nat* ⇒ '*a*),
tp :: [(*nat* ⇒ '*a*), (*nat* ⇒ '*a*)] ⇒ (*nat* ⇒ '*a*), *un* :: (*nat* ⇒ '*a*) |) **where**

fprodrg n B = (| *carrier* = *ac-fProd-Rg n B*,
pop = λ*f*. λ*g*. *prod-pOp* {*i*. *i* ≤ *n*} *B f g*, *mop* = λ*f*. *prod-mOp* {*i*. *i* ≤ *n*} *B f*,
zero = *prod-zero* {*i*. *i* ≤ *n*} *B*, *tp* = λ*f*. λ*g*. *prod-tOp* {*i*. *i* ≤ *n*} *B f g*,

$un = prod\text{-}one\ \{i.\ i \leq n\}\ B\ \rangle\!\rangle$

definition
fPRoject ::[*nat*, *nat* ⇒ ('*a*, '*more*) *Ring-scheme*, *nat*]
⇒ (*nat* ⇒ '*a*) ⇒ '*a* **where**
fPRoject n B x = (λ*f*∈*ac-fProd-Rg n B*. *f x*)

lemma *fprodrg-ring*:∀ *k* ≤ *n*. *Ring* (*B k*) ⟹ *Ring* (*fprodrg n B*)
⟨*proof*⟩

4.8 Chinese remainder theorem

lemma *Chinese-remTr1*:⟦*Ring A*; ∀ *k* ≤ (*n*::*nat*). *ideal A* (*J k*);
∀ *k* ≤ *n*. *B k* = *qring A* (*J k*); ∀ *k* ≤ *n*. *S k* = *pj A* (*J k*) ⟧ ⟹
$ker_{A,(r\Pi_{\{j.\ j \leq n\}}\ B)}$ (*A-to-prodag A* {*j*. *j* ≤ *n*} *S B*) =
⋂ {*I*. ∃ *k*∈{*j*. *j* ≤ *n*}. *I* = (*J k*)}
⟨*proof*⟩

lemma (**in** *Ring*) *coprime-prod-int2Tr*:
((∀ *k* ≤ (*Suc n*). *ideal R* (*J k*)) ∧
(∀ *i* ≤ (*Suc n*). ∀ *j* ≤ (*Suc n*). (*i* ≠*j* ⟶ *coprime-ideals R* (*J i*) (*J j*))))
⟶ (⋂ {*I*. ∃ *k* ≤ (*Suc n*). *I* = (*J k*)} = *ideal-n-prod R* (*Suc n*) *J*)
⟨*proof*⟩

lemma (**in** *Ring*) *coprime-prod-int2*:⟦ ∀ *k* ≤ (*Suc n*). *ideal R* (*J k*);
∀ *i* ≤ (*Suc n*). ∀ *j* ≤ (*Suc n*). (*i* ≠*j* ⟶ *coprime-ideals R* (*J i*) (*J j*))⟧
⟹ (⋂ {*I*. ∃ *k* ≤ (*Suc n*). *I* = (*J k*)} = *ideal-n-prod R* (*Suc n*) *J*)
⟨*proof*⟩

lemma (**in** *Ring*) *coprime-2-n*:⟦*ideal R A*; *ideal R B*⟧ ⟹
(*qring R A*) $\bigoplus_r$ (*qring R B*) = $r\Pi_{\{j.\ j \leq (Suc\ 0)\}}$ (*prodB1* (*qring R A*) (*qring R B*))
⟨*proof*⟩

In this and following lemmata, ideals A and B are of type ('*a*, '*more*) *RingType-scheme*. Don't try ($r\Pi_{(Nset\ n)}$ *B*) $\bigoplus_r$ *B* (*Suc n*)

lemma (**in** *Ring*) *A-to-prodag2-hom*:⟦*ideal R A*; *ideal R B*; *S 0* = *pj R A*;
S (*Suc 0*) = *pj R B*⟧ ⟹
A-to-prodag R {*j*. *j* ≤ (*Suc 0*)} *S* (*prodB1* (*qring R A*) (*qring R B*)) ∈
rHom R (*qring R A* $\bigoplus_r$ *qring R B*)
⟨*proof*⟩

lemma (**in** *Ring*) *A2coprime-rsurjecTr*:⟦*ideal R A*; *ideal R B*; *S 0* = *pj R A*;
S (*Suc 0*) = *pj R B*⟧ ⟹
(*carrier* (*qring R A* $\bigoplus_r$ *qring R B*)) =
carr-prodag {*j*. *j* ≤ (*Suc 0*)} (*prodB1* (*qring R A*) (*qring R B*))
⟨*proof*⟩

lemma (**in** *Ring*) *A2coprime-rsurjec*:⟦*ideal R A*; *ideal R B*; *S 0* = *pj R A*;
S (*Suc 0*) = *pj R B*; *coprime-ideals R A B*⟧ ⟹
$surjec_{R,((qring\ R\ A) \bigoplus_r (qring\ R\ B))}$
(*A-to-prodag R* {*j*. *j*≤(*Suc 0*)} *S* (*prodB1* (*qring R A*) (*qring R B*)))
⟨*proof*⟩

lemma (**in** *Ring*) *prod2-n-Tr1*:⟦∀ *k* ≤ (*Suc 0*). *ideal R* (*J k*);
∀ *k* ≤ (*Suc 0*). *B k* = *qring R* (*J k*);
∀ *k* ≤ (*Suc 0*). *S k* = *pj R* (*J k*) ⟧ ⟹
A-to-prodag R {*j*. *j* ≤ (*Suc 0*)} *S*
(*prodB1* (*qring R* (*J 0*)) (*qring R* (*J* (*Suc 0*)))) =
A-to-prodag R {*j*. *j* ≤ (*Suc 0*)} *S B*
⟨*proof*⟩

lemma (**in** *aGroup*) *restrict-prod-Suc*:⟦∀ *k* ≤ (*Suc* (*Suc n*)). *ideal R* (*J k*);
∀ *k* ≤ (*Suc* (*Suc n*)). *B k* = *R* /$_r$ *J k*;
∀ *k* ≤ (*Suc* (*Suc n*)). *S k* = *pj R* (*J k*);
f ∈ *carrier* ($r\Pi_{\{j.\ j \le (Suc\ (Suc\ n))\}}$ *B*)⟧ ⟹
restrict f {*j*. *j* ≤ (*Suc n*)} ∈ *carrier* ($r\Pi_{\{j.\ j \le (Suc\ n)\}}$ *B*)
⟨*proof*⟩

lemma (**in** *Ring*) *Chinese-remTr2*:(∀ *k* ≤ (*Suc n*). *ideal R* (*J k*)) ∧
(∀ *k*≤(*Suc n*). *B k* = *qring R* (*J k*)) ∧
(∀ *i*≤(*Suc n*). ∀ *j*≤ (*Suc n*). (*i* ≠*j* ⟶
coprime-ideals R (*J i*) (*J j*))) ⟶
$surjec_{R,(r\Pi_{\{j.\ j \le (Suc\ n)\}}\ B)}$
(*A-to-prodag R* {*j*. *j*≤(*Suc n*)} *S B*)
⟨*proof*⟩

lemma (**in** *Ring*) *Chinese-remTr3*:⟦∀ *k* ≤ (*Suc n*). *ideal R* (*J k*);
∀ *k* ≤ (*Suc n*). *B k* = *qring R* (*J k*); ∀ *k*≤ (*Suc n*). *S k* = *pj R* (*J k*);
∀ *i* ≤ (*Suc n*). ∀ *j* ≤ (*Suc n*). (*i* ≠*j* ⟶ *coprime-ideals R* (*J i*) (*J j*))⟧ ⟹
$surjec_{R,(r\Pi_{\{j.\ j \le (Suc\ n)\}}\ B)}$
(*A-to-prodag R* {*j*. *j* ≤ (*Suc n*)} *S B*)
⟨*proof*⟩

lemma (**in** *Ring*) *imset*:⟦∀ *k*≤ (*Suc n*). *ideal R* (*J k*)⟧
⟹ {*I*. ∃ *k*≤ (*Suc n*). *I* = *J k*} = {*J k*| *k*. *k* ∈ {*j*. *j* ≤ (*Suc n*)}}
⟨*proof*⟩

theorem (**in** *Ring*) *Chinese-remThm*:⟦(∀ *k* ≤ (*Suc n*). *ideal R* (*J k*));
∀ *k*≤(*Suc n*). *B k* = *qring R* (*J k*); ∀ *k* ≤ (*Suc n*). *S k* = *pj R* (*J k*);
∀ *i* ≤ (*Suc n*). ∀ *j* ≤ (*Suc n*). (*i* ≠*j* ⟶ *coprime-ideals R* (*J i*) (*J j*))⟧
⟹ $bijec_{(qring\ R\ (\bigcap\ \{J\ k\ |\ k.\ k \in \{j.\ j \le (Suc\ n)\}\})),(r\Pi_{\{j.\ j \le (Suc\ n)\}}\ B)}$
$((A\text{-}to\text{-}prodag\ R\ \{j.\ j \le (Suc\ n)\}\ S\ B)^{\circ}{}_{R,(prodrg\ \{j.\ j \le (Suc\ n)\}\ B)})$
⟨*proof*⟩

lemma (**in** *Ring*) *prod-prime*:⟦*ideal R A*; ∀ *k*≤(*Suc n*). *prime-ideal R* (*P k*);
∀ *l*≤(*Suc n*). ¬ (*A* ⊆ *P l*);
∀ *k*≤ (*Suc n*). ∀ *l*≤ (*Suc n*). *k* = *l* ∨ ¬ (*P k*) ⊆ (*P l*)⟧ ⟹
∀ *i* ≤ (*Suc n*). (*nprod R* (*ppa R P A i*) *n* ∈ *A* ∧
(∀ *l* ∈ {*j*. *j*≤(*Suc n*)} − {*i*}. *nprod R* (*ppa R P A i*) *n* ∈ *P l*) ∧
(*nprod R* (*ppa R P A i*) *n* ∉ *P i*))
⟨*proof*⟩

lemma *skip-im1*:⟦*i* ≤ (*Suc n*); *P* ∈ {*j*. *j* ≤ (*Suc n*)} → *Collect* (*prime-ideal R*)⟧
⟹
compose {*j*. *j* ≤ *n*} *P* (*skip i*) ‘ {*j*. *j* ≤ *n*} = *P* ‘ ({*j*. *j* ≤ (*Suc n*)} − {*i*})
⟨*proof*⟩

lemma (**in** *Ring*) *mutch-aux1*:⟦*ideal R A*; *i* ≤ (*Suc n*);
P ∈ {*j*. *j* ≤ (*Suc n*)} → *Collect* (*prime-ideal R*)⟧ ⟹
compose {*j*. *j* ≤ *n*} *P* (*skip i*) ∈ {*j*. *j* ≤ *n*} → *Collect* (*prime-ideal R*)
⟨*proof*⟩

lemma (**in** *Ring*) *prime-ideal-cont1Tr*:*ideal R A* ⟹
∀ *P*. ((*P* ∈ {*j*. *j* ≤ (*n*::*nat*)} → {*X*. *prime-ideal R X*}) ∧
(*A* ⊆ ⋃ (*P* ‘ {*j*. *j* ≤ *n*}))) ⟶ (∃ *i*≤ *n*. *A* ⊆ (*P i*))
⟨*proof*⟩

lemma (**in** *Ring*) *prime-ideal-cont1*:⟦*ideal R A*; ∀ *i* ≤ (*n*::*nat*).
prime-ideal R (*P i*); *A* ⊆ ⋃ {*X*. (∃ *i* ≤ *n*. *X* = (*P i*))} ⟧ ⟹
∃ *i*≤ *n*. *A*⊆(*P i*)
⟨*proof*⟩

lemma (**in** *Ring*) *prod-n-ideal-contTr0*:(∀ *l*≤ *n*. *ideal R* (*J l*)) ⟶
*i*Π$_{R,n}$ *J* ⊆ ⋂{*X*. (∃ *k*≤*n*. *X* = (*J k*))}
⟨*proof*⟩

lemma (**in** *Ring*) *prod-n-ideal-contTr*:⟦∀ *l*≤ *n*. *ideal R* (*J l*)⟧ ⟹
*i*Π$_{R,n}$ *J* ⊆ ⋂{*X*. (∃ *k* ≤ *n*. *X* = (*J k*))}
⟨*proof*⟩

lemma (**in** *Ring*) *prod-n-ideal-cont2*:⟦∀ *l*≤ (*n*::*nat*). *ideal R* (*J l*);
prime-ideal R P; ⋂{*X*. (∃ *k*≤ *n*. *X* = (*J k*))} ⊆ *P*⟧ ⟹
∃ *l*≤ *n*. (*J l*) ⊆ *P*
⟨*proof*⟩

lemma (**in** *Ring*) *prod-n-ideal-cont3*:⟦∀ *l*≤ (*n*::*nat*). *ideal R* (*J l*);
prime-ideal R P; ⋂{*X*. (∃ *k*≤ *n*. *X* = (*J k*))} = *P*⟧ ⟹
∃ *l*≤ *n*. (*J l*) = *P*
⟨*proof*⟩

definition
ideal-quotient :: [- , *'a set*, *'a set*] ⇒ *'a set* **where**

ideal-quotient R A B = $\{x|\ x.\ x \in carrier\ R \wedge (\forall\, b{\in}B.\ x \cdot_{r\,R}\ b \in A)\}$

abbreviation
IDEALQT ($\langle$(*3-/ †-/ -*)$\rangle$ [*82,82,83*]*82*) **where**
$A \dagger_R B$ == *ideal-quotient R A B*

lemma (**in** *Ring*) *ideal-quotient-is-ideal*:
$[\![$*ideal R A*; *ideal R B*$]\!] \Longrightarrow$ *ideal R* (*ideal-quotient R A B*)
⟨*proof*⟩

4.9 Addition of finite elements of a ring and *ideal-multiplication*

We consider sum in an abelian group

lemma (**in** *aGroup*) *nsum-mem1Tr*: *A +> J* $\Longrightarrow$
$(\forall j \le n.\ f\ j \in J) \longrightarrow$ *nsum A f n* $\in$ *J*
⟨*proof*⟩

lemma (**in** *aGroup*) *fSum-mem*:$[\![\forall j \in nset\ (Suc\ n)\ m.\ f\ j \in carrier\ A;\ n < m]\!]$
$\Longrightarrow$
fSum A f (*Suc n*) *m* $\in$ *carrier A*
⟨*proof*⟩

lemma (**in** *aGroup*) *nsum-mem1*:$[\![A +> J; \forall j \le n.\ f\ j \in J]\!] \Longrightarrow$ *nsum A f n* $\in$ *J*
⟨*proof*⟩

lemma (**in** *aGroup*) *nsum-eq-i*:$[\![\forall j{\le}n.\ f\ j \in carrier\ A; \forall j{\le}n.\ g\ j \in carrier\ A;$
$i \le n; \forall l \le i.\ f\ l = g\ l]\!] \Longrightarrow$ *nsum A f i* = *nsum A g i*
⟨*proof*⟩

lemma (**in** *aGroup*) *nsum-cmp-eq*:$[\![f \in \{j.\ j{\le}(n::nat)\} \to carrier\ A;$
h1 $\in \{j.\ j \le n\} \to \{j.\ j \le n\}$; *h2* $\in \{j.\ j \le n\} \to \{j.\ j \le n\}$; $i \le n]\!] \Longrightarrow$
nsum A (*cmp f* (*cmp h2 h1*)) *i* = *nsum A* (*cmp* (*cmp f h2*) *h1*) *i*
⟨*proof*⟩

lemma (**in** *aGroup*) *nsum-cmp-eq-transpos*:$[\![\ \forall j{\le}(Suc\ n).\ f\ j \in carrier\ A;$
$i \le n; i \ne n\ ]\!] \Longrightarrow$
nsum A (*cmp f* (*cmp* (*transpos i n*) (*cmp* (*transpos n* (*Suc n*)) (*transpos i n*))))
(*Suc n*) = *nsum A* (*cmp f* (*transpos i* (*Suc n*))) (*Suc n*)
⟨*proof*⟩

lemma *transpos-Tr-n1*:*Suc* (*Suc 0*) $\le n \Longrightarrow$
transpos (n − *Suc 0*) n n = n − *Suc 0*
⟨*proof*⟩

lemma *transpos-Tr-n2*:*Suc* (*Suc 0*) $\le n \Longrightarrow$
transpos (n − (*Suc 0*)) n (n − (*Suc 0*)) = n
⟨*proof*⟩

lemma (**in** *aGroup*) *additionTr0*:⟦$0 < n$; $\forall j \leq n.\ f\ j \in$ *carrier* A⟧
$\Longrightarrow$ *nsum* A (*cmp* f (*transpos* $(n - 1)$ n)) n = *nsum* A f n
⟨*proof*⟩

lemma (**in** *aGroup*) *additionTr1*:⟦ $\forall f.\ \forall h.\ f \in \{j.\ j \leq (Suc\ n)\} \to$ *carrier* $A \wedge$
$h \in \{j.\ j \leq (Suc\ n)\} \to \{j.\ j \leq (Suc\ n)\} \wedge$ *inj-on* h $\{j.\ j \leq (Suc\ n)\} \longrightarrow$
nsum A (*cmp* f h) $(Suc\ n)$ = *nsum* A f $(Suc\ n)$;
$f \in \{j.\ j \leq (Suc\ (Suc\ n))\} \to$ *carrier* A;
$h \in \{j.\ j \leq (Suc\ (Suc\ n))\} \to \{j.\ j \leq (Suc\ (Suc\ n))\}$;
inj-on h $\{j.\ j \leq (Suc\ (Suc\ n))\}$; $h\ (Suc\ (Suc\ n)) = Suc\ (Suc\ n)$⟧
$\Longrightarrow$ *nsum* A (*cmp* f h) $(Suc\ (Suc\ n))$ = *nsum* A f $(Suc\ (Suc\ n))$
⟨*proof*⟩

lemma (**in** *aGroup*) *additionTr1-1*:⟦$\forall f.\ \forall h.\ f \in \{j.\ j \leq Suc\ n\} \to$ *carrier* $A \wedge$
$h \in \{j.\ j \leq Suc\ n\} \to \{j.\ j \leq Suc\ n\} \wedge$ *inj-on* h $\{j.\ j \leq Suc\ n\} \longrightarrow$
nsum A (*cmp* f h) $(Suc\ n)$ = *nsum* A f $(Suc\ n)$;
$f \in \{j.\ j \leq Suc\ (Suc\ n)\} \to$ *carrier* A; $i \leq n$⟧ $\Longrightarrow$
nsum A (*cmp* f (*transpos* i $(Suc\ n)$)) $(Suc\ (Suc\ n))$ = *nsum* A f $(Suc\ (Suc\ n))$
⟨*proof*⟩

lemma (**in** *aGroup*) *additionTr1-2*:⟦$\forall f.\ \forall h.\ f \in \{j.\ j \leq Suc\ n\} \to$ *carrier* $A \wedge$
$h \in \{j.\ j \leq Suc\ n\} \to \{j.\ j \leq Suc\ n\} \wedge$
inj-on h $\{j.\ j \leq Suc\ n\} \longrightarrow$
nsum A (*cmp* f h) $(Suc\ n)$ = *nsum* A f $(Suc\ n)$;
$f \in \{j.\ j \leq Suc\ (Suc\ n)\} \to$ *carrier* A; $i \leq (Suc\ n)$⟧ $\Longrightarrow$
nsum A (*cmp* f (*transpos* i $(Suc\ (Suc\ n))$)) $(Suc\ (Suc\ n))$ =
nsum A f $(Suc\ (Suc\ n))$
⟨*proof*⟩

lemma (**in** *aGroup*) *additionTr2*: $\forall f.\ \forall h.\ f \in \{j.\ j \leq (Suc\ n)\} \to$ *carrier* $A \wedge$
$h \in \{j.\ j \leq (Suc\ n)\} \to \{j.\ j \leq (Suc\ n)\} \wedge$
inj-on h $\{j.\ j \leq (Suc\ n)\} \longrightarrow$
nsum A (*cmp* f h) $(Suc\ n)$ = *nsum* A f $(Suc\ n)$
⟨*proof*⟩

lemma (**in** *aGroup*) *addition2*:⟦$f \in \{j.\ j \leq (Suc\ n)\} \to$ *carrier* A;
$h \in \{j.\ j \leq (Suc\ n)\} \to \{j.\ j \leq (Suc\ n)\}$; *inj-on* h $\{j.\ j \leq (Suc\ n)\}$⟧ $\Longrightarrow$
nsum A (*cmp* f h) $(Suc\ n)$ = *nsum* A f $(Suc\ n)$
⟨*proof*⟩

lemma (**in** *aGroup*) *addition21*:⟦$f \in \{j.\ j \leq n\} \to$ *carrier* A;
$h \in \{j.\ j \leq n\} \to \{j.\ j \leq n\}$; *inj-on* h $\{j.\ j \leq n\}$⟧ $\Longrightarrow$
nsum A (*cmp* f h) n = *nsum* A f n
⟨*proof*⟩

lemma (**in** *aGroup*) *addition3*:⟦$\forall j \leq (Suc\ n).\ f\ j \in$ *carrier* A; $j \leq (Suc\ n)$;
$j \neq Suc\ n$⟧ $\Longrightarrow$ *nsum* A f $(Suc\ n)$ = *nsum* A (*cmp* f (*transpos* j $(Suc\ n)$)) $(Suc\ n)$
⟨*proof*⟩

lemma (**in** *aGroup*) *nsum-splitTr*:$(\forall\, j \leq (Suc\ (n + m)).\ f\, j \in carrier\ A) \longrightarrow$
$nsum\ A\ f\ (Suc\ (n + m)) = nsum\ A\ f\ n \pm (nsum\ A\ (cmp\ f\ (slide\ (Suc\ n)))\ m)$
⟨*proof*⟩

lemma (**in** *aGroup*) *nsum-split*:$\forall\, j \leq (Suc\ (n + m)).\ f\, j \in carrier\ A \Longrightarrow$
$nsum\ A\ f\ (Suc\ (n + m)) = nsum\ A\ f\ n \pm (nsum\ A\ (cmp\ f\ (slide\ (Suc\ n)))\ m)$

⟨*proof*⟩

lemma (**in** *aGroup*) *nsum-split1*:$[\![\forall\, j \leq m.\ f\, j \in carrier\ A;\ n < m]\!] \Longrightarrow$
$nsum\ A\ f\ m = nsum\ A\ f\ n \pm (fSum\ A\ f\ (Suc\ n)\ m)$
⟨*proof*⟩

lemma (**in** *aGroup*) *nsum-minusTr*: $(\forall\, j \leq n.\ f\, j \in carrier\ A) \longrightarrow$
$-_a\ (nsum\ A\ f\ n) = nsum\ A\ (\lambda x{\in}\{j.\ j \leq n\}.\ -_a\ (f\, x))\ n$
⟨*proof*⟩

lemma (**in** *aGroup*) *nsum-minus*:$\forall\, j \leq n.\ f\, j \in carrier\ A \Longrightarrow$
$-_a\ (nsum\ A\ f\ n) = nsum\ A\ (\lambda x{\in}\{j.\ j \leq n\}.\ -_a\ (f\, x))\ n$
⟨*proof*⟩

lemma (**in** *aGroup*) *ring-nsum-zeroTr*:$(\forall\, j \leq (n{::}nat).\ f\, j \in carrier\ A) \wedge$
$(\forall\, j \leq n.\ f\, j = \mathbf{0}) \longrightarrow nsum\ A\ f\ n = \mathbf{0}$
⟨*proof*⟩

lemma (**in** *aGroup*) *ring-nsum-zero*:$\forall\, j \leq (n{::}nat).\ f\, j = \mathbf{0} \implies \Sigma_e\ A\ f\ n = \mathbf{0}$
⟨*proof*⟩

lemma (**in** *aGroup*) *ag-nsum-1-nonzeroTr*:
$\forall f.\ (\forall\, j \leq n.\ f\, j \in carrier\ A) \wedge$
$(l \leq n \wedge (\forall\, j \in \{j.\ j \leq n\} - \{l\}.\ f\, j = \mathbf{0}))$
$\longrightarrow nsum\ A\ f\ n = f\ l$
⟨*proof*⟩

lemma (**in** *aGroup*) *ag-nsum-1-nonzero*:$[\![\forall\, j \leq n.\ f\, j \in carrier\ A;\ l \leq n;$
$\forall\, j{\in}(\{j.\ j \leq n\} - \{l\}).\ f\, j = \mathbf{0}\ ]\!] \Longrightarrow nsum\ A\ f\ n = f\ l$
⟨*proof*⟩

definition
set-mult :: [- , *'a set*, *'a set*] ⇒ *'a set* **where**
set-mult R A B = $\{z.\ \exists\, x{\in}A.\ \exists\, y{\in}B.\ \ x \cdot_{r\,R} y = z\}$

definition
sum-mult :: [- , *'a set*, *'a set*] ⇒ *'a set* **where**
sum-mult R A B = $\{x.\ \exists\, n.\ \exists f \in \{j.\ j \leq (n{::}nat)\}$
$\to$ *set-mult R A B. nsum R f n* = $x\}$

lemma (**in** *Ring*) *set-mult-sub*:$[\![A \subseteq carrier\ R;\ B \subseteq carrier\ R]\!] \Longrightarrow$
$set\text{-}mult\ R\ A\ B \subseteq carrier\ R$
⟨*proof*⟩

lemma (**in** *Ring*) *set-mult-mono*:$[\![A1 \subseteq carrier\ R;\ A2 \subseteq carrier\ R;\ A1 \subseteq A2;$
$B \subseteq carrier\ R]\!] \Longrightarrow set\text{-}mult\ R\ A1\ B \subseteq set\text{-}mult\ R\ A2\ B$
⟨*proof*⟩

lemma (**in** *Ring*) *sum-mult-Tr1*:$[\![A \subseteq carrier\ R;\ B \subseteq carrier\ R]\!] \Longrightarrow$
$(\forall j \leq n.\ f\ j \in set\text{-}mult\ R\ A\ B) \longrightarrow nsum\ R\ f\ n \in carrier\ R$
⟨*proof*⟩

lemma (**in** *Ring*) *sum-mult-mem*:$[\![A \subseteq carrier\ R;\ B \subseteq carrier\ R;$
$\forall j \leq n.\ f\ j \in set\text{-}mult\ R\ A\ B]\!] \Longrightarrow nsum\ R\ f\ n \in carrier\ R$
⟨*proof*⟩

lemma (**in** *Ring*) *sum-mult-mem1*:$[\![A \subseteq carrier\ R;\ B \subseteq carrier\ R;$
$x \in sum\text{-}mult\ R\ A\ B]\!] \Longrightarrow$
$\exists n.\ \exists f \in \{j.\ j \leq (n::nat)\} \to set\text{-}mult\ R\ A\ B.\ nsum\ R\ f\ n = x$
⟨*proof*⟩

lemma (**in** *Ring*) *sum-mult-subR*:$[\![A \subseteq carrier\ R;\ B \subseteq carrier\ R]\!] \Longrightarrow$
$sum\text{-}mult\ R\ A\ B \subseteq carrier\ R$
⟨*proof*⟩

lemma (**in** *Ring*) *times-mem-sum-mult*:$[\![A \subseteq carrier\ R;\ B \subseteq carrier\ R;$
$a \in A;\ b \in B\ ]\!] \Longrightarrow\ a \cdot_r b \in sum\text{-}mult\ R\ A\ B$
⟨*proof*⟩

lemma (**in** *Ring*) *mem-minus-sum-multTr2*:$[\![A \subseteq carrier\ R;\ B \subseteq carrier\ R;$
$\forall j \leq n.\ f\ j \in set\text{-}mult\ R\ A\ B;\ i \leq n]\!] \Longrightarrow f\ i \in carrier\ R$
⟨*proof*⟩

lemma (**in** *aGroup*) *nsum-jointfun*:$[\![\forall j \leq n.\ f\ j \in carrier\ A;$
$\forall j \leq m.\ g\ j \in carrier\ A]\!] \Longrightarrow$
$\Sigma_e\ A\ (jointfun\ n\ f\ m\ g)\ (Suc\ (n + m)) = \Sigma_e\ A\ f\ n \pm (\Sigma_e\ A\ g\ m)$
⟨*proof*⟩

lemma (**in** *Ring*) *sum-mult-pOp-closed*:$[\![A \subseteq carrier\ R;\ B \subseteq carrier\ R;$
$a \in sum\text{-}mult\ R\ A\ B;\ \ b \in sum\text{-}mult\ R\ A\ B\ ]\!] \Longrightarrow a \pm_R b \in sum\text{-}mult\ R\ A$
B
⟨*proof*⟩

lemma (**in** *Ring*) *set-mult-mOp-closed*:$[\![A \subseteq carrier\ R;\ ideal\ R\ B;$
$x \in set\text{-}mult\ R\ A\ B]\!] \Longrightarrow -_a\ x \in set\text{-}mult\ R\ A\ B$
⟨*proof*⟩

lemma (**in** *Ring*) *set-mult-ring-times-closed*:$[\![A \subseteq carrier\ R;\ ideal\ R\ B;$
$x \in set\text{-}mult\ R\ A\ B;\ r \in carrier\ R]\!] \Longrightarrow r \cdot_r\ \ x \in set\text{-}mult\ R\ A\ B$

⟨*proof*⟩

lemma (**in** *Ring*) *set-mult-sub-sum-mult*:⟦$A \subseteq$ *carrier* R; *ideal* R B⟧ ⟹
set-mult R A $B \subseteq$ *sum-mult* R A B
⟨*proof*⟩

lemma (**in** *Ring*) *sum-mult-pOp-closedn*:⟦$A \subseteq$ *carrier* R; *ideal* R B⟧ ⟹
$(\forall j \leq n.\ f\ j \in$ *set-mult* R A $B) \longrightarrow \Sigma_e$ R f $n \in$ *sum-mult* R A B
⟨*proof*⟩

lemma (**in** *Ring*) *mem-minus-sum-multTr4*:⟦$A \subseteq$ *carrier* R; *ideal* R B⟧ ⟹
$(\forall j \leq n.\ f\ j \in$ *set-mult* R A $B) \longrightarrow -_a$ (*nsum* R f n) $\in$ *sum-mult* R A B
⟨*proof*⟩

lemma (**in** *Ring*) *sum-mult-iOp-closed*:⟦$A \subseteq$ *carrier* R; *ideal* R B;
$x \in$ *sum-mult* R A B ⟧ ⟹ $-_a\ x \in$ *sum-mult* R A B
⟨*proof*⟩

lemma (**in** *Ring*) *sum-mult-ring-multiplicationTr*:
⟦$A \subseteq$ *carrier* R; *ideal* R B; $r \in$ *carrier* R⟧ ⟹
$(\forall j \leq n.\ f\ j \in$ *set-mult* R A $B) \longrightarrow r \cdot_r$ (*nsum* R f n) $\in$ *sum-mult* R A B
⟨*proof*⟩

lemma (**in** *Ring*) *sum-mult-ring-multiplication*:⟦$A \subseteq$ *carrier* R; *ideal* R B;
$r \in$ *carrier* R; $a \in$ *sum-mult* R A B⟧ ⟹ $r \cdot_r a \in$ *sum-mult* R A B
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-sum-mult*:⟦$A \subseteq$ *carrier* R; $A \neq \{\}$; *ideal* R B⟧ ⟹
ideal R (*sum-mult* R A B)
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-inc-set-multTr*:⟦$A \subseteq$ *carrier* R; *ideal* R B; *ideal* R C;
set-mult R A $B \subseteq C$ ⟧ ⟹
$\forall f \in \{j.\ j \leq (n::nat)\} \to$ *set-mult* R A B. Σ_e R f $n \in C$
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-inc-set-mult*:⟦$A \subseteq$ *carrier* R; *ideal* R B; *ideal* R C;
set-mult R A $B \subseteq C$ ⟧ ⟹ *sum-mult* R A $B \subseteq C$
⟨*proof*⟩

lemma (**in** *Ring*) *AB-inc-sum-mult*:⟦*ideal* R A; *ideal* R B⟧ ⟹
sum-mult R A $B \subseteq A \cap B$
⟨*proof*⟩

lemma (**in** *Ring*) *sum-mult-is-ideal-prod*:⟦*ideal* R A; *ideal* R B⟧ ⟹
sum-mult R A $B =\ A \diamondsuit_r B$
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-prod-assocTr0*:⟦*ideal* R A; *ideal* R B; *ideal* R C; $y \in C$;

$$z \in \textit{set-mult}\ R\ A\ B\rrbracket \Longrightarrow z \cdot_r y \in \textit{sum-mult}\ R\ A\ (B \diamondsuit_r C)$$

⟨*proof*⟩

lemma (**in** *Ring*) *ideal-prod-assocTr1*:⟦*ideal R A*; *ideal R B*; *ideal R C*; $y \in C$⟧
$\Longrightarrow \forall f \in \{j.\ j \leq (n::nat)\} \rightarrow \textit{set-mult}\ R\ A\ B.\ (\Sigma_e\ R\ f\ n) \cdot_r y \in A \diamondsuit_r (B \diamondsuit_r C)$
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-quotient-idealTr*:⟦*ideal R A*; *ideal R B*; *ideal R C*;
$x \in \textit{carrier}\ R; \forall c \in C.\ x \cdot_r c \in \textit{ideal-quotient}\ R\ A\ B\rrbracket \Longrightarrow$
$f \in \{j.\ j \leq n\} \rightarrow \textit{set-mult}\ R\ B\ C \longrightarrow x \cdot_r (\textit{nsum}\ R\ f\ n) \in A$

⟨*proof*⟩

lemma (**in** *Ring*) *ideal-quotient-ideal*:⟦*ideal R A*; *ideal R B*; *ideal R C*⟧ $\Longrightarrow$
$A \dagger_R B \dagger_R C = A \dagger_R B \diamondsuit_r C$
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-prod-assocTr*:⟦*ideal R A*; *ideal R B*; *ideal R C*⟧ $\Longrightarrow$
$\forall f.\ (f \in \{j.\ j \leq (n::nat)\} \rightarrow \textit{set-mult}\ R\ (A \diamondsuit_r B)\ C \longrightarrow$
$(\Sigma_e\ R\ f\ n) \in A \diamondsuit_r (B \diamondsuit_r C))$
⟨*proof*⟩

lemma (**in** *Ring*) *ideal-prod-assoc*:⟦*ideal R A*; *ideal R B*; *ideal R C*⟧ $\Longrightarrow$
$(A \diamondsuit_r B) \diamondsuit_r C = A \diamondsuit_r (B \diamondsuit_r C)$
⟨*proof*⟩

lemma (**in** *Ring*) *prod-principal-idealTr0*: ⟦$a \in \textit{carrier}\ R$; $b \in \textit{carrier}\ R$;
$z \in \textit{set-mult}\ R\ (R \diamondsuit_p a)\ (R \diamondsuit_p b)\rrbracket \Longrightarrow z \in R \diamondsuit_p (a \cdot_r b)$
⟨*proof*⟩

lemma (**in** *Ring*) *prod-principal-idealTr1*: ⟦$a \in \textit{carrier}\ R$; $b \in \textit{carrier}\ R\rrbracket \Longrightarrow$
$\forall f \in \{j.\ j \leq (n::nat)\} \rightarrow \textit{set-mult}\ R\ (R \diamondsuit_p a)\ (R \diamondsuit_p b).$
$\Sigma_e\ R\ f\ n \in R \diamondsuit_p (a \cdot_r b)$
⟨*proof*⟩

lemma (**in** *Ring*) *prod-principal-ideal*:⟦$a \in \textit{carrier}\ R$; $b \in \textit{carrier}\ R\rrbracket \Longrightarrow$
$(Rxa\ R\ a) \diamondsuit_r (Rxa\ R\ b) = Rxa\ R\ (a \cdot_r b)$
⟨*proof*⟩

lemma (**in** *Ring*) *principal-ideal-n-pow1*:$a \in \textit{carrier}\ R \Longrightarrow$
$(Rxa\ R\ a)^{\diamondsuit R\ n} = Rxa\ R\ (a^{\frown R\ n})$
⟨*proof*⟩

lemma (**in** *Ring*) *principal-ideal-n-pow*:⟦$a \in \textit{carrier}\ R$; $I = Rxa\ R\ a\rrbracket \Longrightarrow$
$I^{\diamondsuit R\ n} = Rxa\ R\ (a^{\frown R\ n})$
⟨*proof*⟩

more about *ideal-n-prod*

lemma (**in** *Ring*) *nprod-eqTr*: $f \in \{j.\ j \leq (n::nat)\} \to carrier\ R \wedge$
$g \in \{j.\ j \leq n\} \to carrier\ R \wedge (\forall j \leq n.\ f\ j = g\ j) \longrightarrow$
$nprod\ R\ f\ n = nprod\ R\ g\ n$
⟨*proof*⟩

lemma (**in** *Ring*) *nprod-eq*:$[\![\forall j \leq n.\ f\ j \in carrier\ R;\ \forall j \leq n.\ g\ j \in carrier\ R;$
$(\forall j \leq (n::nat).\ f\ j = g\ j)]\!] \Longrightarrow nprod\ R\ f\ n = nprod\ R\ g\ n$
⟨*proof*⟩

definition
mprod-expR :: [('*b*, '*m*) *Ring-scheme*, *nat* ⇒ *nat*, *nat* ⇒ '*b*, *nat*] ⇒ '*b* **where**
$mprod\text{-}expR\ R\ e\ f\ n = nprod\ R\ (\lambda j.\ ((f\ j)^{\frown R\ (e\ j)}))\ n$

lemma (**in** *Ring*) *mprodR-Suc*:$[\![e \in \{j.\ j \leq (Suc\ n)\} \to \{j.\ (0::nat) \leq j\};$
$f \in \{j.\ j \leq (Suc\ n)\} \to carrier\ R]\!] \Longrightarrow$
$mprod\text{-}expR\ R\ e\ f\ (Suc\ n) =$
$(mprod\text{-}expR\ R\ e\ f\ n) \cdot_r ((f\ (Suc\ n))^{\frown R\ (e\ (Suc\ n))})$
⟨*proof*⟩

lemma (**in** *Ring*) *mprod-expR-memTr*:$e \in \{j.\ j \leq n\} \to \{j.\ (0::nat) \leq j\} \wedge$
$f \in \{j.\ j \leq n\} \to carrier\ R \longrightarrow mprod\text{-}expR\ R\ e\ f\ n \in carrier\ R$
⟨*proof*⟩

lemma (**in** *Ring*) *mprod-expR-mem*:$[\![\ e \in \{j.\ j \leq n\} \to \{j.\ (0::nat) \leq j\};$
$f \in \{j.\ j \leq n\} \to carrier\ R]\!] \Longrightarrow mprod\text{-}expR\ R\ e\ f\ n \in carrier\ R$
⟨*proof*⟩

lemma (**in** *Ring*) *prod-n-principal-idealTr*:$e \in \{j.\ j{\leq}n\} \to \{j.\ (0::nat){\leq}j\} \wedge$
$f \in \{j.\ j{\leq}n\} \to carrier\ R \wedge (\forall k \leq n.\ J\ k = (Rxa\ R\ (f\ k))^{\diamondsuit R\ (e\ k)}) \longrightarrow$
$ideal\text{-}n\text{-}prod\ R\ n\ J = Rxa\ R\ (mprod\text{-}expR\ R\ e\ f\ n)$
⟨*proof*⟩

lemma (**in** *Ring*) *prod-n-principal-ideal*:$[\![e \in \{j.\ j{\leq}n\} \to \{j.\ (0::nat){\leq}j\};$
$f \in \{j.\ j{\leq}n\} \to carrier\ R; \forall k{\leq}\ n.\ J\ k = (Rxa\ R\ (f\ k))^{\diamondsuit R\ (e\ k)}]\!] \Longrightarrow$
$ideal\text{-}n\text{-}prod\ R\ n\ J = Rxa\ R\ (mprod\text{-}expR\ R\ e\ f\ n)$
⟨*proof*⟩

lemma (**in** *Idomain*) *a-notin-n-pow1*:$[\![a \in carrier\ R;\ \neg\ Unit\ R\ a;\ a \neq \mathbf{0};\ 0 < n]\!]$
$\Longrightarrow\ a \notin (Rxa\ R\ a)^{\diamondsuit R\ \ (Suc\ n)}$
⟨*proof*⟩

lemma (**in** *Idomain*) *a-notin-n-pow2*:$[\![a \in carrier\ R;\ \neg\ Unit\ R\ a;\ a \neq \mathbf{0};$
$0 < n]\!] \Longrightarrow a^{\frown R\ n} \notin (Rxa\ R\ a)^{\diamondsuit R\ (Suc\ n)}$
⟨*proof*⟩

lemma (**in** *Idomain*) *n-pow-not-prime*:$[\![a \in carrier\ R;\ a \neq \mathbf{0};\ \ 0 < n]\!]$
$\Longrightarrow\ \ \neg\ prime\text{-}ideal\ R\ ((Rxa\ R\ a)\ ^{\diamondsuit R\ (Suc\ n)})$
⟨*proof*⟩

lemma (**in** *Idomain*) *principal-pow-prime-condTr*:
$[\![a \in carrier\ R;\ a \neq \mathbf{0};\ prime\text{-}ideal\ R\ ((Rxa\ R\ a)\ ^{\diamondsuit R\ (Suc\ n)})]\!] \Longrightarrow n = 0$
⟨*proof*⟩

lemma (**in** *Idomain*) *principal-pow-prime-cond*:
$[\![a \in carrier\ R;\ a \neq \mathbf{0};\ \ prime\text{-}ideal\ R\ ((Rxa\ R\ a)\ ^{\diamondsuit R\ n})]\!] \Longrightarrow n = Suc\ 0$
⟨*proof*⟩

4.10 Extension and contraction

locale *TwoRings* = *Ring* +
fixes R' (**structure**)
assumes *secondR*: *Ring* R'

definition
i-contract :: [$'a \Rightarrow 'b$, ($'a$, *'m1*) *Ring-scheme*, ($'b$, *'m2*) *Ring-scheme*,
$'b$ *set*] $\Rightarrow$ $'a$ *set* **where**
i-contract f R R' J = *invim f* (*carrier R*) *J*

definition
i-extension :: [$'a \Rightarrow 'b$, ($'a$, *'m1*) *Ring-scheme*, ($'b$, *'m2*) *Ring-scheme*,
$'a$ *set*] $\Rightarrow$ $'b$ *set* **where**
i-extension f R R' I = *sum-mult R'* (*f* ' *I*) (*carrier R'*)

lemma (**in** *TwoRings*) *i-contract-sub*:$[\![f \in rHom\ R\ R';\ ideal\ R'\ J\ ]\!] \Longrightarrow$
(*i-contract f R R' J*) $\subseteq$ *carrier R*
⟨*proof*⟩

lemma (**in** *TwoRings*) *i-contract-ideal*:$[\![f \in rHom\ R\ R';\ ideal\ R'\ J\ ]\!] \Longrightarrow$
ideal R (*i-contract f R R' J*)
⟨*proof*⟩

lemma (**in** *TwoRings*) *i-contract-mono*:$[\![f \in rHom\ R\ R';\ ideal\ R'\ J1;\ ideal\ R'\ J2;$
J1 $\subseteq$ *J2* $]\!] \Longrightarrow$ *i-contract f R R' J1* $\subseteq$ *i-contract f R R' J2*
⟨*proof*⟩

lemma (**in** *TwoRings*) *i-contract-prime*:$[\![f \in rHom\ R\ R';\ prime\text{-}ideal\ R'\ P]\!] \Longrightarrow$
prime-ideal R (*i-contract f R R' P*)
⟨*proof*⟩

lemma (**in** *TwoRings*) *i-extension-ideal*:$[\![f \in rHom\ R\ R';\ ideal\ R\ I\ ]\!] \Longrightarrow$
ideal R' (*i-extension f R R' I*)

⟨*proof*⟩

lemma (**in** *TwoRings*) *i-extension-mono*:⟦$f \in rHom\ R\ R'$; *ideal* $R\ I1$; *ideal* $R\ I2$; $I1 \subseteq I2$ ⟧ ⟹ (*i-extension* $f\ R\ R'\ I1$) ⊆ (*i-extension* $f\ R\ R'\ I2$)
⟨*proof*⟩

lemma (**in** *TwoRings*) *e-c-inc-self*:⟦$f \in rHom\ R\ R'$; *ideal* $R\ I$⟧ ⟹
$I \subseteq$ *i-contract* $f\ R\ R'$ (*i-extension* $f\ R\ R'\ I$)
⟨*proof*⟩

lemma (**in** *TwoRings*) *c-e-incd-self*:⟦$f \in rHom\ R\ R'$; *ideal* $R'\ J$ ⟧ ⟹
i-extension $f\ R\ R'$ (*i-contract* $f\ R\ R'\ J$) ⊆ J
⟨*proof*⟩

lemma (**in** *TwoRings*) *c-e-c-eq-c*:⟦$f \in rHom\ R\ R'$; *ideal* $R'\ J$ ⟧ ⟹
i-contract $f\ R\ R'$ (*i-extension* $f\ R\ R'$ (*i-contract* $f\ R\ R'\ J$))
= *i-contract* $f\ R\ R'\ J$
⟨*proof*⟩

lemma (**in** *TwoRings*) *e-c-e-eq-e*:⟦$f \in rHom\ R\ R'$; *ideal* $R\ I$ ⟧ ⟹
i-extension $f\ R\ R'$ (*i-contract* $f\ R\ R'$ (*i-extension* $f\ R\ R'\ I$))
= *i-extension* $f\ R\ R'\ I$
⟨*proof*⟩

4.11 Complete system of representatives

definition
csrp-fn :: [-, *'a set*] ⇒ *'a set* ⇒ *'a* **where**
csrp-fn $R\ I$ = (λx∈*carrier* (R /$_r$ I). (*if* $x = I$ *then* $\mathbf{0}_R$ *else SOME* y. $y \in x$))

definition
csrp :: [- , *'a set*] ⇒ *'a set* **where**
csrp $R\ I$ == (*csrp-fn* $R\ I$) ‘ (*carrier* (R /$_r$ I))

lemma (**in** *Ring*) *csrp-mem*:⟦*ideal* $R\ I$; $a \in$ *carrier* R⟧ ⟹
csrp-fn $R\ I$ ($a \uplus_R I$) ∈ $a \uplus_R I$
⟨*proof*⟩

lemma (**in** *Ring*) *csrp-same*:⟦*ideal* $R\ I$; $a \in$ *carrier* R⟧ ⟹
csrp-fn $R\ I$ ($a \uplus_R I$) $\uplus_R I = a \uplus_R I$
⟨*proof*⟩

lemma (**in** *Ring*) *csrp-mem1*:⟦*ideal* $R\ I$; $x \in$ *carrier* (R /$_r$ I)⟧ ⟹
csrp-fn $R\ I\ x \in x$
⟨*proof*⟩

lemma (**in** *Ring*) *csrp-fn-mem*:⟦*ideal* $R\ I$; $x \in$ *carrier* (R /$_r$ I)⟧ ⟹

$$(csrp\text{-}fn\ R\ I\ x) \in carrier\ R$$

⟨*proof*⟩

lemma (**in** *Ring*) *csrp-eq-coset*:⟦*ideal R I*; $x \in carrier\ (R\ /_r\ I)$⟧ ⟹
$(csrp\text{-}fn\ R\ I\ x) \uplus_R I = x$

⟨*proof*⟩

lemma (**in** *Ring*) *csrp-nz-nz*:⟦*ideal R I*; $x \in carrier\ (R\ /_r\ I)$;
$x \neq \mathbf{0}_{(R\ /_r\ I)}$⟧ ⟹ $(csrp\text{-}fn\ R\ I\ x) \neq \mathbf{0}$

⟨*proof*⟩

lemma (**in** *Ring*) *csrp-diff-in-vpr*:⟦*ideal R I*; $x \in carrier\ R$⟧ ⟹
$x \pm (-_a\ (csrp\text{-}fn\ R\ I\ (pj\ R\ I\ x))) \in I$

⟨*proof*⟩

lemma (**in** *Ring*) *csrp-pj*:⟦*ideal R I*; $x \in carrier\ (R\ /_r\ I)$⟧ ⟹
$(pj\ R\ I)\ (csrp\text{-}fn\ R\ I\ x) = x$

⟨*proof*⟩

4.12 Polynomial ring

In this section, we treat a ring of polynomials over a ring S. Numbers are of type ant

definition
pol-coeff :: [('*a*, '*more*) *Ring-scheme*, (*nat* × (*nat* ⇒ '*a*))] ⇒ *bool* **where**
pol-coeff S c ⟷ (∀ *j* ≤ (*fst c*). (*snd c*) *j* ∈ *carrier S*)

definition
c-max :: [('*a*, '*more*) *Ring-scheme*, *nat* × (*nat* ⇒ '*a*)] ⇒ *nat* **where**
c-max S c = (*if* {*j*. *j* ≤ (*fst c*) ∧ (*snd c*) *j* ≠ $\mathbf{0}_S$} = {} *then 0 else*
n-max {*j*. *j* ≤ (*fst c*) ∧ (*snd c*) *j* ≠ $\mathbf{0}_S$})

definition
polyn-expr :: [('*a*, '*more*) *Ring-scheme*, '*a*, *nat*, *nat* × (*nat* ⇒ '*a*)] ⇒ '*a* **where**
polyn-expr R X k c == *nsum R* (λ*j*. ((*snd c*) *j*) $\cdot_{r\,R}$ ($X^{\frown R}$ *j*)) *k*

definition
algfree-cond :: [('*a*, '*m*) *Ring-scheme*, ('*a*, '*m1*) *Ring-scheme*,
'*a*] ⇒ *bool* **where**
algfree-cond R S X ⟷ (∀ *c*. *pol-coeff S c* ∧ (∀ *k* ≤ (*fst c*).
(*nsum R* (λ*j*. ((*snd c*) *j*) $\cdot_{r\,R}$ ($X^{\frown R}$ *j*)) *k* = $\mathbf{0}_R$ ⟶
(∀ *j* ≤ *k*. (*snd c*) *j* = $\mathbf{0}_S$))))

locale *PolynRg* = *Ring* +
fixes *S* (**structure**)
fixes *X* (**structure**)
assumes *X-mem-R*:*X* ∈ *carrier R*

and *not-zeroring*:$\neg$ *zeroring S*
and *subring*: *Subring R S*
and *algfree*: *algfree-cond R S X*
and *S-X-generate*:$x \in$ *carrier R* $\Longrightarrow$
$\exists f$. *pol-coeff S f* $\wedge$ $x =$ *polyn-expr R X* (*fst f*) *f*

4.13 Addition and multiplication of *polyn-exprs*

4.13.1 Simple properties of a *polyn-ring*

lemma *Subring-subset*:*Subring R S* $\Longrightarrow$ *carrier S* $\subseteq$ *carrier R*
⟨*proof*⟩

lemma (**in** *Ring*) *subring-Ring*:*Subring R S* $\Longrightarrow$ *Ring S*
⟨*proof*⟩

lemma (**in** *Ring*) *mem-subring-mem-ring*:⟦*Subring R S*; $x \in$ *carrier S*⟧ $\Longrightarrow$
$x \in$ *carrier R*
⟨*proof*⟩

lemma (**in** *Ring*) *Subring-pOp-ring-pOp*:⟦*Subring R S*; $a \in$ *carrier S*;
$b \in$ *carrier S* ⟧ $\Longrightarrow$ $a \pm_S b = a \pm b$
⟨*proof*⟩

lemma (**in** *Ring*) *Subring-tOp-ring-tOp*:⟦*Subring R S*; $a \in$ *carrier S*;
$b \in$ *carrier S* ⟧ $\Longrightarrow$ $a \cdot_{r S} b = a \cdot_r b$
⟨*proof*⟩

lemma (**in** *Ring*) *Subring-one-ring-one*:*Subring R S* $\Longrightarrow$ $1_{r S} = 1_r$
⟨*proof*⟩

lemma (**in** *Ring*) *Subring-zero-ring-zero*:*Subring R S* $\Longrightarrow$ $\mathbf{0}_S = \mathbf{0}$
⟨*proof*⟩

lemma (**in** *Ring*) *Subring-minus-ring-minus*:⟦*Subring R S*; $x \in$ *carrier S*⟧
$\Longrightarrow$ $-_{a S} x = -_a x$
⟨*proof*⟩

lemma (**in** *PolynRg*) *Subring-pow-ring-pow*:$x \in$ *carrier S* $\Longrightarrow$
$x^{\frown S} n = x^{\frown R} n$
⟨*proof*⟩

lemma (**in** *PolynRg*) *is-Ring*: *Ring R* ⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-ring-nonzero*:$1_r \neq \mathbf{0}$
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-ring-S-nonzero*:$1_{r S} \neq \mathbf{0}_S$
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-ring-X-nonzero*:$X \neq \mathbf{0}$
⟨*proof*⟩

4.13.2 Coefficients of a polynomial

lemma (**in** *PolynRg*) *pol-coeff-split*:*pol-coeff S f* = *pol-coeff S* (*fst f*, *snd f*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *pol-coeff-cartesian*:*pol-coeff S c* ⟹
(*fst c*, *snd c*) = *c*
⟨*proof*⟩

lemma (**in** *PolynRg*) *split-pol-coeff*:⟦*pol-coeff S c*; *k* ≤ (*fst c*)⟧ ⟹
pol-coeff S (*k*, *snd c*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *pol-coeff-pre*:*pol-coeff S* ((*Suc n*), *f*) ⟹
pol-coeff S (*n*, *f*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *pol-coeff-le*:⟦*pol-coeff S c*; *n* ≤ (*fst c*)⟧ ⟹
pol-coeff S (*n*, (*snd c*))
⟨*proof*⟩

lemma (**in** *PolynRg*) *pol-coeff-mem*:⟦*pol-coeff S c*; *j* ≤ (*fst c*)⟧ ⟹
((*snd c*) *j*) ∈ *carrier S*
⟨*proof*⟩

lemma (**in** *PolynRg*) *pol-coeff-mem-R*:⟦*pol-coeff S c*; *j* ≤ (*fst c*)⟧
⟹ ((*snd c*) *j*) ∈ *carrier R*
⟨*proof*⟩

lemma (**in** *PolynRg*) *Slide-pol-coeff*:⟦*pol-coeff S c*; *n* < (*fst c*)⟧ ⟹
pol-coeff S (((*fst c*) − *Suc n*), (λ*x*. (*snd c*) (*Suc* (*n* + *x*))))
⟨*proof*⟩

4.13.3 Addition of *polyn-exprs*

lemma (**in** *PolynRg*) *monomial-mem*:*pol-coeff S c* ⟹
∀ *j* ≤ (*fst c*). (*snd c*) *j* ·$_r$ $X^{\hat{}R}$ *j* ∈ *carrier R*
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-mem*:⟦*pol-coeff S c*; *k* ≤ (*fst c*)⟧ ⟹
polyn-expr R X k c ∈ *carrier R*
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-exprs-eq*:⟦*pol-coeff S c*; *pol-coeff S d*;
k ≤ (*min* (*fst c*) (*fst d*)); ∀ *j* ≤ *k*. (*snd c*) *j* = (*snd d*) *j*⟧ ⟹
polyn-expr R X k c = *polyn-expr R X k d*

⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-expr-restrict*:*pol-coeff S* (*Suc n*, *f*) ⟹
polyn-expr R X n (*Suc n*, *f*) = *polyn-expr R X n* (*n*, *f*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-expr-short*:⟦*pol-coeff S c*; *k* ≤ (*fst c*)⟧ ⟹
polyn-expr R X k c = *polyn-expr R X k* (*k*, *snd c*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-expr0*:*pol-coeff S c* ⟹
polyn-expr R X 0 c = (*snd c*) *0*
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-expr-split*:
polyn-expr R X k f = *polyn-expr R X k* (*fst f*, *snd f*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-Suc*:*Suc n* ≤ (*fst c*) ⟹
polyn-expr R X (*Suc n*) ((*Suc n*), (*snd c*)) =
polyn-expr R X n c ± ((*snd c*) (*Suc n*)) ·$_r$ ($X^{\frown R}$ (*Suc n*))
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-Suc-split*:*pol-coeff S* (*Suc n*, *f*) ⟹
polyn-expr R X (*Suc n*) ((*Suc n*), *f*) =
polyn-expr R X n (*n*, *f*) ± (*f* (*Suc n*)) ·$_r$ ($X^{\frown R}$ (*Suc n*))
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-n-m*:⟦*pol-coeff S c*; *n* < *m*; *m* ≤ (*fst c*)⟧ ⟹
polyn-expr R X m (*m*, (*snd c*)) = *polyn-expr R X n* (*n*, (*snd c*)) ±
(*fSum R* (λ*j*. ((*snd c*) *j*) ·$_r$ ($X^{\frown R}$ *j*)) (*Suc n*) *m*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-n-m1*:⟦*pol-coeff S c*; *n* < *m*; *m* ≤ (*fst c*)⟧ ⟹
polyn-expr R X m c = *polyn-expr R X n c* ±
(*fSum R* (λ*j*. ((*snd c*) *j*) ·$_r$ ($X^{\frown R}$ *j*)) (*Suc n*) *m*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-n-m-mem*:⟦*pol-coeff S c*; *n* < *m*; *m* ≤ (*fst c*)⟧ ⟹
(*fSum R* (λ*j*. ((*snd c*) *j*) ·$_r$ ($X^{\frown R}$ *j*)) (*Suc n*) *m*) ∈ *carrier R*
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-n-ms-eq*:⟦*pol-coeff S c*; *pol-coeff S d*;
m ≤ *min* (*fst c*) (*fst d*); *n* < *m*;
∀ *j*∈*nset* (*Suc n*) *m*. (*snd c*) *j* = (*snd d*) *j*⟧ ⟹
(*fSum R* (λ*j*. ((*snd c*) *j*) ·$_r$ ($X^{\frown R}$ *j*)) (*Suc n*) *m*) =
(*fSum R* (λ*j*. ((*snd d*) *j*) ·$_r$ ($X^{\frown R}$ *j*)) (*Suc n*) *m*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-addTr*:
$(pol\text{-}coeff\ S\ (n,\ f)) \wedge (pol\text{-}coeff\ S\ (n,\ g)) \longrightarrow$
$(polyn\text{-}expr\ R\ X\ n\ (n,\ f)) \pm (polyn\text{-}expr\ R\ X\ n\ (n,\ g)) =$
$nsum\ R\ (\lambda j.\ ((f\ j) \pm_S (g\ j)) \cdot_r (X^{\frown R}\ j))\ n$
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-add-n*:$[\![pol\text{-}coeff\ S\ (n,\ f);\ pol\text{-}coeff\ S\ (n,\ g)]\!] \Longrightarrow$
$(polyn\text{-}expr\ R\ X\ n\ (n,\ f)) \pm (polyn\text{-}expr\ R\ X\ n\ (n,\ g)) =$
$nsum\ R\ (\lambda j.\ ((f\ j) \pm_S (g\ j)) \cdot_r (X^{\frown R}\ j))\ n$
⟨*proof*⟩

definition
$add\text{-}cf :: [('a,\ 'm)\ Ring\text{-}scheme,\ nat \times (nat \Rightarrow 'a),\ nat \times (nat \Rightarrow 'a)] \Rightarrow$
$nat \times (nat \Rightarrow 'a)$ **where**
$add\text{-}cf\ S\ c\ d =$
$(if\ (fst\ c) < (fst\ d)\ then\ ((fst\ d),\ \lambda j.\ (if\ j \leq (fst\ c)$
$then\ (((snd\ c)\ j) \pm_S ((snd\ d)\ j))\ else\ ((snd\ d)\ j)))$
$else\ if\ (fst\ c) = (fst\ d)\ then\ ((fst\ c),\ \lambda j.\ ((snd\ c)\ j \pm_S (snd\ d)\ j))$
$else\ ((fst\ c),\ \lambda j.\ (if\ j \leq (fst\ d)\ then$
$((snd\ c)\ j \pm_S (snd\ d)\ j)\ else\ ((snd\ c)\ j))))$

lemma (**in** *PolynRg*) *add-cf-pol-coeff*:$[\![pol\text{-}coeff\ S\ c;\ pol\text{-}coeff\ S\ d]\!]$
$\Longrightarrow\ pol\text{-}coeff\ S\ (add\text{-}cf\ S\ c\ d)$
⟨*proof*⟩

lemma (**in** *PolynRg*) *add-cf-len*:$[\![pol\text{-}coeff\ S\ c;\ pol\text{-}coeff\ S\ d]\!]$
$\Longrightarrow fst\ (add\text{-}cf\ S\ c\ d) = (max\ (fst\ c)\ (fst\ d))$
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-expr-restrict1*:$[\![pol\text{-}coeff\ S\ (n,\ f);$
$pol\text{-}coeff\ S\ (Suc\ (m + n),\ g)]\!] \Longrightarrow$
$polyn\text{-}expr\ R\ X\ (m + n)\ (add\text{-}cf\ S\ (n,\ f)\ (m + n,\ g)) =$
$polyn\text{-}expr\ R\ X\ (m + n)\ (m + n,\ snd\ (add\text{-}cf\ S\ (n,\ f)\ (Suc\ (m + n),\ g)))$
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-add-n1*:$[\![pol\text{-}coeff\ S\ (n,\ f);\ pol\text{-}coeff\ S\ (n,\ g)]\!] \Longrightarrow$
$(polyn\text{-}expr\ R\ X\ n\ (n,\ f)) \pm (polyn\text{-}expr\ R\ X\ n\ (n,\ g)) =$
$polyn\text{-}expr\ R\ X\ n\ (add\text{-}cf\ S\ (n,\ f)\ (n,\ g))$
⟨*proof*⟩

lemma (**in** *PolynRg*) *add-cf-val-hi*:$(fst\ c) < (fst\ d) \Longrightarrow$
$snd\ (add\text{-}cf\ S\ c\ d)\ (fst\ d) = (snd\ d)\ (fst\ d)$
⟨*proof*⟩

lemma (**in** *PolynRg*) *add-cf-commute*:$[\![pol\text{-}coeff\ S\ c;\ pol\text{-}coeff\ S\ d]\!]$
$\Longrightarrow \forall j \leq (max\ (fst\ c)\ (fst\ d)).\ snd\ (add\text{-}cf\ S\ c\ d)\ j =$

$$snd\ (add\text{-}cf\ S\ d\ c)\ j$$

⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-addTr1*:*pol-coeff S* $(n, f) \Longrightarrow$
$\forall g.\ pol\text{-}coeff\ S\ (n + m, g) \longrightarrow$
$(polyn\text{-}expr\ R\ X\ n\ (n, f) \pm (polyn\text{-}expr\ R\ X\ (n + m)\ ((n + m), g))$
$= polyn\text{-}expr\ R\ X\ (n + m)\ (add\text{-}cf\ S\ (n, f)\ ((n + m), g)))$

⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-add*:$[\![pol\text{-}coeff\ S\ (n, f);\ pol\text{-}coeff\ S\ (m, g)]\!]$
$\Longrightarrow polyn\text{-}expr\ R\ X\ n\ (n, f) \pm (polyn\text{-}expr\ R\ X\ m\ (m, g))$
$= polyn\text{-}expr\ R\ X\ (max\ n\ m)\ (add\text{-}cf\ S\ (n, f)\ (m, g))$

⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-add1*:$[\![pol\text{-}coeff\ S\ c;\ pol\text{-}coeff\ S\ d]\!]$
$\Longrightarrow polyn\text{-}expr\ R\ X\ (fst\ c)\ c \pm (polyn\text{-}expr\ R\ X\ (fst\ d)\ d)$
$= polyn\text{-}expr\ R\ X\ (max\ (fst\ c)\ (fst\ d))\ (add\text{-}cf\ S\ c\ d)$

⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-minus-nsum*:$[\![pol\text{-}coeff\ S\ c;\ k \leq (fst\ c)]\!] \Longrightarrow$
$-_a\ (polyn\text{-}expr\ R\ X\ k\ c) = nsum\ R\ (\lambda j.\ ((-_a S)\ ((snd\ c)\ j)) \cdot_r (X^{\frown R}\ j)))\ k$

⟨*proof*⟩

lemma (**in** *PolynRg*) *minus-pol-coeff*:*pol-coeff S c* $\Longrightarrow$
$pol\text{-}coeff\ S\ ((fst\ c), (\lambda j.\ (-_a S)\ ((snd\ c)\ j))))$

⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-minus*:$[\![pol\text{-}coeff\ S\ c;\ k \leq (fst\ c)]\!] \Longrightarrow$
$-_a\ (polyn\text{-}expr\ R\ X\ k\ c) =$
$polyn\text{-}expr\ R\ X\ k\ (fst\ c, (\lambda j.\ (-_a S)\ ((snd\ c)\ j))))$

⟨*proof*⟩

definition
m-cf :: $[('a, 'm)\ Ring\text{-}scheme, nat \times (nat \Rightarrow 'a)] \Rightarrow nat \times (nat \Rightarrow 'a)$ **where**
m-cf S c = $(fst\ c, (\lambda j.\ (-_a S)\ ((snd\ c)\ j))))$

lemma (**in** *PolynRg*) *m-cf-pol-coeff*:*pol-coeff S c* $\Longrightarrow$
$pol\text{-}coeff\ S\ (m\text{-}cf\ S\ c)$

⟨*proof*⟩

lemma (**in** *PolynRg*) *m-cf-len*:*pol-coeff S c* $\Longrightarrow$
$fst\ (m\text{-}cf\ S\ c) = fst\ c$

⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-minus-m-cf*:$[\![pol\text{-}coeff\ S\ c;\ k \leq (fst\ c)]\!] \Longrightarrow$
$-_a\ (polyn\text{-}expr\ R\ X\ k\ c) =$
$polyn\text{-}expr\ R\ X\ k\ (m\text{-}cf\ S\ c)$

⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-zero-minus-zero*:⟦*pol-coeff S c*; $k \leq (fst\ c)$⟧ ⟹
$(polyn\text{-}expr\ R\ X\ k\ c = \mathbf{0}) = (polyn\text{-}expr\ R\ X\ k\ (m\text{-}cf\ S\ c) = \mathbf{0})$
⟨*proof*⟩

lemma (**in** *PolynRg*) *coeff-0-pol-0*:⟦*pol-coeff S c*; $k \leq fst\ c$⟧ ⟹
$(\forall j{\leq}\ k.\ (snd\ c)\ j = \mathbf{0}_S) = (polyn\text{-}expr\ R\ X\ k\ c = \mathbf{0})$
⟨*proof*⟩

4.13.4 Multiplication of *pol-exprs*

4.13.5 Multiplication

definition
ext-cf :: [('*a*, '*m*) *Ring-scheme*, *nat*, *nat* × (*nat* ⇒ '*a*)] ⇒
nat × (*nat* ⇒ '*a*) **where**
ext-cf S n c = $(n + fst\ c,\ \lambda i.\ if\ n \leq i\ then\ (snd\ c)\ (sliden\ n\ i)\ else\ \mathbf{0}_S)$

definition
sp-cf :: [('*a*, '*m*) *Ring-scheme*, '*a*, *nat* × (*nat* ⇒ '*a*)] ⇒ *nat* × (*nat* ⇒ '*a*) **where**
sp-cf S a c = $(fst\ c,\ \lambda j.\ a \cdot_{r\,S} ((snd\ c)\ j))$

definition
special-cf :: ('*a*, '*m*) *Ring-scheme* ⇒ *nat* × (*nat* ⇒ '*a*) (‹C_0›) **where**
$C_0\ S = (0,\ \lambda j.\ 1_{r\,S})$

lemma (**in** *PolynRg*) *special-cf-pol-coeff*:*pol-coeff S* (C_0 *S*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *special-cf-len*:*fst* (C_0 *S*) = *0*
⟨*proof*⟩

lemma (**in** *PolynRg*) *ext-cf-pol-coeff*:*pol-coeff S c* ⟹
pol-coeff S (*ext-cf S n c*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *ext-cf-len*:*pol-coeff S c* ⟹
fst (*ext-cf S m c*) = *m* + *fst c*
⟨*proof*⟩

lemma (**in** *PolynRg*) *ext-special-cf-len*:*fst* (*ext-cf S m* (C_0 *S*)) = *m*
⟨*proof*⟩

lemma (**in** *PolynRg*) *ext-cf-self*:*pol-coeff S c* ⟹
$\forall j \leq (fst\ c).\ snd\ (ext\text{-}cf\ S\ 0\ c)\ j = (snd\ c)\ j$
⟨*proof*⟩

lemma (**in** *PolynRg*) *ext-cf-hi*:*pol-coeff S c* ⟹
(*snd c*) (*fst c*) =

$$snd\ (ext\text{-}cf\ S\ n\ c)\ (n + (fst\ c))$$

⟨*proof*⟩

lemma (**in** *PolynRg*) *ext-special-cf-hi*:$snd\ (ext\text{-}cf\ S\ n\ (C_0\ S))\ n = 1_{r\,S}$
⟨*proof*⟩

lemma (**in** *PolynRg*) *ext-cf-lo-zero*:$[\![pol\text{-}coeff\ S\ c;\ 0 < n;\ x \leq (n - Suc\ 0)]\!]$
$\Longrightarrow snd\ (ext\text{-}cf\ S\ n\ c)\ x = \mathbf{0}_S$
⟨*proof*⟩

lemma (**in** *PolynRg*) *ext-special-cf-lo-zero*:$[\![0 < n;\ x \leq (n - Suc\ 0)]\!]$
$\Longrightarrow snd\ (ext\text{-}cf\ S\ n\ (C_0\ S))\ x = \mathbf{0}_S$
⟨*proof*⟩

lemma (**in** *PolynRg*) *sp-cf-pol-coeff*:$[\![pol\text{-}coeff\ S\ c;\ a \in carrier\ S]\!] \Longrightarrow$
$pol\text{-}coeff\ S\ (sp\text{-}cf\ S\ a\ c)$
⟨*proof*⟩

lemma (**in** *PolynRg*) *sp-cf-len*:$[\![pol\text{-}coeff\ S\ c;\ a \in carrier\ S]\!] \Longrightarrow$
$fst\ (sp\text{-}cf\ S\ a\ c) = fst\ c$
⟨*proof*⟩

lemma (**in** *PolynRg*) *sp-cf-val*:$[\![pol\text{-}coeff\ S\ c;\ j \leq (fst\ c);\ a \in carrier\ S]\!] \Longrightarrow$
$snd\ (sp\text{-}cf\ S\ a\ c)\ j = a \cdot_{r\,S} ((snd\ c)\ j)$
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-ext-cf-lo-zero*:$[\![pol\text{-}coeff\ S\ c;\ 0 < j]\!] \Longrightarrow$
$polyn\text{-}expr\ R\ X\ (j - Suc\ 0)\ (ext\text{-}cf\ S\ j\ c) = \mathbf{0}$
⟨*proof*⟩

lemma (**in** *PolynRg*) *monomial-d*:$pol\text{-}coeff\ S\ c \Longrightarrow$
$polyn\text{-}expr\ R\ X\ d\ (ext\text{-}cf\ S\ d\ c) = ((snd\ c)\ 0) \cdot_r X^{\frown R\ d}$
⟨*proof*⟩

lemma (**in** *PolynRg*) *X-to-d*: $X^{\frown R\ d} = polyn\text{-}expr\ R\ X\ d\ (ext\text{-}cf\ S\ d\ (C_0\ S))$
⟨*proof*⟩

lemma (**in** *PolynRg*) *c-max-ext-special-cf*:$c\text{-}max\ S\ (ext\text{-}cf\ S\ n\ (C_0\ S)) = n$
⟨*proof*⟩

lemma (**in** *PolynRg*) *scalar-times-polynTr*:$a \in carrier\ S \Longrightarrow$
$\forall f.\ pol\text{-}coeff\ S\ (n, f) \longrightarrow$
$a \cdot_r (polyn\text{-}expr\ R\ X\ n\ (n, f)) = polyn\text{-}expr\ R\ X\ n\ (sp\text{-}cf\ S\ a\ (n, f))$
⟨*proof*⟩

lemma (**in** *PolynRg*) *scalar-times-pol-expr*:$[\![a \in carrier\ S;\ pol\text{-}coeff\ S\ c;$
$n \leq fst\ c]\!] \Longrightarrow$
$a \cdot_r (polyn\text{-}expr\ R\ X\ n\ c) = polyn\text{-}expr\ R\ X\ n\ (sp\text{-}cf\ S\ a\ c)$

⟨*proof*⟩

lemma (**in** *PolynRg*) *sp-coeff-nonzero*:⟦*Idomain S*; *a* ∈ *carrier S*; *a* ≠ $\mathbf{0}_S$;
pol-coeff S c; (*snd c*) *j* ≠ $\mathbf{0}_S$; *j* ≤ (*fst c*)⟧ ⟹
snd (*sp-cf S a c*) *j* ≠ $\mathbf{0}_S$
⟨*proof*⟩

lemma (**in** *PolynRg*) *ext-cf-inductTl*:*pol-coeff S* (*Suc n*, *f*) ⟹
polyn-expr R X (*n* + *j*) (*ext-cf S j* (*Suc n*, *f*)) =
polyn-expr R X (*n* + *j*) (*ext-cf S j* (*n*, *f*))
⟨*proof*⟩

lemma (**in** *PolynRg*) *low-deg-terms-zeroTr*:
pol-coeff S (*n*, *f*) ⟶
polyn-expr R X (*n* + *j*) (*ext-cf S j* (*n*, *f*)) =
($X^{\frown R}$ *j*) ·$_r$ (*polyn-expr R X n* (*n*, *f*))
⟨*proof*⟩

lemma (**in** *PolynRg*) *low-deg-terms-zero*:*pol-coeff S* (*n*, *f*) ⟹
polyn-expr R X (*n* + *j*) (*ext-cf S j* (*n*, *f*)) =
($X^{\frown R}$ *j*) ·$_r$ (*polyn-expr R X n* (*n*, *f*))
⟨*proof*⟩

lemma (**in** *PolynRg*) *low-deg-terms-zero1*:*pol-coeff S c* ⟹
polyn-expr R X ((*fst c*) + *j*) (*ext-cf S j c*) =
($X^{\frown R}$ *j*) ·$_r$ (*polyn-expr R X* (*fst c*) *c*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-expr-tOpTr*:*pol-coeff S* (*n*, *f*) ⟹
∀ *g*. (*pol-coeff S* (*m*, *g*) ⟶ (∃ *h*. *pol-coeff S* ((*n* + *m*), *h*) ∧
h (*n* + *m*) = (*f n*) ·$_{r\,S}$ (*g m*) ∧
(*polyn-expr R X* (*n* + *m*) (*n* + *m*, *h*) =
(*polyn-expr R X n* (*n*, *f*)) ·$_r$ (*polyn-expr R X m* (*m*, *g*)))))
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-expr-tOp*:⟦
pol-coeff S (*n*, *f*); *pol-coeff S* (*m*, *g*)⟧ ⟹ ∃ *e*. *pol-coeff S* ((*n* + *m*), *e*) ∧
e (*n* + *m*) = (*f n*) ·$_{r\,S}$ (*g m*) ∧
polyn-expr R X (*n* + *m*)(*n* + *m*, *e*) =
(*polyn-expr R X n* (*n*, *f*)) ·$_r$ (*polyn-expr R X m* (*m*, *g*))
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-expr-tOp-c*:⟦*pol-coeff S c*; *pol-coeff S d*⟧ ⟹
∃ *e*. *pol-coeff S e* ∧ (*fst e* = *fst c* + *fst d*) ∧
(*snd e*) (*fst e*) = (*snd c* (*fst c*)) ·$_{r\,S}$ (*snd d*) (*fst d*) ∧
polyn-expr R X (*fst e*) *e* =
(*polyn-expr R X* (*fst c*) *c*) ·$_r$ (*polyn-expr R X* (*fst d*) *d*)

⟨*proof*⟩

4.14 The degree of a polynomial

lemma (**in** *PolynRg*) *polyn-degreeTr*:⟦*pol-coeff S c*; $k \leq (fst\ c)$⟧ ⟹
$(polyn\text{-}expr\ R\ X\ k\ c = \mathbf{0}\ ) = (\{j.\ j \leq k \wedge (snd\ c)\ j \neq \mathbf{0}_S\} = \{\})$
⟨*proof*⟩

lemma (**in** *PolynRg*) *higher-part-zero*:⟦*pol-coeff S c*; $k < fst\ c$;
$\forall j{\in}nset\ (Suc\ k)\ (fst\ c).\ snd\ c\ j = \mathbf{0}_S$⟧ ⟹
$\Sigma_f\ R\ (\lambda j.\ snd\ c\ j \cdot_r X^{\frown R\ j})\ (Suc\ k)\ (fst\ c) = \mathbf{0}$
⟨*proof*⟩

lemma (**in** *PolynRg*) *coeff-nonzero-polyn-nonzero*:⟦*pol-coeff S c*; $k \leq (fst\ c)$⟧
⟹ $(polyn\text{-}expr\ R\ X\ k\ c \neq \mathbf{0}) = (\exists j{\leq}k.\ (snd\ c)\ j \neq \mathbf{0}_S\ )$
⟨*proof*⟩

lemma (**in** *PolynRg*) *pol-expr-unique*:⟦$p \in carrier\ R$; $p \neq \mathbf{0}$;
pol-coeff S c; $p = polyn\text{-}expr\ R\ X\ (fst\ c)\ c$; $(snd\ c)\ (fst\ c) \neq \mathbf{0}_S$;
pol-coeff S d; $p = polyn\text{-}expr\ R\ X\ (fst\ d)\ d$; $(snd\ d)\ (fst\ d) \neq \mathbf{0}_S$⟧ ⟹
$(fst\ c) = (fst\ d) \wedge (\forall j \leq (fst\ c).\ (snd\ c)\ j = (snd\ d)\ j)$
⟨*proof*⟩

lemma (**in** *PolynRg*) *pol-expr-unique2*:⟦*pol-coeff S c*; *pol-coeff S d*;
$fst\ c = fst\ d$⟧ ⟹
$(polyn\text{-}expr\ R\ X\ (fst\ c)\ c = polyn\text{-}expr\ R\ X\ (fst\ d)\ d\ ) =$
$(\forall j \leq (fst\ c).\ (snd\ c)\ j = (snd\ d)\ j)$
⟨*proof*⟩

lemma (**in** *PolynRg*) *pol-expr-unique3*:⟦*pol-coeff S c*; *pol-coeff S d*;
$fst\ c < fst\ d$⟧ ⟹
$(polyn\text{-}expr\ R\ X\ (fst\ c)\ c = polyn\text{-}expr\ R\ X\ (fst\ d)\ d\ ) =$
$((\forall j \leq (fst\ c).\ (snd\ c)\ j = (snd\ d)\ j) \wedge$
$(\forall j{\in}nset\ (Suc\ (fst\ c))\ (fst\ d).\ (snd\ d)\ j = \mathbf{0}_S))$
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-degree-unique*:⟦*pol-coeff S c*; *pol-coeff S d*;
$polyn\text{-}expr\ R\ X\ (fst\ c)\ c = polyn\text{-}expr\ R\ X\ (fst\ d)\ d$⟧ ⟹
$c\text{-}max\ S\ c = c\text{-}max\ S\ d$
⟨*proof*⟩

lemma (**in** *PolynRg*) *ex-polyn-expr*:$p \in carrier\ R$ ⟹
$\exists c.\ pol\text{-}coeff\ S\ c \wedge p = polyn\text{-}expr\ R\ X\ (fst\ c)\ c$
⟨*proof*⟩

lemma (**in** *PolynRg*) *c-max-eqTr0*:⟦*pol-coeff S c*; $k \leq (fst\ c)$;
$polyn\text{-}expr\ R\ X\ k\ c = polyn\text{-}expr\ R\ X\ (fst\ c)\ c$; $\exists j{\leq}k.\ (snd\ c)\ j \neq \mathbf{0}_S$⟧ ⟹
$c\text{-}max\ S\ (k,\ snd\ c) = c\text{-}max\ S\ c$

⟨*proof*⟩

definition
cf-sol :: [('*a*, '*b*) *Ring-scheme*, ('*a*, '*b1*) *Ring-scheme*, '*a*, '*a*,
nat × (*nat* ⇒ '*a*)] ⇒ *bool* **where**
cf-sol R S X p c ⟷ *pol-coeff S c* ∧ (*p* = *polyn-expr R X* (*fst c*) *c*)

definition
deg-n ::[('*a*, '*b*) *Ring-scheme*, ('*a*, '*b1*) *Ring-scheme*, '*a*, '*a*] ⇒ *nat* **where**
deg-n R S X p = *c-max S* (*SOME c. cf-sol R S X p c*)

definition
deg ::[('*a*, '*b*) *Ring-scheme*, ('*a*, '*b1*) *Ring-scheme*, '*a*, '*a*] ⇒ *ant* **where**
deg R S X p = (*if p* = $\mathbf{0}_R$ *then* −∞ *else* (*an* (*deg-n R S X p*)))

lemma (**in** *PolynRg*) *ex-cf-sol*:*p* ∈ *carrier R* ⟹
∃ *c. cf-sol R S X p c*
⟨*proof*⟩

lemma (**in** *PolynRg*) *deg-in-aug-minf*:*p* ∈ *carrier R* ⟹
deg R S X p ∈ $Z_{-\infty}$
⟨*proof*⟩

lemma (**in** *PolynRg*) *deg-noninf*:*p* ∈ *carrier R* ⟹
deg R S X p ≠ ∞
⟨*proof*⟩

lemma (**in** *PolynRg*) *deg-ant-int*:⟦*p* ∈ *carrier R*; *p* ≠ $\mathbf{0}$⟧
⟹ *deg R S X p* = *ant* (*int* (*deg-n R S X p*))
⟨*proof*⟩

lemma (**in** *PolynRg*) *deg-an*:⟦*p* ∈ *carrier R*; *p* ≠ $\mathbf{0}$⟧
⟹ *deg R S X p* = *an* (*deg-n R S X p*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *pol-SOME-1*:*p* ∈ *carrier R* ⟹
cf-sol R S X p (*SOME f. cf-sol R S X p f*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *pol-SOME-2*:*p* ∈ *carrier R* ⟹
pol-coeff S (*SOME c. cf-sol R S X p c*) ∧
p = *polyn-expr R X* (*fst* (*SOME c. cf-sol R S X p c*))
(*SOME c. cf-sol R S X p c*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *coeff-max-zeroTr*:*pol-coeff S c* ⟹
∀ *j*. *j* ≤ (*fst c*) ∧ (*c-max S c*) < *j* ⟶ (*snd c*) *j* = $\mathbf{0}_S$
⟨*proof*⟩

lemma (**in** *PolynRg*) *coeff-max-nonzeroTr*:⟦*pol-coeff S c*;
$\exists j \leq (fst\ c).\ (snd\ c)\ j \neq \mathbf{0}_S$⟧ ⟹ $(snd\ c)\ (c\text{-}max\ S\ c) \neq \mathbf{0}_S$
⟨*proof*⟩

lemma (**in** *PolynRg*) *coeff-max-bddTr*:*pol-coeff S c* ⟹ $c\text{-}max\ S\ c \leq (fst\ c)$
⟨*proof*⟩

lemma (**in** *PolynRg*) *pol-coeff-max*:*pol-coeff S c* ⟹
pol-coeff S $((c\text{-}max\ S\ c),\ snd\ c)$
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-c-max*:*pol-coeff S c* ⟹
$polyn\text{-}expr\ R\ X\ (fst\ c)\ c = polyn\text{-}expr\ R\ X\ (c\text{-}max\ S\ c)\ c$
⟨*proof*⟩

lemma (**in** *PolynRg*) *pol-deg-eq-c-max*:⟦*p* ∈ *carrier R*;
pol-coeff S c; $p = polyn\text{-}expr\ R\ X\ (fst\ c)\ c$⟧ ⟹
$deg\text{-}n\ R\ S\ X\ p = c\text{-}max\ S\ c$
⟨*proof*⟩

lemma (**in** *PolynRg*) *pol-deg-le-n*:⟦*p* ∈ *carrier R*; *pol-coeff S c*;
$p = polyn\text{-}expr\ R\ X\ (fst\ c)\ c$⟧ ⟹ $deg\text{-}n\ R\ S\ X\ p \leq (fst\ c)$
⟨*proof*⟩

lemma (**in** *PolynRg*) *pol-deg-le-n1*:⟦*p* ∈ *carrier R*; *pol-coeff S c*; $k \leq (fst\ c)$;
$p = polyn\text{-}expr\ R\ X\ k\ c$⟧ ⟹ $deg\text{-}n\ R\ S\ X\ p \leq k$
⟨*proof*⟩

lemma (**in** *PolynRg*) *pol-len-gt-deg*:⟦*p* ∈ *carrier R*; *pol-coeff S c*;
$p = polyn\text{-}expr\ R\ X\ (fst\ c)\ c$; $deg\ R\ S\ X\ p < (an\ j)$; $j \leq (fst\ c)$⟧
⟹ $(snd\ c)\ j = \mathbf{0}_S$
⟨*proof*⟩

lemma (**in** *PolynRg*) *pol-diff-deg-less*:⟦*p* ∈ *carrier R*; *pol-coeff S c*;
$p = polyn\text{-}expr\ R\ X\ (fst\ c)\ c$; *pol-coeff S d*;
$fst\ c = fst\ d$; $(snd\ c)\ (fst\ c) = (snd\ d)\ (fst\ d)$⟧ ⟹
$p \pm (-_a\ (polyn\text{-}expr\ R\ X\ (fst\ d)\ d)) = \mathbf{0} \vee$
$deg\text{-}n\ R\ S\ X\ (p \pm (-_a\ (polyn\text{-}expr\ R\ X\ (fst\ d)\ d))) < (fst\ c)$
⟨*proof*⟩

lemma (**in** *PolynRg*) *pol-pre-lt-deg*:⟦*p* ∈ *carrier R*; *pol-coeff S c*;
$deg\text{-}n\ R\ S\ X\ p \leq (fst\ c)$; $(deg\text{-}n\ R\ S\ X\ p) \neq 0$;
$p = polyn\text{-}expr\ R\ X\ (deg\text{-}n\ R\ S\ X\ p)\ c$ ⟧ ⟹
$(deg\text{-}n\ R\ S\ X\ (polyn\text{-}expr\ R\ X\ ((deg\text{-}n\ R\ S\ X\ p) - Suc\ 0)\ c)) < (deg\text{-}n\ R\ S\ X\ p)$
⟨*proof*⟩

lemma (**in** *PolynRg*) *pol-deg-n*:⟦*p* ∈ *carrier R*; *pol-coeff S c*;
$n \leq fst\ c$; $p = polyn\text{-}expr\ R\ X\ n\ c$; $(snd\ c)\ n \neq \mathbf{0}_S$⟧ ⟹
$deg\text{-}n\ R\ S\ X\ p = n$

⟨*proof*⟩

lemma (**in** *PolynRg*) *pol-expr-deg*:⟦$p \in$ *carrier* R; $p \neq \mathbf{0}$⟧
$\Longrightarrow \exists\, c.$ *pol-coeff* S $c \wedge$ *deg-n* R S X $p \leq$ (*fst* c) $\wedge$
$p =$ *polyn-expr* R X (*deg-n* R S X p) $c \wedge$
(*snd* c) (*deg-n* R S X p) $\neq \mathbf{0}_S$
⟨*proof*⟩

lemma (**in** *PolynRg*) *deg-n-pos*:$p \in$ *carrier* $R \Longrightarrow 0 \leq$ *deg-n* R S X p
⟨*proof*⟩

lemma (**in** *PolynRg*) *pol-expr-deg1*:⟦$p \in$ *carrier* R; $d =$ *na* (*deg* R S X p)⟧ $\Longrightarrow$
$\exists\, c.$ (*pol-coeff* S $c \wedge$ $p =$ *polyn-expr* R X d c)
⟨*proof*⟩

end

theory *Algebra6* **imports** *Algebra5* **begin**

definition
s-cf :: [('*a*, '*m*) *Ring-scheme*, ('*a*, '*m1*) *Ring-scheme*, '*a*, '*a*]
$\Rightarrow$ *nat* $\times$ (*nat* $\Rightarrow$ '*a*) **where**
s-cf R S X p = (*if* $p = \mathbf{0}_R$ *then* (*0*, $\lambda j.\ \mathbf{0}_S$) *else*
SOME c. (*pol-coeff* S $c \wedge$ $p =$ *polyn-expr* R X (*fst* c) $c \wedge$
(*snd* c) (*fst* c) $\neq \mathbf{0}_S$))

definition
lcf :: [('*a*, '*m*) *Ring-scheme*, ('*a*, '*m1*) *Ring-scheme*, '*a*, '*a*] $\Rightarrow$ '*a* **where**
lcf R S X p = (*snd* (*s-cf* R S X p)) (*fst* (*s-cf* R S X p))

lemma (**in** *PolynRg*) *lcf-val-0*:*lcf* R S X $\mathbf{0} = \mathbf{0}_S$
⟨*proof*⟩

lemma (**in** *PolynRg*) *lcf-val*:⟦$p \in$ *carrier* R; $p \neq \mathbf{0}$ ⟧ $\Longrightarrow$
lcf R S X p = (*snd* (*s-cf* R S X p)) (*fst* (*s-cf* R S X p))
⟨*proof*⟩

lemma (**in** *PolynRg*) *s-cf-pol-coeff*:$p \in$ *carrier* $R \Longrightarrow$
pol-coeff S (*s-cf* R S X p)
⟨*proof*⟩

lemma (**in** *PolynRg*) *lcf-mem*:$p \in$ *carrier* $R \Longrightarrow$ (*lcf* R S X p) $\in$ *carrier* S
⟨*proof*⟩

lemma (**in** *PolynRg*) *s-cf-expr0*:$p \in$ *carrier* $R \Longrightarrow$
pol-coeff S (*s-cf* R S X p) $\wedge$

$p = \mathit{polyn\text{-}expr}\ R\ X\ (\mathit{fst}\ (\mathit{s\text{-}cf}\ R\ S\ X\ p))\ (\mathit{s\text{-}cf}\ R\ S\ X\ p)$
⟨*proof*⟩

lemma (**in** *PolynRg*) *pos-deg-nonzero*:⟦$p \in \mathit{carrier}\ R$; $0 < \mathit{deg\text{-}n}\ R\ S\ X\ p$⟧ ⟹
$p \neq \mathbf{0}$
⟨*proof*⟩

lemma (**in** *PolynRg*) *s-cf-expr*:⟦$p \in \mathit{carrier}\ R$; $p \neq \mathbf{0}$⟧ ⟹
$\mathit{pol\text{-}coeff}\ S\ (\mathit{s\text{-}cf}\ R\ S\ X\ p) \wedge$
$p = \mathit{polyn\text{-}expr}\ R\ X\ (\mathit{fst}\ (\mathit{s\text{-}cf}\ R\ S\ X\ p))\ (\mathit{s\text{-}cf}\ R\ S\ X\ p) \wedge$
$(\mathit{snd}\ (\mathit{s\text{-}cf}\ R\ S\ X\ p))\ (\mathit{fst}\ (\mathit{s\text{-}cf}\ R\ S\ X\ p)) \neq \mathbf{0}_S$
⟨*proof*⟩

lemma (**in** *PolynRg*) *lcf-nonzero*:⟦$p \in \mathit{carrier}\ R$; $p \neq \mathbf{0}$ ⟧ ⟹
$\mathit{lcf}\ R\ S\ X\ p \neq \mathbf{0}_S$
⟨*proof*⟩

lemma (**in** *PolynRg*) *s-cf-deg*:⟦$p \in \mathit{carrier}\ R$; $p \neq \mathbf{0}$⟧ ⟹
$\mathit{deg\text{-}n}\ R\ S\ X\ p = \mathit{fst}\ (\mathit{s\text{-}cf}\ R\ S\ X\ p)$
⟨*proof*⟩

lemma (**in** *PolynRg*) *pol-expr-edeg*:⟦$p \in \mathit{carrier}\ R$; $\mathit{deg}\ R\ S\ X\ p \leq (\mathit{an}\ d)$⟧ ⟹
$\exists f.\ (\mathit{pol\text{-}coeff}\ S\ f \wedge \mathit{fst}\ f = d \wedge p = \mathit{polyn\text{-}expr}\ R\ X\ d\ f)$
⟨*proof*⟩

lemma (**in** *PolynRg*) *cf-scf*:⟦$\mathit{pol\text{-}coeff}\ S\ c$; $k \leq \mathit{fst}\ c$; $\mathit{polyn\text{-}expr}\ R\ X\ k\ c \neq \mathbf{0}$⟧
⟹ $\forall j \leq \mathit{fst}\ (\mathit{s\text{-}cf}\ R\ S\ X\ (\mathit{polyn\text{-}expr}\ R\ X\ k\ c)).$
$\mathit{snd}\ (\mathit{s\text{-}cf}\ R\ S\ X\ (\mathit{polyn\text{-}expr}\ R\ X\ k\ c))\ j = \mathit{snd}\ c\ j$
⟨*proof*⟩

definition
scf-cond :: [('*a*, '*m*) *Ring-scheme*, ('*a*, '*m1*) *Ring-scheme*, '*a*, '*a*,
nat, *nat* × (*nat* ⇒ '*a*)] ⇒ *bool* **where**
$\mathit{scf\text{-}cond}\ R\ S\ X\ p\ d\ c \longleftrightarrow \mathit{pol\text{-}coeff}\ S\ c \wedge \mathit{fst}\ c = d \wedge p = \mathit{polyn\text{-}expr}\ R\ X\ d\ c$

definition
scf-d :: [('*a*, '*m*) *Ring-scheme*, ('*a*, '*m1*) *Ring-scheme*, '*a*, '*a*, *nat*]
⇒ *nat* × (*nat* ⇒ '*a*) **where**
$\mathit{scf\text{-}d}\ R\ S\ X\ p\ d = (\mathit{SOME}\ f.\ \mathit{scf\text{-}cond}\ R\ S\ X\ p\ d\ f)$

lemma (**in** *PolynRg*) *scf-d-polTr*:⟦$p \in \mathit{carrier}\ R$; $\mathit{deg}\ R\ S\ X\ p \leq \mathit{an}\ d$⟧ ⟹
$\mathit{scf\text{-}cond}\ R\ S\ X\ p\ d\ (\mathit{scf\text{-}d}\ R\ S\ X\ p\ d)$
⟨*proof*⟩

lemma (**in** *PolynRg*) *scf-d-pol*:⟦$p \in \mathit{carrier}\ R$; $\mathit{deg}\ R\ S\ X\ p \leq \mathit{an}\ d$⟧ ⟹
$\mathit{pol\text{-}coeff}\ S\ (\mathit{scf\text{-}d}\ R\ S\ X\ p\ d) \wedge \mathit{fst}\ (\mathit{scf\text{-}d}\ R\ S\ X\ p\ d) = d \wedge$
$p = \mathit{polyn\text{-}expr}\ R\ X\ d\ (\mathit{scf\text{-}d}\ R\ S\ X\ p\ d)$

⟨*proof*⟩

lemma (**in** *PolynRg*) *pol-expr-of-X*:
$X = \textit{polyn-expr}\ R\ X\ (\textit{Suc}\ 0)\ (\textit{ext-cf}\ S\ (\textit{Suc}\ 0)\ (C_0\ S))$
⟨*proof*⟩

lemma (**in** *PolynRg*) *deg-n-of-X*:$\textit{deg-n}\ R\ S\ X\ X = \textit{Suc}\ 0$
⟨*proof*⟩

lemma (**in** *PolynRg*) *pol-X*:$\textit{cf-sol}\ R\ S\ X\ X\ c \Longrightarrow$
$\textit{snd}\ c\ 0 = \mathbf{0}_S \wedge \textit{snd}\ c\ (\textit{Suc}\ 0) = 1_{rS}$

⟨*proof*⟩

lemma (**in** *PolynRg*) *pol-of-deg0*:$[\![p \in \textit{carrier}\ R;\ p \neq \mathbf{0}]\!]$
$\Longrightarrow\ (\textit{deg-n}\ R\ S\ X\ p = 0) = (p \in \textit{carrier}\ S)$
⟨*proof*⟩

lemma (**in** *PolynRg*) *pols-const*:$[\![p \in \textit{carrier}\ R;\ (\textit{deg}\ R\ S\ X\ p) \leq 0]\!]\ \Longrightarrow$
$p \in \textit{carrier}\ S$
⟨*proof*⟩

lemma (**in** *PolynRg*) *less-deg-add-nonzero*:$[\![p \in \textit{carrier}\ R;\ p \neq \mathbf{0};$
$q \in \textit{carrier}\ R;\ q \neq \mathbf{0};$
$(\textit{deg-n}\ R\ S\ X\ p) < (\textit{deg-n}\ R\ S\ X\ q)]\!]\ \Longrightarrow p \pm q \neq \mathbf{0}$
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-deg-add1*:$[\![p \in \textit{carrier}\ R;\ p \neq \mathbf{0};\ q \in \textit{carrier}\ R;$
$q \neq \mathbf{0};\ (\textit{deg-n}\ R\ S\ X\ p) < (\textit{deg-n}\ R\ S\ X\ q)]\!]\ \Longrightarrow$
$\textit{deg-n}\ R\ S\ X\ (p \pm q) = (\textit{deg-n}\ R\ S\ X\ q)$
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-deg-add2*:$[\![p \in \textit{carrier}\ R;\ p \neq \mathbf{0};\ q \in \textit{carrier}\ R;$
$q \neq \mathbf{0};\ p \pm q \neq \mathbf{0};\ (\textit{deg-n}\ R\ S\ X\ p) = (\textit{deg-n}\ R\ S\ X\ q)]\!]\ \Longrightarrow$
$\textit{deg-n}\ R\ S\ X\ (p \pm q) \leq (\textit{deg-n}\ R\ S\ X\ q)$
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-deg-add3*:$[\![p \in \textit{carrier}\ R;\ p \neq \mathbf{0};\ q \in \textit{carrier}\ R;$
$q \neq \mathbf{0};\ p \pm q \neq \mathbf{0};\ (\textit{deg-n}\ R\ S\ X\ p) \leq n;\ (\textit{deg-n}\ R\ S\ X\ q) \leq n]\!]\ \Longrightarrow$
$\textit{deg-n}\ R\ S\ X\ (p \pm q) \leq n$
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-deg-add4*:$[\![p \in \textit{carrier}\ R;\ q \in \textit{carrier}\ R;$
$(\textit{deg}\ R\ S\ X\ p) \leq (\textit{an}\ n);\ (\textit{deg}\ R\ S\ X\ q) \leq (\textit{an}\ n)]\!]\ \Longrightarrow$
$\textit{deg}\ R\ S\ X\ (p \pm q) \leq (\textit{an}\ n)$
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-deg-add5*:$[\![p \in \textit{carrier}\ R;\ q \in \textit{carrier}\ R;$
$(\textit{deg}\ R\ S\ X\ p) \leq a;\ (\textit{deg}\ R\ S\ X\ q) \leq a]\!]\ \Longrightarrow$

$$deg\ R\ S\ X\ (p \pm q) \leq a$$

⟨*proof*⟩

lemma (**in** *PolynRg*) *lower-deg-part*:⟦$p \in carrier\ R$; $p \neq \mathbf{0}$; $0 < deg\text{-}n\ R\ S\ X\ p$⟧
$\Longrightarrow$
$deg\ R\ S\ X\ (polyn\text{-}expr\ R\ X\ (deg\text{-}n\ R\ S\ X\ p - Suc\ 0)(SOME\ f.\ cf\text{-}sol\ R\ S\ X\ p\ f))$
$< deg\ R\ S\ X\ p$
⟨*proof*⟩

definition
ldeg-p :: [('*a*, '*m*) *Ring-scheme*, ('*a*, '*m1*) *Ring-scheme*, '*a*, *nat*, '*a*]
$\Rightarrow$ '*a* **where**
ldeg-p R S X d p = *polyn-expr R X d* (*scf-d R S X p* (*Suc d*))

definition
hdeg-p :: [('*a*, '*m*) *Ring-scheme*, ('*a*, '*m1*) *Ring-scheme*, '*a*, *nat*, '*a*]
$\Rightarrow$ '*a* **where**
hdeg-p R S X d p = (*snd* (*scf-d R S X p d*) *d*) $\cdot_{r\,R}$ ($X^{\frown R}$ *d*)

lemma (**in** *PolynRg*) *ldeg-p-mem*:⟦$p \in carrier\ R$; $deg\ R\ S\ X\ p \leq an\ (Suc\ d)$ ⟧ $\Longrightarrow$
ldeg-p R S X d p $\in$ *carrier R*
⟨*proof*⟩

lemma (**in** *PolynRg*) *ldeg-p-zero*:$p = \mathbf{0}_R \Longrightarrow$ *ldeg-p R S X d p* = $\mathbf{0}_R$
⟨*proof*⟩

lemma (**in** *PolynRg*) *hdeg-p-mem*:⟦$p \in carrier\ R$; $deg\ R\ S\ X\ p \leq an\ (Suc\ d)$⟧ $\Longrightarrow$
hdeg-p R S X (*Suc d*) *p* $\in$ *carrier R*
⟨*proof*⟩

lemma (**in** *PolynRg*) *hdeg-p-zero*:$p = \mathbf{0} \Longrightarrow$ *hdeg-p R S X* (*Suc d*) *p* = $\mathbf{0}$
⟨*proof*⟩

lemma (**in** *PolynRg*) *decompos-p*:⟦$p \in carrier\ R$; $deg\ R\ S\ X\ p \leq an\ (Suc\ d)$⟧ $\Longrightarrow$
$p = (ldeg\text{-}p\ R\ S\ X\ d\ p) \pm (hdeg\text{-}p\ R\ S\ X\ (Suc\ d)\ p)$
⟨*proof*⟩

lemma (**in** *PolynRg*) *deg-ldeg-p*:⟦$p \in carrier\ R$; $deg\ R\ S\ X\ p \leq an\ (Suc\ d)$⟧ $\Longrightarrow$
$deg\ R\ S\ X\ (ldeg\text{-}p\ R\ S\ X\ d\ p) \leq an\ d$

⟨*proof*⟩

lemma (**in** *PolynRg*) *deg-minus-eq*:⟦$p \in$ *carrier* R; $p \neq \mathbf{0}$⟧ ⟹
deg-n R S X ($-_a$ p) = *deg-n* R S X p
⟨*proof*⟩

lemma (**in** *PolynRg*) *deg-minus-eq1*:$p \in$ *carrier* R ⟹
deg R S X ($-_a$ p) = *deg* R S X p
⟨*proof*⟩

lemma (**in** *PolynRg*) *ldeg-p-pOp*:⟦$p \in$ *carrier* R; $q \in$ *carrier* R;
deg R S X $p \leq$ *an* (*Suc* d); *deg* R S X $q \leq$ *an* (*Suc* d)⟧ ⟹
(*ldeg-p* R S X d p) $\pm_R$ (*ldeg-p* R S X d q) =
ldeg-p R S X d ($p \pm_R q$)
⟨*proof*⟩

lemma (**in** *PolynRg*) *hdeg-p-pOp*:⟦$p \in$ *carrier* R; $q \in$ *carrier* R;
deg R S X $p \leq$ *an* (*Suc* d); *deg* R S X $q \leq$ *an* (*Suc* d)⟧ ⟹
(*hdeg-p* R S X (*Suc* d) p) $\pm$ (*hdeg-p* R S X (*Suc* d) q) =
hdeg-p R S X (*Suc* d) ($p \pm q$)
⟨*proof*⟩

lemma (**in** *PolynRg*) *ldeg-p-mOp*:⟦$p \in$ *carrier* R; *deg* R S X $p \leq$ *an* (*Suc* d)⟧ ⟹
$-_a$ (*ldeg-p* R S X d p) = *ldeg-p* R S X d ($-_a$ p)
⟨*proof*⟩

lemma (**in** *PolynRg*) *hdeg-p-mOp*:⟦$p \in$ *carrier* R;*deg* R S X $p \leq$ *an* (*Suc* d)⟧
⟹ $-_a$ (*hdeg-p* R S X (*Suc* d) p) = *hdeg-p* R S X (*Suc* d) ($-_a$ p)
⟨*proof*⟩

4.14.1 Multiplication of polynomials

lemma (**in** *PolynRg*) *deg-mult-pols*:⟦*Idomain* S;
$p \in$ *carrier* R; $p \neq \mathbf{0}$; $q \in$ *carrier* R; $q \neq \mathbf{0}$ ⟧ ⟹
$p \cdot_r q \neq \mathbf{0} \wedge$
deg-n R S X ($p \cdot_r q$) = *deg-n* R S X p + *deg-n* R S X q
⟨*proof*⟩

lemma (**in** *PolynRg*) *deg-mult-pols1*:⟦*Idomain* S; $p \in$ *carrier* R; $q \in$ *carrier* R⟧
⟹
deg R S X ($p \cdot_r q$) = *deg* R S X p + *deg* R S X q
⟨*proof*⟩

lemma (**in** *PolynRg*) *const-times-polyn*:⟦*Idomain* S; $c \in$ *carrier* S; $c \neq \mathbf{0}_S$;
$p \in$ *carrier* R; $p \neq \mathbf{0}$⟧ ⟹ ($c \cdot_r p$) $\neq \mathbf{0}$ $\wedge$
deg-n R S X ($c \cdot_r p$) = *deg-n* R S X p
⟨*proof*⟩

lemma (**in** *PolynRg*) *p-times-monomial-nonzero*:$[\![p \in carrier\ R;\ p \neq \mathbf{0}]\!] \Longrightarrow$
$(X^{\frown R}\ j)\ \cdot_r\ p \neq \mathbf{0}$
⟨*proof*⟩

lemma (**in** *PolynRg*) *p-times-monomial-nonzero1*:$[\![Idomain\ S;\ p \in carrier\ R;$
$p \neq \mathbf{0};\ c \in carrier\ S;\ c \neq \mathbf{0}_S]\!] \Longrightarrow (c\ \cdot_r\ (X^{\frown R}\ j))\ \cdot_r\ p \neq \mathbf{0}$
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-ring-integral*:$Idomain\ S = Idomain\ R$
⟨*proof*⟩

lemma (**in** *PolynRg*) *deg-to-X-d*:$Idomain\ S \Longrightarrow\ deg\text{-}n\ R\ S\ X\ (X^{\frown R}\ d) = d$
⟨*proof*⟩

4.14.2 Degree with value in *aug-minf*

lemma (**in** *PolynRg*) *nonzero-deg-pos*:$[\![p \in carrier\ R;\ p \neq \mathbf{0}]\!] \Longrightarrow$
$0 \leq deg\ R\ S\ X\ p$
⟨*proof*⟩

lemma (**in** *PolynRg*) *deg-minf-pol-0*:$p \in carrier\ R \Longrightarrow$
$(deg\ R\ S\ X\ p = -\infty) = (p = \mathbf{0})$
⟨*proof*⟩

lemma (**in** *PolynRg*) *pol-nonzero*:$p \in carrier\ R \Longrightarrow$
$(0 \leq deg\ R\ S\ X\ p) = (p \neq \mathbf{0})$
⟨*proof*⟩

lemma (**in** *PolynRg*) *minus-deg-in-aug-minf*:$[\![p \in carrier\ R;\ p \neq \mathbf{0}]\!] \Longrightarrow$
$-\ (deg\ R\ S\ X\ p) \in Z_{-\infty}$
⟨*proof*⟩

lemma (**in** *PolynRg*) *deg-of-X*:$deg\ R\ S\ X\ X = 1$
⟨*proof*⟩

lemma (**in** *PolynRg*) *pol-deg-0*:$[\![p \in carrier\ R;\ p \neq \mathbf{0}]\!]$
$\Longrightarrow\ (deg\ R\ S\ X\ p = 0) = (p \in carrier\ S)$
⟨*proof*⟩

lemma (**in** *PolynRg*) *deg-of-X2n*:$Idomain\ S \Longrightarrow deg\ R\ S\ X\ (X^{\frown R}\ n) = an\ n$
⟨*proof*⟩

lemma (**in** *PolynRg*) *add-pols-nonzero*:$[\![p \in carrier\ R;\ q \in carrier\ R;$
$(deg\ R\ S\ X\ p) \neq (deg\ R\ S\ X\ q)]\!] \Longrightarrow\ p \pm q \neq \mathbf{0}$
⟨*proof*⟩

lemma (**in** *PolynRg*) *deg-pols-add1*:$[\![p \in carrier\ R;\ q \in carrier\ R;$
$(deg\ R\ S\ X\ p) < (deg\ R\ S\ X\ q)]\!] \Longrightarrow$

$$deg\ R\ S\ X\ (p \pm q) = deg\ R\ S\ X\ q$$

⟨*proof*⟩

lemma (**in** *PolynRg*) *deg-pols-add2*:⟦$p \in carrier\ R$; $q \in carrier\ R$;
$(deg\ R\ S\ X\ p) = (deg\ R\ S\ X\ q)$⟧ ⟹
$deg\ R\ S\ X\ (p \pm q) \leq (deg\ R\ S\ X\ q)$

⟨*proof*⟩

lemma (**in** *PolynRg*) *deg-pols-add3*:⟦$p \in carrier\ R$; $q \in carrier\ R$;
$(deg\ R\ S\ X\ p) \leq an\ n$; $(deg\ R\ S\ X\ q) \leq an\ n$⟧ ⟹
$deg\ R\ S\ X\ (p \pm q) \leq an\ n$

⟨*proof*⟩

lemma (**in** *PolynRg*) *const-times-polyn1*:⟦*Idomain S*; $p \in carrier\ R$; $c \in carrier\ S$;
$c \neq \mathbf{0}_S$⟧ ⟹ $deg\ R\ S\ X\ (c \cdot_r p) = deg\ R\ S\ X\ p$

⟨*proof*⟩

4.15 Homomorphism of polynomial rings

definition
cf-h :: $('a \Rightarrow 'b) \Rightarrow nat \times (nat \Rightarrow 'a) \Rightarrow nat \times (nat \Rightarrow 'b)$ **where**
$cf\text{-}h\ f = (\lambda c.\ (fst\ c,\ cmp\ f\ (snd\ c)))$

definition
polyn-Hom :: [('a, 'm) *Ring-scheme*, ('a, 'm1) *Ring-scheme*, 'a,
('b, 'n) *Ring-scheme*, ('b, 'n1) *Ring-scheme*, 'b] ⇒
('a ⇒ 'b) *set*
(‹(*pHom* - - -, - - -)› [67,67,67,67,67,68]67) **where**
$pHom\ R\ S\ X,\ A\ B\ Y = \{f.\ f \in rHom\ R\ A \wedge f`(carrier\ S) \subseteq carrier\ B \wedge$
$f\ X = Y\}$

definition
erh :: [('a, 'm) *Ring-scheme*, ('a, 'm1) *Ring-scheme*, 'a,
('b, 'n) *Ring-scheme*, ('b, 'n1) *Ring-scheme*, 'b, 'a ⇒ 'b,
nat, $nat \times (nat \Rightarrow 'a)$] ⇒ 'b **where**
$erh\ R\ S\ X\ A\ B\ Y\ f\ n\ c = polyn\text{-}expr\ A\ Y\ n\ (cf\text{-}h\ f\ c)$

lemma (**in** *PolynRg*) *cf-h-len*:⟦*PolynRg A B Y*; $f \in rHom\ S\ B$;
pol-coeff S c⟧ ⟹ $fst\ (cf\text{-}h\ f\ c) = fst\ c$

⟨*proof*⟩

lemma (**in** *PolynRg*) *cf-h-coeff*:⟦*PolynRg A B Y*; $f \in rHom\ S\ B$;
pol-coeff S c⟧ ⟹ *pol-coeff B* $(cf\text{-}h\ f\ c)$

⟨*proof*⟩

lemma (**in** *PolynRg*) *cf-h-cmp*:⟦*PolynRg A B Y*; *pol-coeff S* (n, f); $h \in rHom\ S\ B$;
$j \leq n$⟧ ⟹

$(snd\ (cf\text{-}h\ h\ (n,\ f)))\ j = (cmp\ h\ f)\ j$
⟨*proof*⟩

lemma (**in** *PolynRg*) *cf-h-special-cf*:⟦*PolynRg A B Y*; $h \in rHom\ S\ B$⟧ ⟹
polyn-expr A Y (*Suc 0*) (*cf-h h* (*ext-cf S* (*Suc 0*) (C_0 *S*))) =
polyn-expr A Y (*Suc 0*) (*ext-cf B* (*Suc 0*) (C_0 *B*))
⟨*proof*⟩

lemma (**in** *PolynRg*) *polyn-Hom-coeff-to-coeff*:
⟦*PolynRg A B Y*; $f \in pHom\ R\ S\ X$, *A B Y*; *pol-coeff S c*⟧
⟹ *pol-coeff B* (*cf-h f c*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *cf-h-len1*:⟦*PolynRg A B Y*; $h \in rHom\ S\ B$;
$f \in pHom\ R\ S\ X$, *A B Y*; $\forall x{\in}carrier\ S.\ f\ x = h\ x$; *pol-coeff S c*⟧ ⟹
fst (*cf-h f c*) = *fst* (*cf-h h c*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *cf-h-len2*:⟦*PolynRg A B Y*; $f \in pHom\ R\ S\ X$, *A B Y*;
pol-coeff S c⟧ ⟹ *fst* (*cf-h f c*) = *fst c*
⟨*proof*⟩

lemma (**in** *PolynRg*) *cmp-pol-coeff*:⟦$f \in rHom\ S\ B$;
pol-coeff S (*n*, *c*)⟧ ⟹ *pol-coeff B* (*n*,(*cmp f c*))
⟨*proof*⟩

lemma (**in** *PolynRg*) *cmp-pol-coeff-e*:⟦*PolynRg A B Y*; $f \in pHom\ R\ S\ X$, *A B Y*;

pol-coeff S (*n*, *c*)⟧ ⟹ *pol-coeff B* (*n*, (*cmp f c*))
⟨*proof*⟩

lemma (**in** *PolynRg*) *cf-h-pol-coeff*:⟦*PolynRg A B Y*; $h \in rHom\ S\ B$;
pol-coeff S (*n*, *f*)⟧ ⟹ *cf-h h* (*n*, *f*) = (*n*, *cmp h f*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *cf-h-polyn*:⟦*PolynRg A B Y*; $h \in rHom\ S\ B$;
pol-coeff S (*n*, *f*)⟧ ⟹
polyn-expr A Y n (*cf-h h* (*n*, *f*)) = *polyn-expr A Y n* (*n*, (*cmp h f*))
⟨*proof*⟩

lemma (**in** *PolynRg*) *pHom-rHom*:⟦*PolynRg A B Y*; $f \in pHom\ R\ S\ X$, *A B Y*⟧
⟹
$f \in rHom\ R\ A$
⟨*proof*⟩

lemma (**in** *PolynRg*) *pHom-X-Y*:⟦*PolynRg A B Y*; $f \in pHom\ R\ S\ X$, *A B Y*⟧
⟹
$f\ X = Y$
⟨*proof*⟩

lemma (**in** *PolynRg*) *pHom-memTr*:⟦*PolynRg A B Y*;
f ∈ *pHom R S X*, *A B Y*⟧ ⟹
∀ *c*. *pol-coeff S* (*n*, *c*) ⟶
f (*polyn-expr R X n* (*n*, *c*)) = *polyn-expr A Y n* (*n*, *cmp f c*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *pHom-mem*:⟦*PolynRg A B Y*;
f ∈ *pHom R S X*, *A B Y*; *pol-coeff S* (*n*, *c*)⟧ ⟹
f (*polyn-expr R X n* (*n*, *c*)) = *polyn-expr A Y n* (*n*, *cmp f c*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *pHom-memc*:⟦*PolynRg A B Y*; *f* ∈ *pHom R S X*, *A B Y*;
pol-coeff S c⟧ ⟹
f (*polyn-expr R X* (*fst c*) *c*) = *polyn-expr A Y* (*fst c*) (*cf-h f c*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *pHom-mem1*:⟦*PolynRg A B Y*; *f* ∈ *pHom R S X*, *A B Y*;
p ∈ *carrier R*⟧ ⟹ *f p* ∈ *carrier A*
⟨*proof*⟩

lemma (**in** *PolynRg*) *pHom-pol-mem*:⟦*PolynRg A B Y*; *f* ∈ *pHom R S X*, *A B Y*;

p ∈ *carrier R*; *p* ≠ **0**⟧ ⟹
f p = *polyn-expr A Y* (*deg-n R S X p*) (*cf-h f* (*s-cf R S X p*))
⟨*proof*⟩

lemma (**in** *PolynRg*) *erh-rHom-coeff*:⟦*PolynRg A B Y*; *h* ∈ *rHom S B*;
pol-coeff S c⟧ ⟹ *erh R S X A B Y h 0 c* = (*cmp h* (*snd c*)) *0*
⟨*proof*⟩

lemma (**in** *PolynRg*) *erh-polyn-exprs*:⟦*PolynRg A B Y*; *h* ∈ *rHom S B*;
pol-coeff S c; *pol-coeff S d*;
polyn-expr R X (*fst c*) *c* = *polyn-expr R X* (*fst d*) *d*⟧ ⟹
erh R S X A B Y h (*fst c*) *c* = *erh R S X A B Y h* (*fst d*) *d*
⟨*proof*⟩

definition
erH :: [('*a*, '*m*) *Ring-scheme*, ('*a*, '*m1*) *Ring-scheme*, '*a*,
('*b*, '*n*) *Ring-scheme*, ('*b*, '*n1*) *Ring-scheme*, '*b*, '*a* ⇒ '*b*] ⇒
'*a* ⇒ '*b* **where**
erH R S X A B Y h = (λ*x*∈*carrier R*. *erh R S X A B Y h*
(*fst* (*s-cf R S X x*)) (*s-cf R S X x*))

lemma (**in** *PolynRg*) *erH-rHom-0*:⟦*PolynRg A B Y*; *h* ∈ *rHom S B*⟧ ⟹
(*erH R S X A B Y h*) **0** = $\mathbf{0}_A$
⟨*proof*⟩

lemma (**in** *PolynRg*) *erH-mem*:⟦*PolynRg A B Y*; $h \in$ *rHom S B*; $p \in$ *carrier R*⟧
$\Longrightarrow$
erH R S X A B Y h p $\in$ *carrier A*
⟨*proof*⟩

lemma (**in** *PolynRg*) *erH-rHom-nonzero*:⟦*PolynRg A B Y*; $f \in$ *rHom S B*;
$p \in$ *carrier R*; (*erH R S X A B Y f*) $p \neq \mathbf{0}_A$⟧ $\Longrightarrow p \neq \mathbf{0}$
⟨*proof*⟩

lemma (**in** *PolynRg*) *erH-rHomTr2*:⟦*PolynRg A B Y*; $h \in$ *rHom S B*⟧ $\Longrightarrow$
(*erH R S X A B Y h*) $(1_r) = (1_{r\,A})$
⟨*proof*⟩

declare *max.absorb1* [*simp*] *max.absorb2* [*simp*]

lemma (**in** *PolynRg*) *erH-multTr*:⟦*PolynRg A B Y*; $h \in$ *rHom S B*;
pol-coeff S c⟧ $\Longrightarrow$
$\forall f\ g.$ *pol-coeff S* $(m, f) \wedge$ *pol-coeff S* $(((fst\ c) + m), g) \wedge$
(*polyn-expr R X* (*fst c*) *c*) $\cdot_r$ (*polyn-expr R X m* (m, f)) =
(*polyn-expr R X* $((fst\ c) + m)$ $((fst\ c) + m, g))$ $\longrightarrow$
(*polyn-expr A Y* (*fst c*) (*cf-h h c*)) $\cdot_{r\,A}$ (*polyn-expr A Y m* (*cf-h h* (m, f))) =
(*polyn-expr A Y* $((fst\ c) + m)$ (*cf-h h* $((fst\ c)+m, g)$))
⟨*proof*⟩

lemma (**in** *PolynRg*) *erH-multTr1*:⟦*PolynRg A B Y*; $h \in$ *rHom S B*;
pol-coeff S c; *pol-coeff S d*; *pol-coeff S e*; *fst e* = *fst c* + *fst d*;
(*polyn-expr R X* (*fst c*) *c*) $\cdot_r$ (*polyn-expr R X* (*fst d*) *d*) =
polyn-expr R X $((fst\ c) + (fst\ d))$ *e* ⟧ $\Longrightarrow$
(*polyn-expr A Y* (*fst c*) (*cf-h h c*)) $\cdot_{r\,A}$ (*polyn-expr A Y* (*fst d*) (*cf-h h d*))
= (*polyn-expr A Y* (*fst e*) (*cf-h h e*))
⟨*proof*⟩

lemma (**in** *PolynRg*) *erHomTr0*:⟦*PolynRg A B Y*; $h \in$ *rHom S B*; $x \in$ *carrier R*⟧
$\Longrightarrow$ *erH R S X A B Y h* $(-_a\ x) = -_{a\,A}$ (*erH R S X A B Y h x*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *erHomTr1*:⟦*PolynRg A B Y*; $h \in$ *rHom S B*;
$a \in$ *carrier R*; $b \in$ *carrier R*; $a \neq \mathbf{0}$; $b \neq \mathbf{0}$; $a \pm b \neq \mathbf{0}$;
deg-n R S X a = *deg-n R S X b*⟧ $\Longrightarrow$
erH R S X A B Y h $(a \pm b) =$ *erH R S X A B Y h a* $\pm_A$
(*erH R S X A B Y h b*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *erHomTr2*:⟦*PolynRg A B Y*; $h \in$ *rHom S B*;
$a \in$ *carrier R*; $b \in$ *carrier R*; $a \neq \mathbf{0}$; $b \neq \mathbf{0}$; $a \pm b \neq \mathbf{0}$;
deg-n R S X a < *deg-n R S X b*⟧ $\Longrightarrow$
erH R S X A B Y h $(a \pm b) =$ *erH R S X A B Y h a* $\pm_A$

$(erH\ R\ S\ X\ A\ B\ Y\ h\ b)$

⟨*proof*⟩

lemma (**in** *PolynRg*) *erH-rHom*:⟦*Idomain S*; *PolynRg A B Y*; $h \in rHom\ S\ B$⟧
$\Longrightarrow erH\ R\ S\ X\ A\ B\ Y\ h \in pHom\ R\ S\ X,\ A\ B\ Y$
⟨*proof*⟩

lemma (**in** *PolynRg*) *erH-q-rHom*:⟦*Idomain S*; *maximal-ideal S P*;
PolynRg $R'\ (S\ /_r\ P)\ Y$⟧ $\Longrightarrow$
$erH\ R\ S\ X\ R'\ (S\ /_r\ P)\ Y\ (pj\ S\ P) \in pHom\ R\ S\ X,\ R'\ (S\ /_r\ P)\ Y$
⟨*proof*⟩

lemma (**in** *PolynRg*) *erH-add*:⟦*Idomain S*; *PolynRg A B Y*; $h \in rHom\ S\ B$;
$p \in carrier\ R$; $q \in carrier\ R$⟧ $\Longrightarrow$
$erH\ R\ S\ X\ A\ B\ Y\ h\ (p \pm q) =$
$(erH\ R\ S\ X\ A\ B\ Y\ h\ p) \pm_A (erH\ R\ S\ X\ A\ B\ Y\ h\ q)$
⟨*proof*⟩

lemma (**in** *PolynRg*) *erH-minus*:⟦*Idomain S*; *PolynRg A B Y*;
$h \in rHom\ S\ B$; $p \in carrier\ R$⟧ $\Longrightarrow$
$erH\ R\ S\ X\ A\ B\ Y\ h\ (-_a\ p) = -_{a\,A}\ (erH\ R\ S\ X\ A\ B\ Y\ h\ p)$
⟨*proof*⟩

lemma (**in** *PolynRg*) *erH-mult*:⟦*Idomain S*; *PolynRg A B Y*; $h \in rHom\ S\ B$;
$p \in carrier\ R$; $q \in carrier\ R$⟧ $\Longrightarrow$
$erH\ R\ S\ X\ A\ B\ Y\ h\ (p \cdot_r q) =$
$(erH\ R\ S\ X\ A\ B\ Y\ h\ p) \cdot_{r\,A} (erH\ R\ S\ X\ A\ B\ Y\ h\ q)$
⟨*proof*⟩

lemma (**in** *PolynRg*) *erH-rHom-cf*:⟦*Idomain S*; *PolynRg A B Y*; $h \in rHom\ S\ B$;
$s \in carrier\ S$⟧ $\Longrightarrow erH\ R\ S\ X\ A\ B\ Y\ h\ s = h\ s$
⟨*proof*⟩

lemma (**in** *PolynRg*) *erH-rHom-coeff*:⟦*Idomain S*; *PolynRg A B Y*; $h \in rHom\ S\ B$;
pol-coeff S (n, f)⟧ $\Longrightarrow$ *pol-coeff B* $(n,\ cmp\ h\ f)$
⟨*proof*⟩

lemma (**in** *PolynRg*) *erH-rHom-unique*:⟦*Idomain S*; *PolynRg A B Y*; $h \in rHom\ S\ B$⟧
$\Longrightarrow\ \exists!g.\ g \in pHom\ R\ S\ X,\ A\ B\ Y \wedge (\forall x{\in}carrier\ S.\ h\ x = g\ x)$
⟨*proof*⟩

lemma (**in** *PolynRg*) *erH-rHom-unique1*:⟦*Idomain S*; *PolynRg A B Y*; $h \in rHom\ S\ B$;
$f \in pHom\ R\ S\ X,\ A\ B\ Y$; $\forall x \in carrier\ S.\ f\ x = h\ x$⟧ $\Longrightarrow$
$f = (erH\ R\ S\ X\ A\ B\ Y\ h)$
⟨*proof*⟩

lemma (**in** *PolynRg*) *pHom-dec-deg*:⟦*PolynRg A B Y*; $f \in$ *pHom R S X*, *A B Y*;
$p \in$ *carrier R*⟧ ⟹
deg A B Y (*f p*) ≤ *deg R S X p*
⟨*proof*⟩

lemma (**in** *PolynRg*) *erH-map*:⟦*Idomain S*; *PolynRg A B Y*; $h \in$ *rHom S B*;
pol-coeff S (*n*, *c*)⟧ ⟹
(*erH R S X A B Y h*) (*polyn-expr R X n* (*n*, *c*)) =
polyn-expr A Y n (*n*, (*cmp h c*))
⟨*proof*⟩

4.16 Relatively prime polynomials

definition
rel-prime-pols :: [('*a*, '*m*) *Ring-scheme*, ('*a*, '*m1*) *Ring-scheme*, '*a*,
'*a*, '*a*] ⇒ *bool* **where**
rel-prime-pols R S X p q ⟷ ($1_{r\,R}$) ∈ ((*Rxa R p*) $\mp_R$ (*Rxa R q*))

definition
div-condn :: [('*a*, '*m*) *Ring-scheme*, ('*a*, '*m1*) *Ring-scheme*, '*a*, *nat*,
'*a*, '*a*] ⇒ *bool* **where**
div-condn R S X n g f ⟷ *f* ∈ *carrier R* ∧ *n* = *deg-n R S X f* ⟶
(∃ *q*. *q* ∈ *carrier R* ∧ ((*f* $\pm_R$ ($-_a R$ (*q* $\cdot_{r\,R}$ *g*)) = $\mathbf{0}_R$) ∨ (*deg-n R S X*
(*f* $\pm_R$ ($-_a R$ (*q* $\cdot_{r\,R}$ *g*))) < *deg-n R S X g*)))

lemma (**in** *PolynRg*) *divisionTr0*:⟦*Idomain S*; *p* ∈ *carrier R*;
c ∈ *carrier S*; *c* ≠ $\mathbf{0}_S$⟧ ⟹
lcf R S X (*c* $\cdot_r$ $X^{\frown R\ n}$ $\cdot_r$ *p*) = *c* $\cdot_{r\,S}$ (*lcf R S X p*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *divisionTr1*:⟦*Corps S*; *g* ∈ *carrier R*; *g* ≠ **0**;
0 < *deg-n R S X g*; *f* ∈ *carrier R*; *f* ≠ **0**; *deg-n R S X g* ≤ *deg-n R S X f*⟧
⟹
f ± $-_a$ ((*lcf R S X f*) $\cdot_{r\,S}$ ((*lcf R S X g*)$^{-\ S}$) $\cdot_r$
($X^{\frown R}$ ((*deg-n R S X f*) − (*deg-n R S X g*))) $\cdot_r$ *g*) = **0** ∨
deg-n R S X (*f* ± $-_a$ ((*lcf R S X f*) $\cdot_{r\,S}$ ((*lcf R S X g*)$^{-\ S}$) $\cdot_r$
($X^{\frown R}$ ((*deg-n R S X f*) − (*deg-n R S X g*))) $\cdot_r$ *g*)) < *deg-n R S X f*
⟨*proof*⟩

lemma (**in** *PolynRg*) *divisionTr2*:⟦*Corps S*; *g* ∈ *carrier R*; *g* ≠ **0**;
0 < *deg-n R S X g*⟧ ⟹ ∀ *f*. *div-condn R S X n g f*
⟨*proof*⟩

lemma (**in** *PolynRg*) *divisionTr3*:⟦*Corps S*; *g* ∈ *carrier R*; *g* ≠ **0**;
0 < *deg-n R S X g*; *f* ∈ *carrier R*⟧ ⟹
∃ *q*∈*carrier R*. (*f* ± $-_a$ (*q* $\cdot_r$ *g*) = **0**) ∨ (*f* ± $-_a$ (*q* $\cdot_r$ *g*) ≠ **0** ∧
deg-n R S X (*f* ± $-_a$ (*q* $\cdot_r$ *g*)) < (*deg-n R S X g*))
⟨*proof*⟩

lemma (**in** *PolynRg*) *divisionTr4*:⟦*Corps S*; $g \in$ *carrier R*; $g \neq \mathbf{0}$;
$0 <$ *deg-n R S X g*; $f \in$ *carrier R*⟧ $\Longrightarrow$
$\exists\, q{\in}$*carrier R*. $(f = q \cdot_r g) \vee (\exists\, r{\in}$*carrier R*. $r \neq \mathbf{0} \wedge (f = (q \cdot_r g) \pm r)$
$\wedge$ (*deg-n R S X r*) $<$ (*deg-n R S X g*))
⟨*proof*⟩

lemma (**in** *PolynRg*) *divisionTr*:⟦*Corps S*; $g \in$ *carrier R*; $0 <$ *deg R S X g*;
$f \in$ *carrier R*⟧ $\Longrightarrow$
$\exists\, q{\in}$*carrier R*. $(\exists\, r{\in}$*carrier R*. $(f = (q \cdot_r g) \pm r) \wedge$
(*deg R S X r*) $<$ (*deg R S X g*))
⟨*proof*⟩

lemma (**in** *PolynRg*) *rel-prime-equation*:⟦*Corps S*; $f \in$ *carrier R*; $g \in$ *carrier R*;
$0 <$ *deg R S X f*; $0 <$ *deg R S X g*; *rel-prime-pols R S X f g*;
$h \in$ *carrier R*⟧ $\Longrightarrow$
$\exists\, u \in$ *carrier R*. $\exists\, v \in$ *carrier R*.
(*deg R S X u*) $\leq$ *amax* ((*deg R S X h*) $-$ (*deg R S X f*)) (*deg R S X g*)) $\wedge$
(*deg R S X v*) $\leq$ (*deg R S X f*)) $\wedge$ ($u \cdot_r f \pm (v \cdot_r g) = h$)
⟨*proof*⟩

4.16.1 Polynomial, coeff mod P

definition
P-mod :: [(*'a*, *'m*) *Ring-scheme*, (*'a*, *'m1*) *Ring-scheme*, *'a*, *'a set*,
'a] $\Rightarrow$ *bool* **where**
P-mod R S X P p $\longleftrightarrow$ $p = \mathbf{0}_R \vee$
$(\forall\, j \leq$ (*fst* (*s-cf R S X p*)). (*snd* (*s-cf R S X p*) *j*) $\in P$)

lemma (**in** *PolynRg*) *P-mod-whole*:$p \in$ *carrier R* $\Longrightarrow$
P-mod R S X (*carrier S*) *p*
⟨*proof*⟩

lemma (**in** *PolynRg*) *zero-P-mod*:*ideal S I* $\Longrightarrow$ *P-mod R S X I* $\mathbf{0}$
⟨*proof*⟩

lemma (**in** *PolynRg*) *P-mod-mod*:⟦*ideal S I*; $p \in$ *carrier R*; *pol-coeff S c*;
$p =$ *polyn-expr R X* (*fst c*) *c*⟧ $\Longrightarrow$
$(\forall\, j \leq$ (*fst c*). (*snd c*) $j \in I) =$ (*P-mod R S X I p*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *monomial-P-mod-mod*:⟦*ideal S I*; $c \in$ *carrier S*;
$p = c \cdot_r (X^{\frown_R}\, d)$⟧ $\Longrightarrow$ $(c \in I) =$ (*P-mod R S X I p*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *P-mod-add*:⟦*ideal S I*; $p \in$ *carrier R*;
$q \in$ *carrier R*; *P-mod R S X I p*; *P-mod R S X I q*⟧ $\Longrightarrow$
P-mod R S X I ($p \pm q$)
⟨*proof*⟩

lemma (**in** *PolynRg*) *P-mod-minus*:⟦*ideal S I*; *p* ∈ *carrier R*; *P-mod R S X I p*⟧
⟹
P-mod R S X I (−$_a$ *p*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *P-mod-pre*:⟦*ideal S I*; *pol-coeff S* ((*Suc n*), *f*);
P-mod R S X I (*polyn-expr R X* (*Suc n*) (*Suc n*, *f*))⟧ ⟹
P-mod R S X I (*polyn-expr R X n* (*n*, *f*))
⟨*proof*⟩

lemma (**in** *PolynRg*) *P-mod-pre1*:⟦*ideal S I*; *pol-coeff S* ((*Suc n*), *f*);
P-mod R S X I (*polyn-expr R X* (*Suc n*) (*Suc n*, *f*))⟧ ⟹
P-mod R S X I (*polyn-expr R X n* (*Suc n*, *f*))
⟨*proof*⟩

lemma (**in** *PolynRg*) *P-mod-coeffTr*:⟦*ideal S I*; *d* ∈ *carrier S*⟧ ⟹
(*P-mod R S X I d*) = (*d* ∈ *I*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *P-mod-mult-const*:⟦*ideal S I*; *ideal S J*;
pol-coeff S (*n*, *f*); *P-mod R S X I* (*polyn-expr R X n* (*n*, *f*));
pol-coeff S (*0*, *g*); *P-mod R S X J* (*polyn-expr R X 0* (*0*, *g*))⟧ ⟹
P-mod R S X (*I* ◇$_{rS}$ *J*) ((*polyn-expr R X n* (*n*, *f*)) ·$_r$
(*polyn-expr R X 0* (*0*, *g*)))
⟨*proof*⟩

lemma (**in** *PolynRg*) *P-mod-mult-const1*:⟦*ideal S I*; *ideal S J*;
pol-coeff S (*n*, *f*); *P-mod R S X I* (*polyn-expr R X n* (*n*, *f*));
d ∈ *J*⟧ ⟹
P-mod R S X (*I* ◇$_{rS}$ *J*) ((*polyn-expr R X n* (*n*, *f*)) ·$_r$ *d*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *P-mod-mult-monomial*:⟦*ideal S I*; *p* ∈ *carrier R*⟧ ⟹
(*P-mod R S X I p*) = (*P-mod R S X I* (*p* ·$_r$ *X*^^{R} *m*))
⟨*proof*⟩

lemma (**in** *PolynRg*) *P-mod-multTr*:⟦*ideal S I*; *ideal S J*; *pol-coeff S* (*n*, *f*);
P-mod R S X I (*polyn-expr R X n* (*n*, *f*))⟧ ⟹ ∀ *g*. ((*pol-coeff S* (*m*, *g*)
∧ (*P-mod R S X J* (*polyn-expr R X m* (*m*, *g*)))) ⟶
P-mod R S X (*I* ◇$_{rS}$ *J*)
((*polyn-expr R X n* (*n*, *f*)) ·$_r$ (*polyn-expr R X m* (*m*, *g*))))
⟨*proof*⟩

lemma (**in** *PolynRg*) *P-mod-mult*:⟦*ideal S I*; *ideal S J*; *pol-coeff S* (*n*, *c*);
pol-coeff S (*m*, *d*); *P-mod R S X I* (*polyn-expr R X n* (*n*, *c*));
P-mod R S X J (*polyn-expr R X m* (*m*, *d*))⟧ ⟹
P-mod R S X (*I* ◇$_{rS}$ *J*) ((*polyn-expr R X n* (*n*, *c*)) ·$_r$
(*polyn-expr R X m* (*m*, *d*)))

⟨*proof*⟩

lemma (**in** *PolynRg*) *P-mod-mult1*:⟦*ideal S I*; *ideal S J*;
p ∈ *carrier R*; *q* ∈ *carrier R*; *P-mod R S X I p*; *P-mod R S X J q*⟧ ⟹
P-mod R S X (*I* ◇$_{r\,S}$ *J*) (*p* ·$_r$ *q*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *P-mod-mult2l*:⟦*ideal S I*; *p* ∈ *carrier R*; *q* ∈ *carrier R*;
P-mod R S X I p⟧ ⟹ *P-mod R S X I* (*p* ·$_r$ *q*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *P-mod-mult2r*:⟦*ideal S I*; *p* ∈ *carrier R*; *q* ∈ *carrier R*;
P-mod R S X I q⟧ ⟹ *P-mod R S X I* (*p* ·$_r$ *q*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *csrp-fn-pol-coeff*:⟦*ideal S P*; *PolynRg R'* (*S* /$_r$ *P*) *Y*;
pol-coeff (*S* /$_r$ *P*) (*n*, *c'*)⟧ ⟹
pol-coeff S (*n*, (*cmp* (*csrp-fn S P*) *c'*))
⟨*proof*⟩

lemma (**in** *PolynRg*) *pj-csrp-mem-coeff*:⟦*ideal S P*; *pol-coeff* (*S* /$_r$ *P*) (*n*, *c'*)⟧
⟹ ∀ *j* ≤ *n*. (*pj S P*) ((*csrp-fn S P*) (*c' j*)) = *c' j*
⟨*proof*⟩

lemma (**in** *PolynRg*) *pHom-pj-csrp*:⟦*Idomain S*; *ideal S P*;
PolynRg R' (*S* /$_r$ *P*) *Y*; *pol-coeff* (*S* /$_r$ *P*) (*n*, *c'*)⟧ ⟹
erH R S X R' (*S* /$_r$ *P*) *Y* (*pj S P*)
(*polyn-expr R X n* (*n*, (*cmp* (*csrp-fn S P*) *c'*)))
= *polyn-expr R' Y n* (*n*, *c'*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *ext-csrp-fn-nonzero*:⟦*Idomain S*; *ideal S P*;
PolynRg R' (*S* /$_r$ *P*) *Y*; *g'* ∈ *carrier R'*; *g'* ≠ **0**$_{R'}$⟧ ⟹
polyn-expr R X (*deg-n R'* (*S* /$_r$ *P*) *Y g'*) ((*deg-n R'* (*S* /$_r$ *P*) *Y g'*),
(*cmp* (*csrp-fn S P*) (*snd* (*s-cf R'* (*S* /$_r$ *P*) *Y g'*)))) ≠ **0**
⟨*proof*⟩

lemma (**in** *PolynRg*) *erH-inv*:⟦*Idomain S*; *ideal S P*; *Ring R'*;
PolynRg R' (*S* /$_r$ *P*) *Y*; *g'* ∈ *carrier R'*⟧ ⟹
∃ *g*∈*carrier R*. *deg R S X g* ≤ (*deg R'* (*S* /$_r$ *P*) *Y g'*) ∧
(*erH R S X R'* (*S* /$_r$ *P*) *Y* (*pj S P*)) *g* = *g'*
⟨*proof*⟩

lemma (**in** *PolynRg*) *P-mod-0*:⟦*Idomain S*; *ideal S P*; *PolynRg R'* (*S* /$_r$ *P*) *Y*;
g ∈ *carrier R*⟧ ⟹
(*erH R S X R'* (*S* /$_r$ *P*) *Y* (*pj S P*) *g* = **0**$_{R'}$) = (*P-mod R S X P g*)
⟨*proof*⟩

lemma (**in** *PolynRg*) *P-mod-I-J*:⟦*p* ∈ *carrier R*; *ideal S I*; *ideal S J*;

$I \subseteq J;\ \ P\text{-}mod\ R\ S\ X\ I\ p\rrbracket \Longrightarrow P\text{-}mod\ R\ S\ X\ J\ p$
⟨*proof*⟩

lemma (**in** *PolynRg*) *P-mod-n-1*:⟦*Idomain S*; $t \in carrier\ S$; $g \in carrier\ R$;
$P\text{-}mod\ R\ S\ X\ (S \diamondsuit_p\ (t^{\frown S}\ (Suc\ n)))\ g\rrbracket \Longrightarrow P\text{-}mod\ R\ S\ X\ (S \diamondsuit_p\ t)\ g$
⟨*proof*⟩

lemma (**in** *PolynRg*) *P-mod-n-m*:⟦*Idomain S*; $t \in carrier\ S$; $g \in carrier\ R$;
$m \leq n$; $P\text{-}mod\ R\ S\ X\ (S \diamondsuit_p\ (t^{\frown S}\ (Suc\ n)))\ g\rrbracket \Longrightarrow$
$P\text{-}mod\ R\ S\ X\ (S \diamondsuit_p\ (t^{\frown S}\ (Suc\ m)))\ g$
⟨*proof*⟩

lemma (**in** *PolynRg*) *P-mod-diff*:⟦*Idomain S*; *ideal S P*; *PolynRg* $R'\ (S\ /_r\ P)\ Y$;
$g \in carrier\ R$; $h \in carrier\ R\rrbracket \Longrightarrow$
$(erH\ R\ S\ X\ R'\ (S\ /_r\ P)\ Y\ (pj\ S\ P)\ g = (erH\ R\ S\ X\ R'\ (S\ /_r\ P)\ Y\ (pj\ S\ P)\ h))$
$= (P\text{-}mod\ R\ S\ X\ P\ (g \pm -_a\ h))$
⟨*proof*⟩

lemma (**in** *PolynRg*) *P-mod-erH*:⟦*Idomain S*; *ideal S P*; *PolynRg* $R'\ (S\ /_r\ P)\ Y$;
$g \in carrier\ R$; $v \in carrier\ R$; $t \in P\ \rrbracket \Longrightarrow$
$(erH\ R\ S\ X\ R'\ (S\ /_r\ P)\ Y\ (pj\ S\ P)\ g =$
$(erH\ R\ S\ X\ R'\ (S\ /_r\ P)\ Y\ (pj\ S\ P)\ (g \pm (t \cdot_r v))))$
⟨*proof*⟩

lemma (**in** *PolynRg*) *coeff-principalTr*:⟦$t \in carrier\ S\rrbracket \Longrightarrow$
$\forall f.\ pol\text{-}coeff\ S\ (n, f) \wedge (\forall j \leq n.\ f\ j \in S \diamondsuit_p\ t) \longrightarrow$
$(\exists f'.\ pol\text{-}coeff\ S\ (n, f') \wedge (\forall j \leq n.\ f\ j = t \cdot_{rS} (f'\ j)))$
⟨*proof*⟩

lemma (**in** *PolynRg*) *coeff-principal*:⟦$t \in carrier\ S$; $pol\text{-}coeff\ S\ (n, f)$;
$\forall j \leq n.\ f\ j \in S \diamondsuit_p\ t\rrbracket \Longrightarrow$
$\exists f'.\ pol\text{-}coeff\ S\ (n, f') \wedge (\forall j \leq n.\ f\ j = t \cdot_{rS} (f'\ j))$
⟨*proof*⟩

lemma (**in** *PolynRg*) *Pmod-0-principal*:⟦*Idomain S*; $t \in carrier\ S$; $g \in carrier\ R$;
$P\text{-}mod\ R\ S\ X\ (S \diamondsuit_p\ t)\ g\rrbracket \Longrightarrow \exists h \in carrier\ R.\ g = t \cdot_r h$
⟨*proof*⟩

lemma (**in** *PolynRg*) *Pmod0-principal-rev*:⟦*Idomain S*; $t \in carrier\ S$;
$g \in carrier\ R$; $\exists h \in carrier\ R.\ g = t \cdot_r\ h\rrbracket \Longrightarrow$
$P\text{-}mod\ R\ S\ X\ (S \diamondsuit_p\ t)\ g$
⟨*proof*⟩

lemma (**in** *PolynRg*) *Pmod0-principal-rev1*:⟦*Idomain S*; $t \in carrier\ S$;
$h \in carrier\ R\rrbracket \Longrightarrow P\text{-}mod\ R\ S\ X\ (S \diamondsuit_p\ t)\ (t \cdot_r h)$
⟨*proof*⟩

lemma (**in** *PolynRg*) *Pmod0-principal-erH-vanish-t*:⟦*Idomain S*; *ideal S* (*S* ◇$_p$ *t*);
t ∈ *carrier S*; *t* ≠ **0**$_S$; *PolynRg R'* (*S* /$_r$ (*S* ◇$_p$ *t*)) *Y* ⟧ ⟹
erH R S X R' (*S* /$_r$ (*S* ◇$_p$ *t*)) *Y* (*pj S* (*S* ◇$_p$ *t*)) *t* = **0**$_{R'}$
⟨*proof*⟩

lemma (**in** *PolynRg*) *P-mod-diffxxx1*:⟦*Idomain S*; *t* ∈ *carrier S*; *t* ≠ **0**$_S$;
maximal-ideal S (*S* ◇$_p$ *t*); *PolynRg R'* (*S* /$_r$ (*S* ◇$_p$ *t*)) *Y*;
f ∈ *carrier R*; *g* ∈ *carrier R*; *h* ∈ *carrier R*;
f ≠ **0**; *g* ≠ **0**; *h* ≠ **0**; *u* ∈ *carrier R*; *v* ∈ *carrier R*;
erH R S X R' (*S* /$_r$ (*S* ◇$_p$ *t*)) *Y* (*pj S* (*S* ◇$_p$ *t*)) *g* ≠ **0**$_{R'}$;
erH R S X R' (*S* /$_r$ (*S* ◇$_p$ *t*)) *Y* (*pj S* (*S* ◇$_p$ *t*)) *h* ≠ **0**$_{R'}$;
ra ∈ *carrier R*;
f ± −$_a$ (*g* ·$_r$ *h*) = *t*^$^{S\ m}$ ·$_r$ *ra*; *0* < *m*;
(*erH R S X R'* (*S* /$_r$ (*S* ◇$_p$ *t*)) *Y* (*pj S* (*S* ◇$_p$ *t*)) *u*)
·$_{r\,R'}$ *erH R S X R'* (*S* /$_r$ (*S* ◇$_p$ *t*)) *Y* (*pj S* (*S* ◇$_p$ *t*)) *g* ±$_{R'}$
(*erH R S X R'* (*S* /$_r$ (*S* ◇$_p$ *t*)) *Y* (*pj S* (*S* ◇$_p$ *t*)) *v*)
·$_{r\,R'}$ *erH R S X R'* (*S* /$_r$ (*S* ◇$_p$ *t*)) *Y* (*pj S* (*S* ◇$_p$ *t*)) *h* =
erH R S X R' (*S* /$_r$ (*S* ◇$_p$ *t*)) *Y* (*pj S* (*S* ◇$_p$ *t*)) *ra*⟧
⟹ *P-mod R S X* (*S* ◇$_p$ (*t*^^{S} (*Suc m*)))
(*f* ± −$_a$ ((*g* ± *t*^$^{S\ m}$ ·$_r$ *v*) ·$_r$ (*h* ± *t*^$^{S\ m}$ ·$_r$ *u*)))
⟨*proof*⟩

lemma (**in** *PolynRg*) *P-mod-diffxxx2*:⟦*Idomain S*; *t* ∈ *carrier S*; *t* ≠ **0**$_S$;
maximal-ideal S (*S* ◇$_p$ *t*); *PolynRg R'* (*S* /$_r$ (*S* ◇$_p$ *t*)) *Y*;
f ∈ *carrier R*; *g* ∈ *carrier R*; *h* ∈ *carrier R*;
deg R S X g ≤ *deg R'* (*S* /$_r$ (*S* ◇$_p$ *t*)) *Y*
(*erH R S X R'* (*S* /$_r$ (*S* ◇$_p$ *t*)) *Y* (*pj S* (*S* ◇$_p$ *t*)) *g*);
deg R S X h +
deg R' (*S* /$_r$ (*S* ◇$_p$ *t*)) *Y* (*erH R S X R'* (*S* /$_r$ (*S* ◇$_p$ *t*)) *Y* (*pj S* (*S* ◇$_p$ *t*)) *g*)
≤ *deg R S X f*;
0 < *deg R'* (*S* /$_r$ (*S* ◇$_p$ *t*)) *Y*
(*erH R S X R'* (*S* /$_r$ (*S* ◇$_p$ *t*)) *Y* (*pj S* (*S* ◇$_p$ *t*)) *g*);
0 < *deg R'* (*S* /$_r$ (*S* ◇$_p$ *t*)) *Y*
(*erH R S X R'* (*S* /$_r$ (*S* ◇$_p$ *t*)) *Y* (*pj S* (*S* ◇$_p$ *t*)) *h*);
rel-prime-pols R' (*S* /$_r$ (*S* ◇$_p$ *t*)) *Y*
(*erH R S X R'* (*S* /$_r$ (*S* ◇$_p$ *t*)) *Y* (*pj S* (*S* ◇$_p$ *t*)) *g*)
(*erH R S X R'* (*S* /$_r$ (*S* ◇$_p$ *t*)) *Y* (*pj S* (*S* ◇$_p$ *t*)) *h*);
P-mod R S X (*S* ◇$_p$ (*t*^$^{S\ m}$)) (*f* ± −$_a$ (*g* ·$_r$ *h*)); *0* < *m*⟧ ⟹
∃ *g1 h1*. *g1* ∈*carrier R* ∧ *h1* ∈ *carrier R* ∧
(*deg R S X g1* ≤ *deg R'* (*S* /$_r$ (*S* ◇$_p$ *t*)) *Y*
(*erH R S X R'* (*S* /$_r$ (*S* ◇$_p$ *t*)) *Y* (*pj S* (*S* ◇$_p$ *t*)) *g1*)) ∧
P-mod R S X (*S* ◇$_p$ (*t*^$^{S\ m}$)) (*g* ± −$_a$ *g1*) ∧ (*deg R S X h1* +
deg R' (*S* /$_r$ (*S* ◇$_p$ *t*)) *Y* (*erH R S X R'* (*S* /$_r$ (*S* ◇$_p$ *t*)) *Y* (*pj S* (*S* ◇$_p$ *t*)) *g1*)
≤ *deg R S X f*) ∧
P-mod R S X (*S* ◇$_p$ (*t*^$^{S\ m}$)) (*h* ± −$_a$ *h1*) ∧
P-mod R S X (*S* ◇$_p$ (*t*^^{S} (*Suc m*))) (*f* ± (−$_a$ (*g1* ·$_r$ *h1*)))
⟨*proof*⟩

definition
Hensel-next :: [('a, 'b) *Ring-scheme*, ('a, 'c) *Ring-scheme*, 'a, 'a,
('a *set*, 'm) *Ring-scheme*, 'a *set*,'a, *nat*] ⇒ ('a × 'a) ⇒ ('a × 'a)
(‹(*9Hen - - - - - - - - -*)› [67,67,67,67,67,67,67,68]67) **where**

$Hen_{R\ S\ X\ t\ R'\ Y\ f}\ m\ gh = (SOME\ gh1.$
$gh1 \in carrier\ R \times carrier\ R \wedge$
$(deg\ R\ S\ X\ (fst\ gh1) \leq deg\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y$
$(erH\ R\ S\ X\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))\ (fst\ gh1))) \wedge$
$P\text{-}mod\ R\ S\ X\ (S \diamondsuit_p\ (t^{\frown S}\ m))\ ((fst\ gh) \pm_R -_a R\ (fst\ gh1)) \wedge$
$(deg\ R\ S\ X\ (snd\ gh1) + deg\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y\ (erH\ R\ S\ X\ R'$
$(S\ /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))\ (fst\ gh1)) \leq deg\ R\ S\ X\ f) \wedge$
$P\text{-}mod\ R\ S\ X\ (S \diamondsuit_p\ (t^{\frown S}\ m))\ ((snd\ gh) \pm_R -_a R\ (snd\ gh1)) \wedge$
$P\text{-}mod\ R\ S\ X\ (S \diamondsuit_p\ (t^{\frown S}\ (Suc\ m)))\ (f \pm_R (-_a R\ ((fst\ gh1) \cdot_r R\ (snd\ gh1)))))$

lemma *cart-prod-fst*:$x \in A \times B \Longrightarrow fst\ x \in A$
⟨*proof*⟩

lemma *cart-prod-snd*:$x \in A \times B \Longrightarrow snd\ x \in B$
⟨*proof*⟩

lemma *cart-prod-split*:$((x,y) \in A \times B) = (x \in A \wedge y \in B)$
⟨*proof*⟩

lemma (**in** *PolynRg*) *P-mod-diffxxx3*:⟦*Idomain S*; $t \in carrier\ S$; $t \neq \mathbf{0}_S$;
maximal-ideal S $(S \diamondsuit_p\ t)$; *PolynRg* $R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y$;
$f \in carrier\ R$; $gh \in carrier\ R \times carrier\ R$;
$deg\ R\ S\ X\ (fst\ gh) \leq deg\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y\ (erH\ R\ S\ X\ R'$
$(S\ /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))\ (fst\ gh))$;
$deg\ R\ S\ X\ (snd\ gh) + deg\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y\ (erH\ R\ S\ X\ R'$
$(S\ /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))\ (fst\ gh)) \leq deg\ R\ S\ X\ f$;
$0 < deg\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y\ (erH\ R\ S\ X\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y$
$(pj\ S\ (S \diamondsuit_p\ t))\ (fst\ gh))$;
$0 < deg\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y$
$(erH\ R\ S\ X\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))\ (snd\ gh))$;
rel-prime-pols $R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y$
$(erH\ R\ S\ X\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))\ (fst\ gh))$
$(erH\ R\ S\ X\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))\ (snd\ gh))$;
$P\text{-}mod\ R\ S\ X\ (S \diamondsuit_p\ (t^{\frown S}\ m))\ (f \pm -_a\ ((fst\ gh) \cdot_r\ (snd\ gh)))$; $0 < m$⟧ ⟹
$\exists\, gh1.\ gh1 \in carrier\ R \times carrier\ R \wedge$
$(deg\ R\ S\ X\ (fst\ gh1) \leq deg\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y$
$(erH\ R\ S\ X\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))\ (fst\ gh1))) \wedge$
$P\text{-}mod\ R\ S\ X\ (S \diamondsuit_p\ (t^{\frown S}\ m))\ ((fst\ gh) \pm -_a\ (fst\ gh1)) \wedge$
$(deg\ R\ S\ X\ (snd\ gh1) + deg\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y$
$(erH\ R\ S\ X\ R'\ (S\ /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))\ (fst\ gh1)) \leq$

$$deg\ R\ S\ X\ f) \wedge$$
$$P\text{-}mod\ R\ S\ X\ (S \diamondsuit_p\ (t^{\frown S}\ m))\ ((snd\ gh) \pm -_a\ (snd\ gh1)) \wedge$$
$$P\text{-}mod\ R\ S\ X\ (S \diamondsuit_p\ (t^{\frown S}\ (Suc\ m)))\ (f \pm (-_a\ ((fst\ gh1) \cdot_r\ (snd\ gh1))))$$
⟨*proof*⟩

lemma (**in** *PolynRg*) *P-mod-diffxxx4*:⟦*Idomain S*; $t \in carrier\ S$; $t \neq \mathbf{0}_S$;
maximal-ideal S $(S \diamondsuit_p\ t)$; *PolynRg* $R'\ (S /_r\ (S \diamondsuit_p\ t))\ Y$; $f \in carrier\ R$;
$gh \in carrier\ R \times carrier\ R$;
$deg\ R\ S\ X\ (fst\ gh) \leq deg\ R'\ (S /_r\ (S \diamondsuit_p\ t))\ Y$
$(erH\ R\ S\ X\ R'\ (S /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))\ (fst\ gh))$;
$deg\ R\ S\ X\ (snd\ gh) + deg\ R'\ (S /_r\ (S \diamondsuit_p\ t))\ Y\ (erH\ R\ S\ X\ R'$
$(S /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))\ (fst\ gh)) \leq deg\ R\ S\ X\ f$;
$0 < deg\ R'\ (S /_r\ (S \diamondsuit_p\ t))\ Y$
$(erH\ R\ S\ X\ R'\ (S /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))\ (fst\ gh))$;
$0 < deg\ R'\ (S /_r\ (S \diamondsuit_p\ t))\ Y$
$(erH\ R\ S\ X\ R'\ (S /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))\ (snd\ gh))$;
rel-prime-pols $R'\ (S /_r\ (S \diamondsuit_p\ t))\ Y$
$(erH\ R\ S\ X\ R'\ (S /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))\ (fst\ gh))$
$(erH\ R\ S\ X\ R'\ (S /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))\ (snd\ gh))$;
P-mod $R\ S\ X\ (S \diamondsuit_p\ (t^{\frown S}\ m))\ (f \pm -_a\ ((fst\ gh) \cdot_r\ (snd\ gh)))$; $0 < m$⟧ ⟹
$(Hen_{R\ S\ X\ t\ R'\ Y\ f}\ m\ gh) \in carrier\ R \times carrier\ R\ \wedge (deg\ R\ S\ X$
$(fst\ (Hen_{R\ S\ X\ t\ R'\ Y\ f}\ m\ gh)) \leq deg\ R'\ (S /_r\ (S \diamondsuit_p\ t))\ Y$
$(erH\ R\ S\ X\ R'\ (S /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))$
$(fst\ (Hen_{R\ S\ X\ t\ R'\ Y\ f}\ m\ gh)))) \wedge$
$P\text{-}mod\ R\ S\ X\ (S \diamondsuit_p\ (t^{\frown S}\ m))\ ((fst\ gh) \pm -_a\ (fst\ (Hen_{R\ S\ X\ t\ R'\ Y\ f}\ m\ gh))) \wedge$
$(deg\ R\ S\ X\ (snd\ (Hen_{R\ S\ X\ t\ R'\ Y\ f}\ m\ gh)) + deg\ R'\ (S /_r\ (S \diamondsuit_p\ t))\ Y$
$(erH\ R\ S\ X\ R'\ (S /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))$
$(fst\ (Hen_{R\ S\ X\ t\ R'\ Y\ f}\ m\ gh))) \leq\ deg\ R\ S\ X\ f) \wedge$
$P\text{-}mod\ R\ S\ X\ (S \diamondsuit_p\ (t^{\frown S}\ m))\ ((snd\ gh) \pm -_a\ (snd\ (Hen_{R\ S\ X\ t\ R'\ Y\ f}\ m\ gh)))$
$\wedge$
$P\text{-}mod\ R\ S\ X\ (S \diamondsuit_p\ (t^{\frown S}\ (Suc\ m)))\ (f \pm (-_a\ ((fst\ (Hen_{R\ S\ X\ t\ R'\ Y\ f}\ m\ gh))$
$\cdot_r$
$(snd\ (Hen_{R\ S\ X\ t\ R'\ Y\ f}\ m\ gh)))))$
⟨*proof*⟩

primrec
Hensel-pair :: [('*a*, '*b*) *Ring-scheme*, ('*a*, '*c*) *Ring-scheme*, '*a*, '*a*,
('*a set*, '*m*) *Ring-scheme*, '*a set*, '*a*, '*a*, '*a*, *nat*] ⇒ ('*a* × '*a*)
(‹(*10Hpr* - - - - - - - - - -)› [67,67,67,67,67,67,67,67,67,68]67)
where
Hpr-0: $Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ 0 = (g, h)$
| *Hpr-Suc*: $Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ (Suc\ m) =$
$Hen_{R\ S\ X\ t\ R'\ Y\ f}\ (Suc\ m)\ (Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ m)$

lemma (**in** *PolynRg*) *fst-xxx*: ⟦$t \in carrier\ S$; $t \neq \mathbf{0}_S$; *ideal* $S\ (S \diamondsuit_p\ t)$;

$\forall (n{::}nat).\ (F\ n) \in carrier\ R \times carrier\ R;$
$\forall m.\ P\text{-}mod\ R\ S\ X\ (S \diamondsuit_p\ t)\ (fst\ (F\ m) \pm -_a\ (fst\ (F\ (Suc\ m))))\rrbracket \Longrightarrow$
$P\text{-}mod\ R\ S\ X\ (S \diamondsuit_p\ t)\ (fst\ (F\ 0) \pm -_a\ (fst\ (F\ n)))$
⟨*proof*⟩

lemma (**in** *PolynRg*) *snd-xxx*:⟦$t \in carrier\ S;\ t \neq \mathbf{0}_S;$
$ideal\ S\ (S \diamondsuit_p\ t);\ \forall (n{::}nat).\ (F\ n) \in carrier\ R \times carrier\ R;$
$\forall m.\ P\text{-}mod\ R\ S\ X\ (S \diamondsuit_p\ t)\ (snd\ (F\ m) \pm -_a\ (snd\ (F\ (Suc\ m))))\rrbracket \Longrightarrow$
$P\text{-}mod\ R\ S\ X\ (S \diamondsuit_p\ t)\ (snd\ (F\ 0) \pm -_a\ (snd\ (F\ n)))$
⟨*proof*⟩

lemma (**in** *PolynRg*) *P-mod-diffxxx5*:⟦$Idomain\ S;\ t \in carrier\ S;\ t \neq \mathbf{0}_S;$
$maximal\text{-}ideal\ S\ (S \diamondsuit_p\ t);\ PolynRg\ R'\ (S /_r\ (S \diamondsuit_p\ t))\ Y;$
$f \in carrier\ R;\ (g,\ h) \in carrier\ R \times carrier\ R;$
$deg\ R\ S\ X\ (fst\ (g,\ h)) \leq deg\ R'\ (S /_r\ (S \diamondsuit_p\ t))\ Y$
$(erH\ R\ S\ X\ R'\ (S /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))\ (fst\ (g,\ h)));$
$deg\ R\ S\ X\ (snd\ (g,\ h)) + deg\ R'\ (S /_r\ (S \diamondsuit_p\ t))\ Y$
$(erH\ R\ S\ X\ R'\ (S /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))\ (fst\ (g,\ h))) \leq deg\ R\ S\ X\ f;$
$0 < deg\ R'\ (S /_r\ (S \diamondsuit_p\ t))\ Y\ (erH\ R\ S\ X\ R'\ (S /_r\ (S \diamondsuit_p\ t))\ Y$
$(pj\ S\ (S \diamondsuit_p\ t))\ (fst\ (g,\ h)));$
$0 < deg\ R'\ (S /_r\ (S \diamondsuit_p\ t))\ Y\ (erH\ R\ S\ X\ R'\ (S /_r\ (S \diamondsuit_p\ t))\ Y$
$(pj\ S\ (S \diamondsuit_p\ t))\ (snd\ (g,\ h)));$
$rel\text{-}prime\text{-}pols\ R'\ (S /_r\ (S \diamondsuit_p\ t))\ Y$
$(erH\ R\ S\ X\ R'\ (S /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))\ (fst\ (g,\ h)))$
$(erH\ R\ S\ X\ R'\ (S /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))\ (snd\ (g,\ h)));$
$P\text{-}mod\ R\ S\ X\ (S \diamondsuit_p\ t)\ (f \pm -_a\ (g \cdot_r\ h))\rrbracket \Longrightarrow$
$(Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ (Suc\ m)) \in carrier\ R \times carrier\ R\ \wedge$
$erH\ R\ S\ X\ R'\ (S /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))$
$(fst\ (Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ (Suc\ m))) =$
$erH\ R\ S\ X\ R'\ (S /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))\ (fst\ (g,\ h)) \wedge$
$erH\ R\ S\ X\ R'\ (S /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))$
$(snd\ (Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ (Suc\ m))) =$
$erH\ R\ S\ X\ R'\ (S /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))\ (snd\ (g,\ h)) \wedge$
$(deg\ R\ S\ X\ (fst\ (Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ (Suc\ m))) \leq deg\ R'\ (S /_r\ (S \diamondsuit_p\ t))\ Y$
$(erH\ R\ S\ X\ R'\ (S /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))$
$(fst\ (Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ (Suc\ m)))) \wedge$
$P\text{-}mod\ R\ S\ X\ (S \diamondsuit_p\ (t^{\frown S}\ (Suc\ m)))\ ((fst\ (Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ m)) \pm -_a$
$(fst\ (Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ (Suc\ m)))) \wedge$
$(deg\ R\ S\ X\ (snd\ (Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ (Suc\ m))) + deg\ R'\ (S /_r\ (S \diamondsuit_p\ t))\ Y$
$(erH\ R\ S\ X\ R'\ (S /_r\ (S \diamondsuit_p\ t))\ Y\ (pj\ S\ (S \diamondsuit_p\ t))\ (fst\ (Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}$
$(Suc\ m)))) \leq\ deg\ R\ S\ X\ f) \wedge$
$P\text{-}mod\ R\ S\ X\ (S \diamondsuit_p\ (t^{\frown S}\ (Suc\ m)))\ ((snd\ (Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ m)) \pm -_a\ (snd$
$(Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ (Suc\ m)))) \wedge$
$P\text{-}mod\ R\ S\ X\ (S \diamondsuit_p\ (t^{\frown S}\ (Suc\ (Suc\ m))))\ (f \pm -_a\ ((fst\ (Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}$
$(Suc\ m))) \cdot_r\ (snd\ (Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ (Suc\ m)))))$
⟨*proof*⟩

lemma (**in** *PolynRg*) *P-mod-diffxxx5-1*:⟦*Idomain S*; $t \in$ *carrier S*; $t \neq \mathbf{0}_S$;
maximal-ideal S (*S* $\diamondsuit_p$ *t*); *PolynRg R'* (*S* $/_r$ (*S* $\diamondsuit_p$ *t*)) *Y*;
$f \in$ *carrier R*; $g \in$ *carrier R*; $h \in$ *carrier R*;
deg R S X g $\leq$ *deg R'* (*S* $/_r$ (*S* $\diamondsuit_p$ *t*)) *Y*
(*erH R S X R'* (*S* $/_r$ (*S* $\diamondsuit_p$ *t*)) *Y* (*pj S* (*S* $\diamondsuit_p$ *t*)) *g*);
deg R S X h + *deg R'* (*S* $/_r$ (*S* $\diamondsuit_p$ *t*)) *Y* (*erH R S X R'*
(*S* $/_r$ (*S* $\diamondsuit_p$ *t*)) *Y* (*pj S* (*S* $\diamondsuit_p$ *t*)) *g*) $\leq$ *deg R S X f*;
0 $<$ *deg R'* (*S* $/_r$ (*S* $\diamondsuit_p$ *t*)) *Y*
(*erH R S X R'* (*S* $/_r$ (*S* $\diamondsuit_p$ *t*)) *Y* (*pj S* (*S* $\diamondsuit_p$ *t*)) *g*);
0 $<$ *deg R'* (*S* $/_r$ (*S* $\diamondsuit_p$ *t*)) *Y*
(*erH R S X R'* (*S* $/_r$ (*S* $\diamondsuit_p$ *t*)) *Y* (*pj S* (*S* $\diamondsuit_p$ *t*)) *h*);
rel-prime-pols R' (*S* $/_r$ (*S* $\diamondsuit_p$ *t*)) *Y*
(*erH R S X R'* (*S* $/_r$ (*S* $\diamondsuit_p$ *t*)) *Y* (*pj S* (*S* $\diamondsuit_p$ *t*)) *g*)
(*erH R S X R'* (*S* $/_r$ (*S* $\diamondsuit_p$ *t*)) *Y* (*pj S* (*S* $\diamondsuit_p$ *t*)) *h*);
P-mod R S X (*S* $\diamondsuit_p$ *t*) (*f* $\pm$ $-_a$ (*g* $\cdot_r$ *h*))⟧ $\Longrightarrow$
(*Hpr*$_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}$ (*Suc m*)) $\in$ *carrier R* $\times$ *carrier R* $\wedge$
erH R S X R' (*S* $/_r$ (*S* $\diamondsuit_p$ *t*)) *Y* (*pj S* (*S* $\diamondsuit_p$ *t*))
(*fst* (*Hpr*$_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}$ (*Suc m*))) =
erH R S X R' (*S* $/_r$ (*S* $\diamondsuit_p$ *t*)) *Y* (*pj S* (*S* $\diamondsuit_p$ *t*)) (*fst* (*g*, *h*)) $\wedge$
erH R S X R' (*S* $/_r$ (*S* $\diamondsuit_p$ *t*)) *Y* (*pj S* (*S* $\diamondsuit_p$ *t*))
(*snd* (*Hpr*$_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}$ (*Suc m*))) =
erH R S X R' (*S* $/_r$ (*S* $\diamondsuit_p$ *t*)) *Y* (*pj S* (*S* $\diamondsuit_p$ *t*)) (*snd* (*g*, *h*)) $\wedge$
(*deg R S X* (*fst* (*Hpr*$_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}$ (*Suc m*))) $\leq$ *deg R'*
(*S* $/_r$ (*S* $\diamondsuit_p$ *t*)) *Y* (*erH R S X R'* (*S* $/_r$ (*S* $\diamondsuit_p$ *t*)) *Y*
(*pj S* (*S* $\diamondsuit_p$ *t*)) (*fst* (*Hpr*$_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}$ (*Suc m*))))) $\wedge$
P-mod R S X (*S* $\diamondsuit_p$ ($t^{\frown S}$ (*Suc m*))) ((*fst* (*Hpr*$_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}$ *m*)) $\pm$ $-_a$
(*fst* (*Hpr*$_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}$ (*Suc m*)))) $\wedge$
(*deg R S X* (*snd* (*Hpr*$_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}$ (*Suc m*))) +
deg R' (*S* $/_r$ (*S* $\diamondsuit_p$ *t*)) *Y* (*erH R S X R'* (*S* $/_r$ (*S* $\diamondsuit_p$ *t*)) *Y*
(*pj S* (*S* $\diamondsuit_p$ *t*)) (*fst* (*Hpr*$_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}$ (*Suc m*)))) $\leq$ *deg R S X f*) $\wedge$
P-mod R S X (*S* $\diamondsuit_p$ ($t^{\frown S}$ (*Suc m*))) ((*snd* (*Hpr*$_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}$ *m*)) $\pm$ $-_a$
(*snd* (*Hpr*$_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}$ (*Suc m*)))) $\wedge$
P-mod R S X (*S* $\diamondsuit_p$ ($t^{\frown S}$ (*Suc* (*Suc m*)))) (*f* $\pm$ $-_a$
((*fst* (*Hpr*$_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}$ (*Suc m*))) $\cdot_r$ (*snd* (*Hpr*$_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}$ (*Suc m*)))))
⟨*proof*⟩

lemma (**in** *PolynRg*) *P-mod-diffxxx5-2*:⟦*Idomain S*; $t \in$ *carrier S*; $t \neq \mathbf{0}_S$;
maximal-ideal S (*S* $\diamondsuit_p$ *t*); *PolynRg R'* (*S* $/_r$ (*S* $\diamondsuit_p$ *t*)) *Y*; $f \in$ *carrier R*;
$g \in$ *carrier R*; $h \in$ *carrier R*;
deg R S X g $\leq$ *deg R'* (*S* $/_r$ (*S* $\diamondsuit_p$ *t*)) *Y*
(*erH R S X R'* (*S* $/_r$ (*S* $\diamondsuit_p$ *t*)) *Y* (*pj S* (*S* $\diamondsuit_p$ *t*)) *g*);
deg R S X h + *deg R'* (*S* $/_r$ (*S* $\diamondsuit_p$ *t*)) *Y* (*erH R S X R'*
(*S* $/_r$ (*S* $\diamondsuit_p$ *t*)) *Y* (*pj S* (*S* $\diamondsuit_p$ *t*)) *g*) $\leq$ *deg R S X f*;
0 $<$ *deg R'* (*S* $/_r$ (*S* $\diamondsuit_p$ *t*)) *Y*

$(erH\ R\ S\ X\ R'\ (S\ /_r\ (S\ \diamondsuit_p\ t))\ Y\ (pj\ S\ (S\ \diamondsuit_p\ t))\ g);$
$0 < deg\ R'\ (S\ /_r\ (S\ \diamondsuit_p\ t))\ Y$
$(erH\ R\ S\ X\ R'\ (S\ /_r\ (S\ \diamondsuit_p\ t))\ Y\ (pj\ S\ (S\ \diamondsuit_p\ t))\ h);$
$rel\text{-}prime\text{-}pols\ R'\ (S\ /_r\ (S\ \diamondsuit_p\ t))\ Y$
$(erH\ R\ S\ X\ R'\ (S\ /_r\ (S\ \diamondsuit_p\ t))\ Y\ (pj\ S\ (S\ \diamondsuit_p\ t))\ g)$
$(erH\ R\ S\ X\ R'\ (S\ /_r\ (S\ \diamondsuit_p\ t))\ Y\ (pj\ S\ (S\ \diamondsuit_p\ t))\ h);$
$P\text{-}mod\ R\ S\ X\ (S\ \diamondsuit_p\ t)\ (f \pm -_a\ (g\ \cdot_r\ h))\rrbracket \Longrightarrow$
$(Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ m) \in carrier\ R \times carrier\ R$

⟨*proof*⟩

lemma (**in** *PolynRg*) *P-mod-diffxxx5-3*:⟦*Idomain S*; $t \in carrier\ S$; $t \neq \mathbf{0}_S$;
$maximal\text{-}ideal\ S\ (S\ \diamondsuit_p\ t)$; $PolynRg\ R'\ (S\ /_r\ (S\ \diamondsuit_p\ t))\ Y$; $f \in carrier\ R$;
$g \in carrier\ R$; $h \in carrier\ R$;
$deg\ R\ S\ X\ g \leq deg\ R'\ (S\ /_r\ (S\ \diamondsuit_p\ t))\ Y$
$(erH\ R\ S\ X\ R'\ (S\ /_r\ (S\ \diamondsuit_p\ t))\ Y\ (pj\ S\ (S\ \diamondsuit_p\ t))\ g);$
$deg\ R\ S\ X\ h + deg\ R'\ (S\ /_r\ (S\ \diamondsuit_p\ t))\ Y\ (erH\ R\ S\ X\ R'$
$(S\ /_r\ (S\ \diamondsuit_p\ t))\ Y\ (pj\ S\ (S\ \diamondsuit_p\ t))\ g) \leq deg\ R\ S\ X\ f;$
$0 < deg\ R'\ (S\ /_r\ (S\ \diamondsuit_p\ t))\ Y$
$(erH\ R\ S\ X\ R'\ (S\ /_r\ (S\ \diamondsuit_p\ t))\ Y\ (pj\ S\ (S\ \diamondsuit_p\ t))\ g);$
$0 < deg\ R'\ (S\ /_r\ (S\ \diamondsuit_p\ t))\ Y$
$(erH\ R\ S\ X\ R'\ (S\ /_r\ (S\ \diamondsuit_p\ t))\ Y\ (pj\ S\ (S\ \diamondsuit_p\ t))\ h);$
$rel\text{-}prime\text{-}pols\ R'\ (S\ /_r\ (S\ \diamondsuit_p\ t))\ Y$
$(erH\ R\ S\ X\ R'\ (S\ /_r\ (S\ \diamondsuit_p\ t))\ Y\ (pj\ S\ (S\ \diamondsuit_p\ t))\ g)$
$(erH\ R\ S\ X\ R'\ (S\ /_r\ (S\ \diamondsuit_p\ t))\ Y\ (pj\ S\ (S\ \diamondsuit_p\ t))\ h);$
$P\text{-}mod\ R\ S\ X\ (S\ \diamondsuit_p\ t)\ (f \pm -_a\ (g\ \cdot_r\ h))\rrbracket \Longrightarrow$
$P\text{-}mod\ R\ S\ X\ (S\ \diamondsuit_p\ (t^{\frown S}\ m))\ ((fst\ (Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ m)) \pm$
$-_a\ (fst\ (Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ (m + n)))) \wedge$
$P\text{-}mod\ R\ S\ X\ (S\ \diamondsuit_p\ (t^{\frown S}\ m))\ ((snd\ (Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ m)) \pm$
$-_a\ (snd\ (Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ (m + n))))$

⟨*proof*⟩

lemma (**in** *PolynRg*) *P-mod-diffxxx5-4*:⟦*Idomain S*; $t \in carrier\ S$; $t \neq \mathbf{0}_S$;
$maximal\text{-}ideal\ S\ (S\ \diamondsuit_p\ t)$; $PolynRg\ R'\ (S\ /_r\ (S\ \diamondsuit_p\ t))\ Y$; $f \in carrier\ R$;
$g \in carrier\ R$; $h \in carrier\ R$;
$deg\ R\ S\ X\ g \leq deg\ R'\ (S\ /_r\ (S\ \diamondsuit_p\ t))\ Y$
$(erH\ R\ S\ X\ R'\ (S\ /_r\ (S\ \diamondsuit_p\ t))\ Y\ (pj\ S\ (S\ \diamondsuit_p\ t))\ g);$
$deg\ R\ S\ X\ h + deg\ R'\ (S\ /_r\ (S\ \diamondsuit_p\ t))\ Y\ (erH\ R\ S\ X\ R'$
$(S\ /_r\ (S\ \diamondsuit_p\ t))\ Y\ (pj\ S\ (S\ \diamondsuit_p\ t))\ g) \leq deg\ R\ S\ X\ f;$
$0 < deg\ R'\ (S\ /_r\ (S\ \diamondsuit_p\ t))\ Y$
$(erH\ R\ S\ X\ R'\ (S\ /_r\ (S\ \diamondsuit_p\ t))\ Y\ (pj\ S\ (S\ \diamondsuit_p\ t))\ g);$
$0 < deg\ R'\ (S\ /_r\ (S\ \diamondsuit_p\ t))\ Y$
$(erH\ R\ S\ X\ R'\ (S\ /_r\ (S\ \diamondsuit_p\ t))\ Y\ (pj\ S\ (S\ \diamondsuit_p\ t))\ h);$
$rel\text{-}prime\text{-}pols\ R'\ (S\ /_r\ (S\ \diamondsuit_p\ t))\ Y$
$(erH\ R\ S\ X\ R'\ (S\ /_r\ (S\ \diamondsuit_p\ t))\ Y\ (pj\ S\ (S\ \diamondsuit_p\ t))\ g)$
$(erH\ R\ S\ X\ R'\ (S\ /_r\ (S\ \diamondsuit_p\ t))\ Y\ (pj\ S\ (S\ \diamondsuit_p\ t))\ h);$
$P\text{-}mod\ R\ S\ X\ (S\ \diamondsuit_p\ t)\ (f \pm -_a\ (g\ \cdot_r\ h))\rrbracket \Longrightarrow$

$deg\ R\ S\ X\ (fst\ (Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ m)) \leq deg\ R\ S\ X\ g \wedge$
$deg\ R\ S\ X\ (snd\ (Hpr_{R\ S\ X\ t\ R'\ Y\ f\ g\ h}\ m)) \leq deg\ R\ S\ X\ f$
⟨*proof*⟩

end

theory *Algebra7* **imports** *Algebra6* **begin**

Chapter 5

Modules

5.1 Basic properties of Modules

record $('a, 'b)$ *Module* $= 'a$ *aGroup* $+$
  sprod $:: 'b \Rightarrow 'a \Rightarrow 'a$ (**infixl** $\langle\cdot_{s1}\rangle$ *76*)

locale *Module* $=$ *aGroup* M **for** M (**structure**) $+$
  fixes R (**structure**)
  assumes *sc-Ring*: *Ring* R
  and *sprod-closed* :
    $[\![\ a \in carrier\ R;\ m \in carrier\ M]\!] \Longrightarrow a \cdot_s m \in carrier\ M$
   and *sprod-l-distr*:
    $[\![a \in carrier\ R;\ b \in carrier\ R;\ m \in carrier\ M]\!] \Longrightarrow$
     $(a \pm_R b) \cdot_s m = a \cdot_s m \pm_M b \cdot_s m$
   and *sprod-r-distr*:
    $[\![\ a \in carrier\ R;\ m \in carrier\ M;\ n \in carrier\ M\ ]\!] \Longrightarrow$
    $a \cdot_s (m \pm_M n) = a \cdot_s m \pm_M a \cdot_s n$
   and *sprod-assoc*:
    $[\![\ a \in carrier\ R;\ b \in carrier\ R;\ m \in carrier\ M\ ]\!] \Longrightarrow$
    $(a \cdot_{r\,R} b) \cdot_s m = a \cdot_s (b \cdot_s m)$
   and *sprod-one*:
    $m \in carrier\ M \Longrightarrow (1_{r\,R}) \cdot_s m = m$

definition
  submodule $::$ $[('b, 'm)$ *Ring-scheme*, $('a, 'b, 'c)$ *Module-scheme*, $'a$ *set*$] \Rightarrow$
      bool **where**
  submodule $R\ A\ H \longleftrightarrow H \subseteq carrier\ A \wedge A +> H \wedge (\forall a.\ \forall m.$
      $(a \in carrier\ R \wedge m \in H) \longrightarrow (sprod\ A\ a\ m) \in H)$

definition
  mdl $::$ $[('a, 'b, 'm)$ *Module-scheme*, $'a$ *set*$] \Rightarrow ('a, 'b)$ *Module* **where**
  mdl $M\ H = (\!|carrier = H,\ pop = pop\ M,\ mop = mop\ M,\ zero = zero\ M,$
    $sprod = \lambda a.\ \lambda x{\in}H.\ sprod\ M\ a\ x|\!)$

abbreviation

MODULE (**infixl** ‹*module*› *58*) **where**
R module M == *Module M R*

lemma (**in** *Module*) *module-is-ag*: *aGroup M* ⟨*proof*⟩

lemma (**in** *Module*) *module-inc-zero*: $\mathbf{0}_M \in$ *carrier M*
⟨*proof*⟩

lemma (**in** *Module*) *submodule-subset*:*submodule R M H* $\Longrightarrow$ *H* $\subseteq$ *carrier M*
⟨*proof*⟩

lemma (**in** *Module*) *submodule-asubg*:*submodule R M H* $\Longrightarrow$ *M* +> *H*
⟨*proof*⟩

lemma (**in** *Module*) *submodule-subset1*:⟦*submodule R M H*; *h* $\in$ *H*⟧ $\Longrightarrow$
h $\in$ *carrier M*
⟨*proof*⟩

lemma (**in** *Module*) *submodule-inc-0*:*submodule R M H* $\Longrightarrow$
$\mathbf{0}_M \in H$
⟨*proof*⟩

lemma (**in** *Module*) *sc-un*: *m* $\in$ *carrier M* $\Longrightarrow$ $1_{r\,R}$ $\cdot_s$ *m* = *m*
⟨*proof*⟩

lemma (**in** *Module*) *sc-mem*:⟦*a* $\in$ *carrier R*; *m* $\in$ *carrier M*⟧ $\Longrightarrow$
a $\cdot_s$ *m* $\in$ *carrier M*
⟨*proof*⟩

lemma (**in** *Module*) *submodule-sc-closed*:⟦*submodule R M H*;
a $\in$ *carrier R*; *h* $\in$ *H*⟧ $\Longrightarrow$ *a* $\cdot_s$ *h* $\in$ *H*
⟨*proof*⟩

lemma (**in** *Module*) *sc-assoc*:⟦*a* $\in$ *carrier R*; *b* $\in$ *carrier R*;
m $\in$ *carrier M*⟧ $\Longrightarrow$ (*a* $\cdot_{r\,R}$ *b*) $\cdot_s$ *m* = *a* $\cdot_s$ (*b* $\cdot_s$ *m*)
⟨*proof*⟩

lemma (**in** *Module*) *sc-l-distr*:⟦*a* $\in$ *carrier R*; *b* $\in$ *carrier R*;
m $\in$ *carrier M*⟧ $\Longrightarrow$ (*a* $\pm_R$ *b*)$\cdot_s$ *m* = *a* $\cdot_s$ *m* $\pm$ *b* $\cdot_s$ *m*
⟨*proof*⟩

lemma (**in** *Module*) *sc-r-distr*:⟦*a* $\in$ *carrier R*; *m* $\in$ *carrier M*; *n* $\in$ *carrier M*⟧ $\Longrightarrow$
a $\cdot_s$ (*m* $\pm$ *n*) = *a* $\cdot_s$ *m* $\pm$ *a* $\cdot_s$ *n*
⟨*proof*⟩

lemma (**in** *Module*) *sc-0-m*:*m* $\in$ *carrier M* $\Longrightarrow$ $\mathbf{0}_R{}_{\cdot s}$ *m* = $\mathbf{0}_M$
⟨*proof*⟩

lemma (**in** *Module*) *sc-a-0*:$a \in$ *carrier* $R \Longrightarrow a \cdot_s \mathbf{0} = \mathbf{0}$
⟨*proof*⟩

lemma (**in** *Module*) *sc-minus-am*:$[\![a \in$ *carrier* $R;\ m \in$ *carrier* $M]\!]$
$\Longrightarrow -_a\ (a \cdot_s m) = a \cdot_s (-_a\ m)$
⟨*proof*⟩

lemma (**in** *Module*) *sc-minus-am1*:$[\![a \in$ *carrier* $R;\ m \in$ *carrier* $M]\!]$
$\Longrightarrow -_a\ (a \cdot_s m) = (-_{a\,R}\ a) \cdot_s m$
⟨*proof*⟩

lemma (**in** *Module*) *submodule-0*:*submodule* $R\ M\ \{\mathbf{0}\}$
⟨*proof*⟩

lemma (**in** *Module*) *submodule-whole*:*submodule* $R\ M$ (*carrier* M)
⟨*proof*⟩

lemma (**in** *Module*) *submodule-pOp-closed*:$[\![$*submodule* $R\ M\ H;\ h \in H;\ k \in H]\!]$
$\Longrightarrow$
$h \pm k \in H$
⟨*proof*⟩

lemma (**in** *Module*) *submodule-mOp-closed*:$[\![$*submodule* $R\ M\ H;\ h \in H]\!]$
$\Longrightarrow -_a\ h \in H$
⟨*proof*⟩

definition
mHom :: [('*b*, '*m*) *Ring-scheme*, ('*a*, '*b*, '*m1*) *Module-scheme*,
('*c*, '*b*, '*m2*) *Module-scheme*] ⇒ ('*a* ⇒ '*c*) *set*
where
mHom $R\ M\ N = \{f.\ f \in aHom\ M\ N \wedge$
$(\forall\, a{\in}carrier\ R.\ \forall\, m{\in}carrier\ M.\ f\ (a \cdot_{s\,M} m) = a \cdot_{s\,N} (f\ m))\}$

definition
mimg :: [('*b*, '*m*) *Ring-scheme*, ('*a*, '*b*, '*m1*) *Module-scheme*,
('*c*, '*b*, '*m2*) *Module-scheme*, '*a* ⇒ '*c*] ⇒ ('*c*, '*b*) *Module*
(‹(*4mimg*_ _,_/ _)› [*88*,*88*,*88*,*89*]*88*) **where**
$mimg_{R\ M,N}\ f = mdl\ N\ (f\ `\ (carrier\ M))$

definition
mzeromap :: [('*a*, '*b*, '*m1*) *Module-scheme*, ('*c*, '*b*, '*m2*) *Module-scheme*]
⇒ ('*a* ⇒ '*c*) **where**
mzeromap $M\ N = (\lambda x{\in}carrier\ M.\ \mathbf{0}_N)$

lemma (**in** *Ring*) *mHom-func*:$f \in mHom\ R\ M\ N \Longrightarrow f \in carrier\ M \to carrier\ N$
⟨*proof*⟩

lemma (**in** *Module*) *mHom-test*:$[\![R$ *module* $N;\ f \in carrier\ M \to carrier\ N \wedge$
$f \in extensional\ (carrier\ M) \wedge$

$(\forall\, m{\in}carrier\ M.\ \forall\, n{\in}carrier\ M.\ f\ (m \pm_M n) = f\ m \pm_N (f\ n)) \wedge$
$(\forall\, a{\in}carrier\ R.\ \forall\, m{\in}carrier\ M.\ f\ (a \cdot_{s\,M} m) = a \cdot_{s\,N} (f\ m))\rrbracket \Longrightarrow$
$f \in mHom\ R\ M\ N$
⟨*proof*⟩

lemma (**in** *Module*) *mHom-mem*:⟦R *module* N; $f \in mHom\ R\ M\ N$; $m \in carrier\ M$⟧
$\Longrightarrow f\ m \in carrier\ N$
⟨*proof*⟩

lemma (**in** *Module*) *mHom-add*:⟦R *module* N; $f \in mHom\ R\ M\ N$; $m \in carrier\ M$;
$n \in carrier\ M\rrbracket \Longrightarrow f\ (m \pm n) = f\ m \pm_N (f\ n)$
⟨*proof*⟩

lemma (**in** *Module*) *mHom-0*:⟦R *module* N; $f \in mHom\ R\ M\ N\rrbracket \Longrightarrow f\ (\mathbf{0}) = \mathbf{0}_N$
⟨*proof*⟩

lemma (**in** *Module*) *mHom-inv*:⟦R *module* N; $m \in carrier\ M$; $f \in mHom\ R\ M\ N\rrbracket$
$\Longrightarrow$
$f\ (-_a\ m) = -_{a\,N}\ (f\ m)$
⟨*proof*⟩

lemma (**in** *Module*) *mHom-lin*:⟦R *module* N; $m \in carrier\ M$; $f \in mHom\ R\ M\ N$;
$a \in carrier\ R\rrbracket \Longrightarrow f\ (a \cdot_s m) = a \cdot_{s\,N} (f\ m)$
⟨*proof*⟩

lemma (**in** *Module*) *mker-inc-zero*:
⟦R *module* N; $f \in mHom\ R\ M\ N$ ⟧ $\Longrightarrow \mathbf{0} \in (ker_{M,N}\ f)$
⟨*proof*⟩

lemma (**in** *Module*) *mHom-eq-ker*:⟦R *module* N; $f \in mHom\ R\ M\ N$; $a \in carrier\ M$;
$b{\in}\ carrier\ M$; $a \pm (-_a\ b) \in ker_{M,N}\ f\rrbracket \Longrightarrow f\ a = f\ b$
⟨*proof*⟩

lemma (**in** *Module*) *mHom-ker-eq*:⟦R *module* N; $f \in mHom\ R\ M\ N$; $a \in carrier\ M$;
$b{\in}\ carrier\ M$; $f\ a = f\ b\rrbracket \Longrightarrow a \pm (-_a\ b) \in ker_{M,N}\ f$
⟨*proof*⟩

lemma (**in** *Module*) *mker-submodule*:⟦R *module* N; $f \in mHom\ R\ M\ N\rrbracket \Longrightarrow$
$submodule\ R\ M\ (ker_{M,N}\ f)$
⟨*proof*⟩

lemma (**in** *Module*) *mker-mzeromap*:R *module* $N \Longrightarrow$
$ker_{M,N}\ (mzeromap\ M\ N) = carrier\ M$
⟨*proof*⟩

lemma (**in** *Module*) *mdl-carrier*:*submodule R M H* $\Longrightarrow$ *carrier* (*mdl M H*) = *H*
⟨*proof*⟩

lemma (**in** *Module*) *mdl-is-ag*:*submodule R M H* $\Longrightarrow$ *aGroup* (*mdl M H*)
⟨*proof*⟩

lemma (**in** *Module*) *mdl-is-module*:*submodule R M H* $\Longrightarrow$ *R module* (*mdl M H*)
⟨*proof*⟩

lemma (**in** *Module*) *submodule-of-mdl*:⟦*submodule R M H*; *submodule R M N*; *H* ⊆ *N*⟧
$\Longrightarrow$ *submodule R* (*mdl M N*) *H*
⟨*proof*⟩

lemma (**in** *Module*) *img-set-submodule*:⟦*R module N*; *f* ∈ *mHom R M N*⟧ $\Longrightarrow$
submodule R N (*f* ‘ (*carrier M*))
⟨*proof*⟩

lemma (**in** *Module*) *mimg-module*:⟦*R module N*; *f* ∈ *mHom R M N*⟧ $\Longrightarrow$
R module (*mimg R M N f*)
⟨*proof*⟩

lemma (**in** *Module*) *surjec-to-mimg*:⟦*R module N*; *f* ∈ *mHom R M N*⟧ $\Longrightarrow$
$surjec_{M,\ (mimg\ R\ M\ N\ f)}\ f$
⟨*proof*⟩

definition
tOp-mHom :: [('*b*, '*m*) *Ring-scheme*, ('*a*, '*b*, '*m1*) *Module-scheme*,
('*c*, '*b*, '*m2*) *Module-scheme*] ⇒ ('*a* ⇒ '*c*) ⇒ ('*a* ⇒ '*c*) ⇒ ('*a* ⇒ '*c*) **where**
tOp-mHom R M N f g = (λ*x* ∈ *carrier M*. (*f x* $\pm_N$ (*g x*)))

definition
iOp-mHom :: [('*b*, '*m*) *Ring-scheme*, ('*a*, '*b*, '*m1*) *Module-scheme*,
('*c*, '*b*, '*m2*) *Module-scheme*] ⇒ ('*a* ⇒ '*c*) ⇒ ('*a* ⇒ '*c*) **where**
iOp-mHom R M N f = (λ*x* ∈ *carrier M*. ($-_a$ $_N$ (*f x*)))

definition
sprod-mHom ::[('*b*, '*m*) *Ring-scheme*, ('*a*, '*b*, '*m1*) *Module-scheme*,
('*c*, '*b*, '*m2*) *Module-scheme*] ⇒ '*b* ⇒ ('*a* ⇒ '*c*) ⇒ ('*a* ⇒ '*c*) **where**
sprod-mHom R M N a f = (λ*x* ∈ *carrier M*. *a* $\cdot_{s\,N}$ (*f x*))

definition
HOM :: [('*b*, '*more*) *Ring-scheme*, ('*a*, '*b*, '*more1*) *Module-scheme*,
('*c*, '*b*, '*more2*) *Module-scheme*] ⇒ ('*a* ⇒ '*c*, '*b*) *Module*
(‹(*3HOM*_ -/ -)› [*90*, *90*, *91*] *90*) **where**
HOM_R *M N* = ⦇*carrier* = *mHom R M N*, *pop* = *tOp-mHom R M N*,
mop = *iOp-mHom R M N*, *zero* = *mzeromap M N*, *sprod* =*sprod-mHom R M N* ⦈

lemma (**in** *Module*) *zero-HOM*:*R module N* $\Longrightarrow$
mzeromap M N = $\mathbf{0}_{HOM_R\ M\ N}$
⟨*proof*⟩

lemma (**in** *Module*) *tOp-mHom-closed*:⟦*R module N*; *f* ∈ *mHom R M N*; *g* ∈ *mHom R M N*⟧
$\Longrightarrow$ *tOp-mHom R M N f g* ∈ *mHom R M N*
⟨*proof*⟩

lemma (**in** *Module*) *iOp-mHom-closed*:⟦*R module N*; *f* ∈ *mHom R M N*⟧
$\Longrightarrow$ *iOp-mHom R M N f* ∈ *mHom R M N*
⟨*proof*⟩

lemma (**in** *Module*) *mHom-ex-zero*:*R module N* $\Longrightarrow$ *mzeromap M N* ∈ *mHom R M N*
⟨*proof*⟩

lemma (**in** *Module*) *mHom-eq*:⟦*R module N*; *f* ∈ *mHom R M N*; *g* ∈ *mHom R M N*;
∀ *m*∈*carrier M*. *f m* = *g m*⟧ $\Longrightarrow$ *f* = *g*
⟨*proof*⟩

lemma (**in** *Module*) *mHom-l-zero*:⟦*R module N*; *f* ∈ *mHom R M N*⟧
$\Longrightarrow$ *tOp-mHom R M N* (*mzeromap M N*) *f* = *f*
⟨*proof*⟩

lemma (**in** *Module*) *mHom-l-inv*:⟦*R module N*; *f* ∈ *mHom R M N*⟧
$\Longrightarrow$ *tOp-mHom R M N* (*iOp-mHom R M N f*) *f* = *mzeromap M N*
⟨*proof*⟩

lemma (**in** *Module*) *mHom-tOp-assoc*:⟦*R module N*; *f* ∈ *mHom R M N*; *g* ∈ *mHom R M N*;
h ∈ *mHom R M N*⟧ $\Longrightarrow$ *tOp-mHom R M N* (*tOp-mHom R M N f g*) *h* =
tOp-mHom R M N f (*tOp-mHom R M N g h*)
⟨*proof*⟩

lemma (**in** *Module*) *mHom-tOp-commute*:⟦*R module N*; *f* ∈ *mHom R M N*;
g ∈ *mHom R M N*⟧ $\Longrightarrow$ *tOp-mHom R M N f g* = *tOp-mHom R M N g f*
⟨*proof*⟩

lemma (**in** *Module*) *HOM-is-ag*:*R module N* $\Longrightarrow$ *aGroup* ($HOM_R\ M\ N$)
⟨*proof*⟩

lemma (**in** *Module*) *sprod-mHom-closed*:⟦*R module N*; *a* ∈ *carrier R*;
f ∈ *mHom R M N*⟧ $\Longrightarrow$ *sprod-mHom R M N a f* ∈ *mHom R M N*
⟨*proof*⟩

lemma (**in** *Module*) *HOM-is-module*:*R module N* $\Longrightarrow$ *R module* ($HOM_R\ M\ N$)
⟨*proof*⟩

5.2 Injective hom, surjective hom, bijective hom and inverse hom

definition
invmfun :: [('b, 'm) *Ring-scheme*, ('a, 'b, 'm1) *Module-scheme*,
('c, 'b, 'm2) *Module-scheme*, 'a ⇒ 'c] ⇒ 'c ⇒ 'a **where**
invmfun R M N (*f* :: 'a ⇒ 'c) =
(λy∈(*carrier N*). *SOME x*. (*x* ∈ (*carrier M*) ∧ *f x* = *y*))

definition
misomorphic :: [('b, 'm) *Ring-scheme*, ('a, 'b, 'm1) *Module-scheme*,
('c, 'b, 'm2) *Module-scheme*] ⇒ *bool* **where**
misomorphic R M N ⟷ (∃*f*. *f* ∈ *mHom R M N* ∧ *bijec*$_{M,N}$ *f*)

definition
mId :: ('a, 'b, 'm1) *Module-scheme* ⇒ 'a ⇒ 'a (‹(*mId*_/)› [89]88) **where**
mId$_M$ = (λm∈*carrier M*. *m*)

definition
mcompose :: [('a, 'r, 'm1) *Module-scheme*, 'b ⇒ 'c, 'a ⇒ 'b] ⇒ 'a ⇒ 'c **where**
mcompose M g f = *compose* (*carrier M*) *g f*

abbreviation
MISOM (‹(3- ≅_ -)› [82,82,83]82) **where**
M ≅$_R$ *N* == *misomorphic R M N*

lemma (**in** *Module*) *minjec-inj*:⟦*R module N*; *injec*$_{M,N}$ *f*⟧ ⟹
inj-on f (*carrier M*)
⟨*proof*⟩

lemma (**in** *Module*) *invmfun-l-inv*:⟦*R module N*; *bijec*$_{M,N}$ *f*; *m* ∈ *carrier M*⟧ ⟹
(*invmfun R M N f*) (*f m*) = *m*
⟨*proof*⟩

lemma (**in** *Module*) *invmfun-mHom*:⟦*R module N*; *bijec*$_{M,N}$ *f*; *f* ∈ *mHom R M N*
⟧ ⟹
invmfun R M N f ∈ *mHom R N M*
⟨*proof*⟩

lemma (**in** *Module*) *invmfun-r-inv*:⟦*R module N*; *bijec*$_{M,N}$ *f*; *n* ∈ *carrier N*⟧ ⟹
f ((*invmfun R M N f*) *n*) = *n*
⟨*proof*⟩

lemma (**in** *Module*) *mHom-compos*:⟦*R module L*; *R module N*; *f* ∈ *mHom R L M*;

g ∈ *mHom R M N* ⟧ ⟹ *compos L g f* ∈ *mHom R L N*
⟨*proof*⟩

lemma (**in** *Module*) *mcompos-inj-inj*:⟦*R module L*; *R module N*; *f* ∈ *mHom R L*

M;
$g \in mHom\ R\ M\ N$; $injec_{L,M}\ f$; $injec_{M,N}\ g\ \rrbracket \Longrightarrow injec_{L,N}\ (compos\ L\ g\ f)$
⟨*proof*⟩

lemma (**in** *Module*) *mcompos-surj-surj*:⟦R *module* L; R *module* N; $surjec_{L,M}\ f$;
$surjec_{M,N}\ g$; $f \in mHom\ R\ L\ M$; $g \in mHom\ R\ M\ N\ \rrbracket \Longrightarrow$
$surjec_{L,N}\ (compos\ L\ g\ f)$
⟨*proof*⟩

lemma (**in** *Module*) *mId-mHom*:$mId_M \in mHom\ R\ M\ M$
⟨*proof*⟩

lemma (**in** *Module*) *mHom-mId-bijec*:⟦R *module* N; $f \in mHom\ R\ M\ N$; $g \in mHom\ R\ N\ M$;
$compose\ (carrier\ M)\ g\ f = mId_M$; $compose\ (carrier\ N)\ f\ g = mId_N$⟧ $\Longrightarrow$
$bijec_{M,N}\ f$
⟨*proof*⟩

definition
sup-sharp :: [('*r*, '*n*) *Ring-scheme*, ('*b*, '*r*, '*m1*) *Module-scheme*,
('*c*, '*r*, '*m2*) *Module-scheme*, ('*a*, '*r*, '*m*) *Module-scheme*, '*b* ⇒ '*c*]
⇒ ('*c* ⇒ '*a*) ⇒ ('*b* ⇒ '*a*) **where**
sup-sharp R M N L u = (λ*f*∈*mHom R N L*. *compos M f u*)

definition
sub-sharp :: [('*r*, '*n*) *Ring-scheme*, ('*a*, '*r*, '*m*) *Module-scheme*,
('*b*, '*r*, '*m1*) *Module-scheme*, ('*c*, '*r*, '*m2*) *Module-scheme*, '*b* ⇒ '*c*]
⇒ ('*a* ⇒ '*b*) ⇒ ('*a* ⇒ '*c*) **where**
sub-sharp R L M N u = (λ*f*∈*mHom R L M*. *compos L u f*)

lemma (**in** *Module*) *sup-sharp-homTr*:⟦R *module* N; R *module* L; $u \in mHom\ R\ M\ N$;
$f \in mHom\ R\ N\ L\ \rrbracket \Longrightarrow$ *sup-sharp R M N L u f* $\in mHom\ R\ M\ L$
⟨*proof*⟩

lemma (**in** *Module*) *sup-sharp-hom*:⟦R *module* N; R *module* L; $u \in mHom\ R\ M\ N$⟧ $\Longrightarrow$
sup-sharp R M N L u $\in mHom\ R\ (HOM_R\ N\ L)\ (HOM_R\ M\ L)$
⟨*proof*⟩

lemma (**in** *Module*) *sub-sharp-homTr*:⟦R *module* N; R *module* L; $u \in mHom\ R\ M\ N$;
$f \in mHom\ R\ L\ M$⟧ $\Longrightarrow$ *sub-sharp R L M N u f* $\in mHom\ R\ L\ N$
⟨*proof*⟩

lemma (**in** *Module*) *sub-sharp-hom*:⟦R *module* N; R *module* L; $u \in mHom\ R\ M\ N$⟧ $\Longrightarrow$

$$sub\text{-}sharp\ R\ L\ M\ N\ u \in mHom\ R\ (HOM_R\ L\ M)\ (HOM_R\ L\ N)$$
⟨*proof*⟩

lemma (**in** *Module*) *mId-bijec*:$bijec_{M,M}\ (mId_M)$
⟨*proof*⟩

lemma (**in** *Module*) *invmfun-bijec*:⟦R *module* N; $f \in mHom\ R\ M\ N$; $bijec_{M,N}\ f$⟧
$\Longrightarrow$
$$bijec_{N,M}\ (invmfun\ R\ M\ N\ f)$$
⟨*proof*⟩

lemma (**in** *Module*) *misom-self*:$M \cong_R M$
⟨*proof*⟩

lemma (**in** *Module*) *misom-sym*:⟦R *module* N; $M \cong_R N$⟧ $\Longrightarrow$ $N \cong_R M$
⟨*proof*⟩

lemma (**in** *Module*) *misom-trans*:⟦R *module* L; R *module* N; $L \cong_R M$; $M \cong_R N$⟧
$\Longrightarrow$
$$L \cong_R N$$
⟨*proof*⟩

definition
$mr\text{-}coset :: ['a, ('a, 'b, 'more)\ Module\text{-}scheme, 'a\ set] \Rightarrow 'a\ set$ **where**
$mr\text{-}coset\ a\ M\ H = a \uplus_M H$

definition
$set\text{-}mr\text{-}cos :: [('a, 'b, 'more)\ Module\text{-}scheme, 'a\ set] \Rightarrow 'a\ set\ set$ **where**
$set\text{-}mr\text{-}cos\ M\ H = \{X.\ \exists\, a{\in}carrier\ M.\ X = a \uplus_M H\}$

definition
$mr\text{-}cos\text{-}sprod :: [('a, 'b, 'more)\ Module\text{-}scheme, 'a\ set] \Rightarrow 'b \Rightarrow 'a\ set \Rightarrow 'a\ set$ **where**
$mr\text{-}cos\text{-}sprod\ M\ H\ a\ X = \{z.\ \exists\, x{\in}X.\ \exists\, h{\in}H.\ z = h \pm_M (a \cdot_{s M} x)\}$

definition
$mr\text{-}cospOp :: [('a, 'b, 'more)\ Module\text{-}scheme, 'a\ set] \Rightarrow 'a\ set \Rightarrow 'a\ set \Rightarrow 'a\ set$ **where**
$mr\text{-}cospOp\ M\ H = (\lambda X.\ \lambda Y.\ c\text{-}top\ (b\text{-}ag\ M)\ H\ X\ Y)$

definition
$mr\text{-}cosmOp :: [('a, 'b, 'more)\ Module\text{-}scheme, 'a\ set] \Rightarrow 'a\ set \Rightarrow 'a\ set$ **where**
$mr\text{-}cosmOp\ M\ H = (\lambda X.\ c\text{-}iop\ (b\text{-}ag\ M)\ H\ X)$

definition
$qmodule :: [('a, 'r, 'more)\ Module\text{-}scheme, 'a\ set] \Rightarrow ('a\ set, 'r)\ Module$ **where**
$qmodule\ M\ H = (\!|\ carrier = set\text{-}mr\text{-}cos\ M\ H,\ pop = mr\text{-}cospOp\ M\ H,$

$mop = mr\text{-}cosmOp\ M\ H,\ zero = H,\ sprod = mr\text{-}cos\text{-}sprod\ M\ H\rparr$

definition
$sub\text{-}mr\text{-}set\text{-}cos :: [('a, 'r, 'more)\ Module\text{-}scheme, 'a\ set, 'a\ set] \Rightarrow 'a\ set\ set$ **where**
$sub\text{-}mr\text{-}set\text{-}cos\ M\ H\ N = \{X.\ \exists\, n{\in}N.\ X = n \uplus_M H\}$

abbreviation
$QMODULE$ (**infixl** $\langle'/'_m\rangle$ 200) **where**
$M\ /_m\ H == qmodule\ M\ H$

abbreviation
$SUBMRSET$ ($\langle(3\text{-}/\ _s'/'\text{-}/\ \text{-})\rangle$ [82,82,83]82) **where**
$N\ _s/_M\ H == sub\text{-}mr\text{-}set\text{-}cos\ M\ H\ N$

lemma (**in** *Module*) *qmodule-carr*:$submodule\ R\ M\ H \Longrightarrow carrier\ (qmodule\ M\ H) = set\text{-}mr\text{-}cos\ M\ H$
⟨*proof*⟩

lemma (**in** *Module*) *set-mr-cos-mem*:$[\![submodule\ R\ M\ H;\ m \in carrier\ M]\!] \Longrightarrow m \uplus_M H \in set\text{-}mr\text{-}cos\ M\ H$
⟨*proof*⟩

lemma (**in** *Module*) *mem-set-mr-cos*:$[\![submodule\ R\ M\ N;\ x \in set\text{-}mr\text{-}cos\ M\ N]\!] \Longrightarrow \exists\, m \in carrier\ M.\ x = m \uplus_M N$
⟨*proof*⟩

lemma (**in** *Module*) *m-in-mr-coset*:$[\![submodule\ R\ M\ H;\ m \in carrier\ M]\!] \Longrightarrow m \in m \uplus_M H$
⟨*proof*⟩

lemma (**in** *Module*) *mr-cos-h-stable*:$[\![submodule\ R\ M\ H;\ h \in H]\!] \Longrightarrow H = h \uplus_M H$
⟨*proof*⟩

lemma (**in** *Module*) *mr-cos-h-stable1*:$[\![submodule\ R\ M\ H;\ m \in carrier\ M;\ h \in H]\!] \Longrightarrow (m \pm h) \uplus_M H = m \uplus_M H$
⟨*proof*⟩

lemma (**in** *Module*) *x-in-mr-coset*:$[\![submodule\ R\ M\ H;\ m \in carrier\ M;\ x \in m \uplus_M H]\!] \Longrightarrow \exists\, h{\in}H.\ m \pm h = x$
⟨*proof*⟩

lemma (**in** *Module*) *mr-cos-sprodTr*:$[\![submodule\ R\ M\ H;\ a \in carrier\ R;\ m \in carrier\ M]\!] \Longrightarrow mr\text{-}cos\text{-}sprod\ M\ H\ a\ (m \uplus_M H) = (a \cdot_s m) \uplus_M H$
⟨*proof*⟩

lemma (**in** *Module*) *mr-cos-sprod-mem*:⟦*submodule R M H*; *a* ∈ *carrier R*;
X ∈ *set-mr-cos M H*⟧ ⟹ *mr-cos-sprod M H a X* ∈ *set-mr-cos M H*
⟨*proof*⟩

lemma (**in** *Module*) *mr-cos-sprod-assoc*:⟦*submodule R M H*; *a* ∈ *carrier R*;
b ∈ *carrier R*; *X* ∈ *set-mr-cos M H*⟧ ⟹ *mr-cos-sprod M H* (*a* ·$_{r\,R}$ *b*) *X* =
mr-cos-sprod M H a (*mr-cos-sprod M H b X*)
⟨*proof*⟩

lemma (**in** *Module*) *mr-cos-sprod-one*:⟦*submodule R M H*; *X* ∈ *set-mr-cos M H*⟧
⟹
mr-cos-sprod M H (1$_{r\,R}$) *X* = *X*
⟨*proof*⟩

lemma (**in** *Module*) *mr-cospOpTr*:⟦*submodule R M H*; *m* ∈ *carrier M*; *n* ∈ *carrier M*⟧
⟹ *mr-cospOp M H* (*m* ⊎$_M$ *H*) (*n* ⊎$_M$ *H*) = (*m* ± *n*) ⊎$_M$ *H*
⟨*proof*⟩

lemma(**in** *Module*) *mr-cos-sprod-distrib1*:⟦*submodule R M H*; *a* ∈ *carrier R*;
b ∈ *carrier R*; *X* ∈ *set-mr-cos M H*⟧ ⟹
mr-cos-sprod M H (*a* ±$_R$ *b*) *X* =
mr-cospOp M H (*mr-cos-sprod M H a X*) (*mr-cos-sprod M H b X*)
⟨*proof*⟩

lemma (**in** *Module*) *mr-cos-sprod-distrib2*:⟦*submodule R M H*;
a ∈ *carrier R*; *X* ∈ *set-mr-cos M H*; *Y* ∈ *set-mr-cos M H*⟧ ⟹
mr-cos-sprod M H a (*mr-cospOp M H X Y*) =
mr-cospOp M H (*mr-cos-sprod M H a X*) (*mr-cos-sprod M H a Y*)
⟨*proof*⟩

lemma (**in** *Module*) *mr-cosmOpTr*:⟦*submodule R M H*; *m* ∈ *carrier M*⟧ ⟹
mr-cosmOp M H (*m* ⊎$_M$ *H*) = (−$_a$ *m*) ⊎$_M$ *H*
⟨*proof*⟩

lemma (**in** *Module*) *mr-cos-oneTr*:*submodule R M H* ⟹ *H* = **0** ⊎$_M$ *H*
⟨*proof*⟩

lemma (**in** *Module*) *mr-cos-oneTr1*:⟦*submodule R M H*; *m* ∈ *carrier M*⟧ ⟹
mr-cospOp M H H (*m* ⊎$_M$ *H*) = *m* ⊎$_M$ *H*
⟨*proof*⟩

lemma (**in** *Module*) *qmodule-is-ag*:*submodule R M H* ⟹ *aGroup* (*M* /$_m$ *H*)
⟨*proof*⟩

lemma (**in** *Module*) *qmodule-module*:*submodule R M H* ⟹ *R module* (*M* /$_m$ *H*)
⟨*proof*⟩

definition
indmhom :: [('b, 'm) *Ring-scheme*, ('a, 'b, 'm1) *Module-scheme*,
('c, 'b, 'm2) *Module-scheme*, 'a ⇒ 'c] ⇒ 'a *set* ⇒ 'c **where**
indmhom R M N f = (λX∈ (*set-mr-cos M* ($ker_{M,N}$ *f*)). *f* (*SOME x*. *x* ∈ *X*))

abbreviation
INDMHOM (‹(4-$^\flat$_ _, _)› [92,92,92,93]92) **where**
$f^\flat_{R\ M,N}$ == *indmhom R M N f*

lemma (**in** *Module*) *indmhom-someTr*:⟦*R module N*; *f* ∈ *mHom R M N*;
X ∈ *set-mr-cos M* ($ker_{M,N}$ *f*)⟧ ⟹ *f* (*SOME xa*. *xa* ∈ *X*) ∈ *f* ‘(*carrier M*)
⟨*proof*⟩

lemma (**in** *Module*) *indmhom-someTr1*:⟦*R module N*; *f* ∈ *mHom R M N*; *m* ∈ *carrier M*⟧
⟹ *f* (*SOME xa*. *xa* ∈ (*ar-coset m M* ($ker_{M,N}$ *f*))) = *f m*
⟨*proof*⟩

lemma (**in** *Module*) *indmhom-someTr2*:⟦*R module N*; *f* ∈ *mHom R M N*;
submodule R M H; *m* ∈ *carrier M*; *H* ⊆ $ker_{M,N}$ *f*⟧ ⟹
f (*SOME xa*. *xa* ∈ *m* ⊎$_M$ *H*) = *f m*
⟨*proof*⟩

lemma (**in** *Module*) *indmhomTr1*:⟦*R module N*; *f* ∈ *mHom R M N*; *m* ∈ *carrier M*⟧ ⟹
($f^\flat_{R\ M,N}$) (*m* ⊎$_M$ ($ker_{M,N}$ *f*)) = *f m*
⟨*proof*⟩

lemma (**in** *Module*) *indmhomTr2*:⟦*R module N*; *f* ∈ *mHom R M N*⟧
⟹ ($f^\flat_{R\ M,N}$) ∈ *set-mr-cos M* ($ker_{M,N}$ *f*) → *carrier N*
⟨*proof*⟩

lemma (**in** *Module*) *indmhom*:⟦*R module N*; *f* ∈ *mHom R M N*⟧
⟹ ($f^\flat_{R\ M,N}$) ∈ *mHom R* (*M* /$_m$ ($ker_{M,N}$ *f*)) *N*
⟨*proof*⟩

lemma (**in** *Module*) *indmhom-injec*:⟦*R module N*; *f* ∈ *mHom R M N*⟧ ⟹
$injec_{(M\ /_m\ (ker_{M,N}\ f)),N}$ ($f^\flat_{R\ M,N}$)
⟨*proof*⟩

lemma (**in** *Module*) *indmhom-surjec1*:⟦*R module N*; $surjec_{M,N}$ *f*;
f ∈ *mHom R M N*⟧ ⟹ $surjec_{(M\ /_m\ (ker_{M,N}\ f)),N}$ ($f^\flat_{R\ M,N}$)
⟨*proof*⟩

lemma (**in** *Module*) *module-homTr*:⟦*R module N*; *f* ∈ *mHom R M N*⟧ ⟹
f ∈ *mHom R M* ($mimg_{R\ M,N}$ *f*)
⟨*proof*⟩

lemma (**in** *Module*) *ker-to-mimg*:⟦*R module N*; $f \in$ *mHom R M N*⟧ ⟹
$ker_{M,mimg_{R\ M,N}\ f}\ f = ker_{M,N}\ f$
⟨*proof*⟩

lemma (**in** *Module*) *module-homTr1*:⟦*R module N*; $f \in$ *mHom R M N*⟧ ⟹
$(mimg_{R\ (M\ /_m\ (ker_{M,N}\ f)),N}\ (f^\flat_{R\ M,N})) = mimg_{R\ M,N}\ f$ ⟨*proof*⟩

lemma (**in** *Module*) *module-Homth-1*:⟦*R module N*; $f \in$ *mHom R M N*⟧ ⟹
$M\ /_m\ (ker_{M,N}\ f) \cong_R mimg_{R\ M,N}\ f$
⟨*proof*⟩

definition
mpj :: [('*a*, '*r*, '*m*) *Module-scheme*, '*a set*] ⇒ ('*a* => '*a set*) **where**
mpj M H = (λ*x*∈*carrier M*. *x* ⊎$_M$ *H*)

lemma (**in** *Module*) *elem-mpj*:⟦*m* ∈ *carrier M*; *submodule R M H*⟧ ⟹
mpj M H m = *m* ⊎$_M$ *H*
⟨*proof*⟩

lemma (**in** *Module*) *mpj-mHom*:*submodule R M H* ⟹ *mpj M H* ∈ *mHom R M* (*M* /$_m$ *H*)
⟨*proof*⟩

lemma (**in** *Module*) *mpj-mem*:⟦*submodule R M H*; *m* ∈ *carrier M*⟧ ⟹
mpj M H m ∈ *carrier* (*M* /$_m$ *H*)
⟨*proof*⟩

lemma (**in** *Module*) *mpj-surjec*:*submodule R M H* ⟹
$surjec_{M,(M\ /_m\ H)}$ (*mpj M H*)
⟨*proof*⟩

lemma (**in** *Module*) *mpj-0*:⟦*submodule R M H*; *h* ∈ *H*⟧ ⟹
mpj M H h = $\mathbf{0}_{(M\ /_m\ H)}$
⟨*proof*⟩

lemma (**in** *Module*) *mker-of-mpj*:*submodule R M H* ⟹
$ker_{M,(M\ /_m\ H)}$ (*mpj M H*) = *H*
⟨*proof*⟩

lemma (**in** *Module*) *indmhom1*:⟦*submodule R M H*; *R module N*; *f* ∈ *mHom R M N*; *H* ⊆ $ker_{M,N}\ f$⟧ ⟹ ∃!*g*. *g* ∈ (*mHom R* (*M* /$_m$ *H*) *N*) ∧ (*compos M g* (*mpj M H*)) = *f*
⟨*proof*⟩

definition
mQmp :: [('*a*, '*r*, '*m*) *Module-scheme*, '*a set*, '*a set*] ⇒
('*a set* ⇒ '*a set*) **where**

$mQmp\ M\ H\ N = (\lambda X \in \textit{set-mr-cos}\ M\ H.\ \{z.\ \exists\ x \in X.\ \exists\ y \in N.\ (y \pm_M x = z)\})$

abbreviation
$MQP\ \ (\langle(3Mp\_\ \_,\_)\rangle\ [82,82,83]82)$ **where**
$Mp_{M\ H,N} == mQmp\ M\ H\ N$

lemma (**in** *Module*) *mQmpTr0*:$[\![\textit{submodule}\ R\ M\ H;\ \textit{submodule}\ R\ M\ N;\ H \subseteq N;$
$m \in \textit{carrier}\ M]\!] \Longrightarrow\ mQmp\ M\ H\ N\ (m \uplus_M H) = m \uplus_M N$
⟨*proof*⟩

lemma (**in** *Module*) *mQmpTr1*:$[\![\textit{submodule}\ R\ M\ H;\ \textit{submodule}\ R\ M\ N;\ H \subseteq N;$
$m \in \textit{carrier}\ M;\ n \in \textit{carrier}\ M;\ m \uplus_M H = n \uplus_M H]\!] \Longrightarrow\ m \uplus_M N = n \uplus_M N$
⟨*proof*⟩

lemma (**in** *Module*) *mQmpTr2*:$[\![\textit{submodule}\ R\ M\ H;\ \textit{submodule}\ R\ M\ N;\ H \subseteq N\ ;$
$X \in \textit{carrier}\ (M\ /_m\ H)]\!] \Longrightarrow (mQmp\ M\ H\ N)\ X \in \textit{carrier}\ (M\ /_m\ N)$
⟨*proof*⟩

lemma (**in** *Module*) *mQmpTr2-1*:$[\![\textit{submodule}\ R\ M\ H;\ \textit{submodule}\ R\ M\ N;\ H \subseteq N$
$]\!]$
$\Longrightarrow mQmp\ M\ H\ N \in \textit{carrier}\ (M\ /_m\ H) \to \textit{carrier}\ (M\ /_m\ N)$
⟨*proof*⟩

lemma (**in** *Module*) *mQmpTr3*:$[\![\textit{submodule}\ R\ M\ H;\ \textit{submodule}\ R\ M\ N;\ H \subseteq N\ ;$
$X \in \textit{carrier}\ (M\ /_m\ H);\ Y \in \textit{carrier}\ (M\ /_m\ H)]\!] \Longrightarrow (mQmp\ M\ H\ N)\ (\textit{mr-cospOp}$
$M\ H\ X\ Y) = \textit{mr-cospOp}\ M\ N\ ((mQmp\ M\ H\ N)\ X)\ ((mQmp\ M\ H\ N)\ Y)$
⟨*proof*⟩

lemma (**in** *Module*) *mQmpTr4*:$[\![\textit{submodule}\ R\ M\ H;\ \textit{submodule}\ R\ M\ N;\ H \subseteq N;$
$a \in N]\!] \Longrightarrow \textit{mr-coset}\ a\ (\textit{mdl}\ M\ N)\ H = \textit{mr-coset}\ a\ M\ H$
⟨*proof*⟩

lemma (**in** *Module*) *mQmp-mHom*:$[\![\textit{submodule}\ R\ M\ H;\ \textit{submodule}\ R\ M\ N;\ H \subseteq$
$N]\!] \Longrightarrow$
$(Mp_{M\ H,N}) \in \textit{mHom}\ R\ (M\ /_m\ H)\ (M\ /_m\ N)$
⟨*proof*⟩

lemma (**in** *Module*) *Mp-surjec*:$[\![\textit{submodule}\ R\ M\ H;\ \textit{submodule}\ R\ M\ N;\ H \subseteq N]\!]$
$\Longrightarrow$
$\textit{surjec}_{(M\ /_m\ H),(M\ /_m\ N)}\ (Mp_{M\ H,N})$
⟨*proof*⟩

lemma (**in** *Module*) *kerQmp*:$[\![\textit{submodule}\ R\ M\ H;\ \textit{submodule}\ R\ M\ N;\ H \subseteq N]\!]$
$\Longrightarrow \textit{ker}_{(M\ /_m\ H),(M\ /_m\ N)}\ (Mp_{M\ H,N}) = \textit{carrier}\ ((\textit{mdl}\ M\ N)\ /_m\ H)$
⟨*proof*⟩

lemma (**in** *Module*) *misom2Tr*:⟦*submodule R M H*; *submodule R M N*; $H \subseteq N$⟧
$\Longrightarrow$
$(M\ /_m\ H)\ /_m\ (carrier\ ((mdl\ M\ N)\ /_m\ H)) \cong_R (M\ /_m\ N)$
⟨*proof*⟩

lemma (**in** *Module*) *eq-class-of-Submodule*:⟦*submodule R M H*; *submodule R M N*;
$H \subseteq N$⟧ $\Longrightarrow$ $carrier\ ((mdl\ M\ N)\ /_m\ H) = N\ {}_s/_M\ H$
⟨*proof*⟩

theorem (**in** *Module*) *misom2*:⟦*submodule R M H*; *submodule R M N*; $H \subseteq N$⟧
$\Longrightarrow$
$(M\ /_m\ H)\ /_m\ (N\ {}_s/_M\ H) \cong_R (M\ /_m\ N)$
⟨*proof*⟩

primrec *natm* :: ('*a*, '*m*) *aGroup-scheme* => *nat* ⇒ '*a* => '*a*
where
natm-0: *natm M 0 x* = $\mathbf{0}_M$
| *natm-Suc*: *natm M* (*Suc n*) *x* = (*natm M n x*) $\pm_M$ *x*

definition
finitesum-base :: [('*a*, '*r*, '*m*) *Module-scheme*, '*b set*, '*b* ⇒ '*a set*]
⇒ '*a set* **where**
finitesum-base M I f = $\bigcup\{f\ i \mid i.\ i \in I\}$

definition
finitesum :: [('*a*, '*r*, '*m*) *Module-scheme*, '*b set*, '*b* ⇒ '*a set*]
⇒ '*a set* **where**
finitesum M I f = {*x*. ∃ *n*. ∃ *g*. *g* ∈ {*j*. *j* ≤ (*n*::*nat*)} → *finitesum-base M I f*
∧ *x* = *nsum M g n*}

lemma (**in** *Module*) *finitesumbase-sub-carrier*:*f* ∈ *I* → {*X*. *submodule R M X*}
$\Longrightarrow$
finitesum-base M I f ⊆ *carrier M*
⟨*proof*⟩

lemma (**in** *Module*) *finitesum-sub-carrier*:*f* ∈ *I* → {*X*. *submodule R M X*} $\Longrightarrow$
finitesum M I f ⊆ *carrier M*
⟨*proof*⟩

lemma (**in** *Module*) *finitesum-inc-zero*:⟦*f* ∈ *I* → {*X*. *submodule R M X*}; *I* ≠ {}⟧
$\Longrightarrow$ $\mathbf{0}$ ∈ *finitesum M I f*
⟨*proof*⟩

lemma (**in** *Module*) *finitesum-mOp-closed*:
⟦*f* ∈ *I* → {*X*. *submodule R M X*}; *I* ≠ {}; *a* ∈ *finitesum M I f*⟧ $\Longrightarrow$
$-_a\ a \in$ *finitesum M I f*

⟨*proof*⟩

lemma (**in** *Module*) *finitesum-pOp-closed*:
⟦$f \in I \to \{X.\ submodule\ R\ M\ X\}$; $a \in finitesum\ M\ I\ f$; $b \in finitesum\ M\ I\ f$⟧
$\Longrightarrow$ $a \pm b \in finitesum\ M\ I\ f$
⟨*proof*⟩

lemma (**in** *Module*) *finitesum-sprodTr*:⟦$f \in I \to \{X.\ submodule\ R\ M\ X\}$; $I \neq \{\}$;
$r \in carrier\ R$⟧ $\Longrightarrow$ $g \in \{j.\ j \leq (n::nat)\} \to (finitesum\text{-}base\ M\ I\ f)$
$\longrightarrow$ $r \cdot_s (nsum\ M\ g\ n) =$ $nsum\ M\ (\lambda x.\ r \cdot_s (g\ x))\ n$
⟨*proof*⟩

lemma (**in** *Module*) *finitesum-sprod*:⟦$f \in I \to \{X.\ submodule\ R\ M\ X\}$; $I \neq \{\}$;
$r \in carrier\ R$; $g \in \{j.\ j \leq (n::nat)\} \to (finitesum\text{-}base\ M\ I\ f)$ ⟧ $\Longrightarrow$
$r \cdot_s (nsum\ M\ g\ n) =$ $nsum\ M\ (\lambda x.\ r \cdot_s (g\ x))\ n$
⟨*proof*⟩

lemma (**in** *Module*) *finitesum-subModule*:⟦$f \in I \to \{X.\ submodule\ R\ M\ X\}$; $I \neq \{\}$⟧
$\Longrightarrow$ $submodule\ R\ M\ (finitesum\ M\ I\ f)$
⟨*proof*⟩

lemma (**in** *Module*) *sSum-cont-H*:⟦$submodule\ R\ M\ H$; $submodule\ R\ M\ K$⟧ $\Longrightarrow$
$H \subseteq$ $H \mp K$
⟨*proof*⟩

lemma (**in** *Module*) *sSum-commute*:⟦$submodule\ R\ M\ H$; $submodule\ R\ M\ K$⟧ $\Longrightarrow$
$H \mp K =$ $K \mp H$
⟨*proof*⟩

lemma (**in** *Module*) *Sum-of-SubmodulesTr*:⟦$submodule\ R\ M\ H$; $submodule\ R\ M\ K$⟧
$\Longrightarrow$
$g \in \{j.\ j \leq (n::nat)\} \to H \cup K \longrightarrow \Sigma_e\ M\ g\ n \in H \mp K$
⟨*proof*⟩

lemma (**in** *Module*) *sSum-two-Submodules*:⟦$submodule\ R\ M\ H$; $submodule\ R\ M\ K$⟧
$\Longrightarrow$
$submodule\ R\ M\ (H \mp K)$
⟨*proof*⟩

definition
$iotam :: [('a, 'r, 'm)\ Module\text{-}scheme, 'a\ set, 'a\ set] \Rightarrow ('a \Rightarrow 'a)$
$(\langle(3\iota m_{-}\ _{-},_{-})\rangle\ [82, 82, 83]82)$ **where**
$\iota m_{M\ H,K} = (\lambda x{\in}H.\ (x \pm_M \mathbf{0}_M))$

lemma (**in** *Module*) *iotam-mHom*:⟦$submodule\ R\ M\ H$; $submodule\ R\ M\ K$⟧
$\Longrightarrow$ $\iota m_{M\ H,K} \in mHom\ R\ (mdl\ M\ H)\ (mdl\ M\ (H \mp K))$
⟨*proof*⟩

lemma (**in** *Module*) *mhomom3Tr*:⟦*submodule R M H*; *submodule R M K*⟧ ⟹
submodule R (*mdl M* (*H* ∓ *K*)) *K*
⟨*proof*⟩

lemma (**in** *Module*) *mhomom3Tr0*:⟦*submodule R M H*; *submodule R M K*⟧
⟹ *compos* (*mdl M H*) (*mpj* (*mdl M* (*H* ∓ *K*)) *K*) (*ιm*$_{M\ H,K}$)
∈ *mHom R* (*mdl M H*) (*mdl M* (*H* ∓ *K*) /$_m$ *K*)
⟨*proof*⟩

lemma (**in** *Module*) *mhomom3Tr1*:⟦*submodule R M H*; *submodule R M K*⟧ ⟹
surjec$_{(mdl\ M\ H),((mdl\ M\ (H \mp K))/_m\ K)}$
(*compos* (*mdl M H*) (*mpj* (*mdl M* (*H* ∓ *K*)) *K*) (*ιm*$_{M\ H,K}$))
⟨*proof*⟩

lemma (**in** *Module*) *mhomom3Tr2*:⟦*submodule R M H*; *submodule R M K*⟧ ⟹
ker$_{(mdl\ M\ H),((mdl\ M\ (H \mp K))\ /_m\ K)}$
(*compos* (*mdl M H*) (*mpj* (*mdl M* (*H* ∓ *K*)) *K*) (*ιm*$_{M\ H,K}$)) = *H* ∩ *K*
⟨*proof*⟩

lemma (**in** *Module*) *mhomom-3*:⟦*submodule R M H*; *submodule R M K*⟧ ⟹
(*mdl M H*) /$_m$ (*H* ∩ *K*) ≅$_R$ (*mdl M* (*H* ∓ *K*)) /$_m$ *K*
⟨*proof*⟩

definition
l-comb :: [('*r*, '*m*) *Ring-scheme*, ('*a*, '*r*, '*m1*) *Module-scheme*, *nat*] ⇒
(*nat* ⇒ '*r*) ⇒ (*nat* ⇒ '*a*) ⇒ '*a* **where**
l-comb R M n s m = *nsum M* (λ*j*. (*s j*) ·$_{sM}$ (*m j*)) *n*

definition
linear-span :: [('*r*, '*m*) *Ring-scheme*, ('*a*, '*r*, '*m1*) *Module-scheme*, '*r set*,
'*a set*] ⇒ '*a set* **where**
linear-span R M A H = (*if H* = {} *then* {**0**$_M$} *else*
{*x*. ∃ *n*. ∃ *f* ∈ {*j*. *j* ≤ (*n*::*nat*)} → *H*.
∃ *s*∈{*j*. *j* ≤ (*n*::*nat*)} → *A*. *x* = *l-comb R M n s f*})

definition
coefficient :: [('*r*, '*m*) *Ring-scheme*, ('*a*, '*r*, '*m1*) *Module-scheme*,
nat, *nat* ⇒ '*r*, *nat* ⇒ '*a*] ⇒ *nat* ⇒ '*r* **where**
coefficient R M n s m j = *s j*

definition
body :: [('*r*, '*m*) *Ring-scheme*, ('*a*, '*r*, '*m1*) *Module-scheme*, *nat*, *nat* ⇒ '*r*,
nat ⇒ '*a*] ⇒ *nat* ⇒ '*a* **where**
body R M n s m j = *m j*

lemma (**in** *Module*) *l-comb-mem-linear-span*:⟦*ideal R A*; *H* ⊆ *carrier M*;
s ∈ {*j*. *j* ≤ (*n*::*nat*)} → *A*; *f* ∈ {*j*. *j* ≤ *n*} → *H*⟧ ⟹
l-comb R M n s f ∈ *linear-span R M A H*
⟨*proof*⟩

lemma (**in** *Module*) *linear-comb-eqTr*:$H \subseteq carrier\ M \Longrightarrow$
$s \in \{j.\ j \leq (n::nat)\} \to carrier\ R \wedge$
$f \in \{j.\ j \leq n\} \to H \wedge$
$g \in \{j.\ j \leq n\} \to H \wedge$
$(\forall j{\in}\{j.\ j \leq n\}.\ f\ j = g\ j) \longrightarrow$
$l\text{-}comb\ R\ M\ n\ s\ f = l\text{-}comb\ R\ M\ n\ s\ g$
⟨*proof*⟩

lemma (**in** *Module*) *linear-comb-eq*:$[\![H \subseteq carrier\ M;$
$s \in \{j.\ j \leq (n::nat)\} \to carrier\ R;\ f \in \{j.\ j \leq n\} \to H;$
$g \in \{j.\ j \leq n\} \to H;\ \forall j{\in}\{j.\ j \leq n\}.\ f\ j = g\ j]\!] \Longrightarrow$
$l\text{-}comb\ R\ M\ n\ s\ f = l\text{-}comb\ R\ M\ n\ s\ g$
⟨*proof*⟩

lemma (**in** *Module*) *l-comb-Suc*:$[\![H \subseteq carrier\ M;\ ideal\ R\ A;$
$s \in \{j.\ j \leq (Suc\ n)\} \to carrier\ R;\ f \in \{j.\ j \leq (Suc\ n)\} \to H]\!] \Longrightarrow$
$l\text{-}comb\ R\ M\ (Suc\ n)\ s\ f = l\text{-}comb\ R\ M\ n\ s\ f \pm s\ (Suc\ n) \cdot_s f\ (Suc\ n)$
⟨*proof*⟩

lemma (**in** *Module*) *l-comb-jointfun-jj*:$[\![H \subseteq carrier\ M;\ ideal\ R\ A;$
$s \in \{j.\ j \leq (n::nat)\} \to A;\ f \in \{j.\ j \leq (n::nat)\} \to H;$
$t \in \{j.\ j \leq (m::nat)\} \to A;\ g \in \{j.\ j \leq (m::nat)\} \to H]\!] \Longrightarrow$
$nsum\ M\ (\lambda j.\ (jointfun\ n\ s\ m\ t)\ j \cdot_s (jointfun\ n\ f\ m\ g)\ j)\ n =$
$nsum\ M\ (\lambda j.\ s\ j \cdot_s f\ j)\ n$
⟨*proof*⟩

lemma (**in** *Module*) *l-comb-jointfun-jj1*:$[\![H \subseteq carrier\ M;\ ideal\ R\ A;$
$s \in \{j.\ j \leq (n::nat)\} \to A;\ f \in \{j.\ j \leq (n::nat)\} \to H;$
$t \in \{j.\ j \leq (m::nat)\} \to A;\ g \in \{j.\ j \leq (m::nat)\} \to H]\!] \Longrightarrow$
$l\text{-}comb\ R\ M\ n\ (jointfun\ n\ s\ m\ t)\ (jointfun\ n\ f\ m\ g) =$
$l\text{-}comb\ R\ M\ n\ s\ f$
⟨*proof*⟩

lemma (**in** *Module*) *l-comb-jointfun-jf*:$[\![H \subseteq carrier\ M;\ ideal\ R\ A;$
$s \in \{j.\ j \leq (n::nat)\} \to A;\ f \in \{j.\ j \leq Suc\ (n + m)\} \to H;$
$t \in \{j.\ j \leq (m::nat)\} \to A]\!] \Longrightarrow$
$nsum\ M\ (\lambda j.\ (jointfun\ n\ s\ m\ t)\ j \cdot_s f\ j)\ n =$
$nsum\ M\ (\lambda j.\ s\ j \cdot_s f\ j)\ n$
⟨*proof*⟩

lemma (**in** *Module*) *l-comb-jointfun-jf1*:$[\![H \subseteq carrier\ M;\ ideal\ R\ A;$
$s \in \{j.\ j \leq (n::nat)\} \to A;\ f \in \{j.\ j \leq Suc\ (n + m)\} \to H;$
$t \in \{j.\ j \leq (m::nat)\} \to A]\!] \Longrightarrow$
$l\text{-}comb\ R\ M\ n\ (jointfun\ n\ s\ m\ t)\ f = l\text{-}comb\ R\ M\ n\ s\ f$
⟨*proof*⟩

lemma (**in** *Module*) *l-comb-jointfun-fj*:$[\![H \subseteq carrier\ M;\ ideal\ R\ A;$
$s \in \{j.\ j \leq Suc\ (n + m)\} \to A;\ f \in \{j.\ j \leq (n::nat)\} \to H;$

$g \in \{j.\ j \leq (m::nat)\} \to H\rrbracket \Longrightarrow$
$nsum\ M\ (\lambda j.\ s\ j \cdot_s (jointfun\ n\ f\ m\ g)\ j)\ n =$
$nsum\ M\ (\lambda j.\ s\ j \cdot_s f\ j)\ n$
⟨*proof*⟩

lemma (**in** *Module*) *l-comb-jointfun-fj1*:⟦$H \subseteq carrier\ M$; $ideal\ R\ A$;
$s \in \{j.\ j \leq Suc\ (n + m)\} \to A$; $f \in \{j.\ j \leq (n::nat)\} \to H$;
$g \in \{j.\ j \leq (m::nat)\} \to H\rrbracket \Longrightarrow$
$l\text{-}comb\ R\ M\ n\ s\ (jointfun\ n\ f\ m\ g) = l\text{-}comb\ R\ M\ n\ s\ f$
⟨*proof*⟩

lemma (**in** *Module*) *linear-comb0-1Tr*:$H \subseteq carrier\ M \Longrightarrow$
$s \in \{j.\ j \leq (n::nat)\} \to \{\mathbf{0}_R\} \wedge$
$m \in \{j.\ j \leq n\} \to H \longrightarrow l\text{-}comb\ R\ M\ n\ s\ m = \mathbf{0}_M$
⟨*proof*⟩

lemma (**in** *Module*) *linear-comb0-1*:⟦$H \subseteq carrier\ M$;
$s \in \{j.\ j \leq (n::nat)\} \to \{\mathbf{0}_R\}$; $m \in \{j.\ j \leq n\} \to H\ \rrbracket \Longrightarrow$
$l\text{-}comb\ R\ M\ n\ s\ m = \mathbf{0}_M$
⟨*proof*⟩

lemma (**in** *Module*) *linear-comb0-2Tr*:$ideal\ R\ A \Longrightarrow s \in \{j.\ j \leq (n::nat)\} \to A$
$\wedge\ \ m \in \{j.\ j \leq n\} \to \{\mathbf{0}_M\} \longrightarrow l\text{-}comb\ R\ M\ n\ s\ m = \mathbf{0}_M$
⟨*proof*⟩

lemma (**in** *Module*) *linear-comb0-2*:⟦$ideal\ R\ A$; $\ s \in \{j.\ j \leq (n::nat)\} \to A$;
$m \in \{j.\ j \leq n\} \to \{\mathbf{0}_M\}\ \rrbracket \Longrightarrow\ l\text{-}comb\ R\ M\ n\ s\ m = \mathbf{0}_M$
⟨*proof*⟩

lemma (**in** *Module*) *liear-comb-memTr*:⟦$ideal\ R\ A$; $H \subseteq carrier\ M\rrbracket \Longrightarrow$
$\forall s.\ \forall m.\ s \in \{j.\ j \leq (n::nat)\} \to A \wedge$
$m \in \{j.\ j \leq n\} \to H \longrightarrow l\text{-}comb\ R\ M\ n\ s\ m \in carrier\ M$
⟨*proof*⟩

lemma (**in** *Module*) *l-comb-mem*:⟦$ideal\ R\ A$; $H \subseteq carrier\ M$;
$s \in \{j.\ j \leq (n::nat)\} \to A$; $m \in \{j.\ j \leq n\} \to H\rrbracket \Longrightarrow$
$l\text{-}comb\ R\ M\ n\ s\ m \in carrier\ M$
⟨*proof*⟩

lemma (**in** *Module*) *l-comb-transpos*: ⟦$ideal\ R\ A$; $H \subseteq carrier\ M$;
$s \in \{l.\ l \leq Suc\ n\} \to A$; $f \in \{l.\ l \leq Suc\ n\} \to H$;
$j < Suc\ n\ \rrbracket \Longrightarrow$
$\Sigma_e\ M\ (cmp\ (\lambda k.\ s\ k \cdot_s f\ k)\ (transpos\ j\ (Suc\ n)))\ (Suc\ n) =$
$\Sigma_e\ M\ (\lambda k.\ (cmp\ s\ (transpos\ j\ (Suc\ n)))\ k \cdot_s$
$(cmp\ f\ (transpos\ j\ (Suc\ n)))\ k)\ (Suc\ n)$
⟨*proof*⟩

lemma (**in** *Module*) *l-comb-transpos1*: ⟦$ideal\ R\ A$; $H \subseteq carrier\ M$;
$s \in \{l.\ l \leq Suc\ n\} \to A$; $f \in \{l.\ l \leq Suc\ n\} \to H$; $j < Suc\ n\ \rrbracket \Longrightarrow$

l-comb R M (Suc n) s f =
l-comb R M (Suc n) (cmp s (transpos j (Suc n))) (cmp f (transpos j (Suc n)))
⟨*proof*⟩

lemma (**in** *Module*) *sc-linear-span*:⟦*ideal R A*; *H* ⊆ *carrier M*; *a* ∈ *A*; *h* ∈ *H*⟧ ⟹ *a* ·$_s$ *h* ∈ *linear-span R M A H*
⟨*proof*⟩

lemma (**in** *Module*) *l-span-cont-H*:*H* ⊆ *carrier M* ⟹ *H* ⊆ *linear-span R M* (*carrier R*) *H*
⟨*proof*⟩

lemma (**in** *Module*) *linear-span-inc-0*:⟦*ideal R A*; *H* ⊆ *carrier M*⟧ ⟹ **0** ∈ *linear-span R M A H*
⟨*proof*⟩

lemma (**in** *Module*) *linear-span-iOp-closedTr1*:⟦*ideal R A*; *s* ∈ {*j*. *j* ≤ (*n*::*nat*)} → *A*⟧ ⟹ (λ*x*∈{*j*. *j* ≤ *n*}. −$_a$ *R* (*s x*)) ∈ {*j*. *j* ≤ *n*} → *A*
⟨*proof*⟩

lemma (**in** *Module*) *l-span-gen-mono*:⟦*K* ⊆ *H*; *H* ⊆ *carrier M*; *ideal R A*⟧ ⟹ *linear-span R M A K* ⊆ *linear-span R M A H*
⟨*proof*⟩

lemma (**in** *Module*) *l-comb-add*:⟦*ideal R A*; *H* ⊆ *carrier M*; *s* ∈ {*j*. *j* ≤ (*n*::*nat*)} → *A*; *f* ∈ {*j*. *j* ≤ *n*} → *H*; *t* ∈ {*j*. *j* ≤ (*m*::*nat*)} → *A*; *g* ∈ {*j*. *j* ≤ *m*} → *H*⟧ ⟹ *l-comb R M* (*Suc* (*n* + *m*)) (*jointfun n s m t*) (*jointfun n f m g*) = *l-comb R M n s f* ± *l-comb R M m t g*
⟨*proof*⟩

lemma (**in** *Module*) *l-comb-add1Tr*:⟦*ideal R A*; *H* ⊆ *carrier M*⟧ ⟹ *f* ∈ {*j*. *j* ≤ (*n*::*nat*)} → *H* ∧ *s* ∈ {*j*. *j* ≤ *n*} → *A* ∧ *t* ∈ {*j*. *j* ≤ *n*} → *A* ⟶ *l-comb R M n* (λ*x*∈{*j*. *j* ≤ *n*}. (*s x*) ±$_R$ (*t x*)) *f* = *l-comb R M n s f* ± *l-comb R M n t f*
⟨*proof*⟩

lemma (**in** *Module*) *l-comb-add1*:⟦*ideal R A*; *H* ⊆ *carrier M*; *f* ∈ {*j*. *j* ≤ (*n*::*nat*)} → *H*; *s* ∈ {*j*. *j* ≤ *n*} → *A*; *t* ∈ {*j*. *j* ≤ *n*} → *A* ⟧ ⟹ *l-comb R M n* (λ*x*∈{*j*. *j* ≤ *n*}. (*s x*) ±$_R$ (*t x*)) *f* = *l-comb R M n s f* ± *l-comb R M n t f*
⟨*proof*⟩

lemma (**in** *Module*) *linear-span-iOp-closedTr2*:⟦*ideal R A*; *H* ⊆ *carrier M*; *f* ∈ {*j*. *j* ≤ (*n*::*nat*)} → *H*; *s* ∈ {*j*. *j* ≤ *n*} → *A*⟧ ⟹ −$_a$ (*l-comb R M n s f*) = *l-comb R M n* (λ*x*∈{*j*. *j* ≤ *n*}. −$_a$ *R* (*s x*)) *f*
⟨*proof*⟩

lemma (**in** *Module*) *linear-span-iOp-closed*:⟦*ideal R A*; $H \subseteq$ *carrier M*;
$a \in$ *linear-span R M A H*⟧ $\Longrightarrow -_a\ a \in$ *linear-span R M A H*
⟨*proof*⟩

lemma (**in** *Module*) *linear-span-pOp-closed*:
⟦*ideal R A*; $H \subseteq$ *carrier M*; $a \in$ *linear-span R M A H*; $b \in$ *linear-span R M A H*⟧
$\Longrightarrow a \pm b \in$ *linear-span R M A H*
⟨*proof*⟩

lemma (**in** *Module*) *l-comb-scTr*:⟦*ideal R A*; $H \subseteq$ *carrier M*;
$r \in$ *carrier R*; $H \neq \{\}$⟧ $\Longrightarrow s \in \{j.\ j \leq (n::nat)\} \to A \wedge$
$g \in \{j.\ j \leq n\} \to\ H \longrightarrow r \cdot_s (nsum\ M\ (\lambda k.\ (s\ k) \cdot_s (g\ k))\ \ n) =$
$nsum\ M\ (\lambda k.\ r \cdot_s ((s\ k) \cdot_s (g\ k)))\ n$
⟨*proof*⟩

lemma (**in** *Module*) *l-comb-sc1Tr*:⟦*ideal R A*; $H \subseteq$ *carrier M*;
$r \in$ *carrier R*; $H \neq \{\}$⟧ $\Longrightarrow s \in \{j.\ j \leq (n::nat)\} \to A \wedge$
$g \in \{j.\ j \leq n\} \to\ H \longrightarrow r \cdot_s (nsum\ M\ (\lambda k.\ (s\ k) \cdot_s (g\ k))\ \ n) =$
$nsum\ M\ (\lambda k.\ (r \cdot_{r\,R} (s\ k)) \cdot_s (g\ k))\ n$
⟨*proof*⟩

lemma (**in** *Module*) *l-comb-sc*:⟦*ideal R A*; $H \subseteq$ *carrier M*; $r \in$ *carrier R*;
$s \in \{j.\ j \leq (n::nat)\} \to A$; $g \in \{j.\ j \leq n\} \to\ H$⟧ $\Longrightarrow$
$r \cdot_s (nsum\ M\ (\lambda k.\ (s\ k) \cdot_s (g\ k))\ n) = nsum\ M\ (\lambda k.\ r \cdot_s ((s\ k) \cdot_s (g\ k)))\ n$
⟨*proof*⟩

lemma (**in** *Module*) *l-comb-sc1*:⟦*ideal R A*; $H \subseteq$ *carrier M*; $r \in$ *carrier R*;
$s \in \{j.\ j \leq (n::nat)\} \to A$; $g \in \{j.\ j \leq n\} \to\ H$⟧ $\Longrightarrow$
$r \cdot_s (nsum\ M\ (\lambda k.\ (s\ k) \cdot_s (g\ k))\ n) = nsum\ M\ (\lambda k.\ (r \cdot_{r\,R} (s\ k)) \cdot_s (g\ k))\ n$
⟨*proof*⟩

lemma (**in** *Module*) *linear-span-sc-closed*:⟦*ideal R A*; $H \subseteq$ *carrier M*;
$r \in$ *carrier R*; $x \in$ *linear-span R M A H*⟧ $\Longrightarrow r \cdot_s x \in$ *linear-span R M A H*
⟨*proof*⟩

lemma (**in** *Module*) *mem-single-l-spanTr*:⟦*ideal R A*; $h \in$ *carrier M*⟧ $\Longrightarrow$
$s \in \{j.\ j \leq (n::nat)\} \to A \wedge$
$f \in \{j.\ j \leq n\} \to \{h\} \wedge$ *l-comb R M n s f* $\in$ *linear-span R M A* $\{h\}$
$\longrightarrow (\exists\, a \in A.$ *l-comb R M n s f* $= a \cdot_s h)$
⟨*proof*⟩

lemma (**in** *Module*) *mem-single-l-span*:⟦*ideal R A*; $h \in$ *carrier M*;
$s \in \{j.\ j \leq (n::nat)\} \to A$; $f \in \{j.\ j \leq n\} \to \{h\}$;
l-comb R M n s f $\in$ *linear-span R M A* $\{h\}$⟧ $\Longrightarrow$
$\exists\, a \in A.$ *l-comb R M n s f* $= a \cdot_s h$
⟨*proof*⟩

lemma (**in** *Module*) *mem-single-l-span1*:⟦*ideal R A*; $h \in$ *carrier M*;

$x \in$ *linear-span R M A* $\{h\}\rrbracket \Longrightarrow \exists\, a \in A.\ x = a \cdot_s h$
⟨*proof*⟩

lemma (**in** *Module*) *linear-span-subModule*:⟦*ideal R A*; $H \subseteq$ *carrier M*⟧ $\Longrightarrow$
submodule R M (*linear-span R M A H*)
⟨*proof*⟩

lemma (**in** *Module*) *l-comb-mem-submoduleTr*:⟦*ideal R A*; *submodule R M N*⟧ $\Longrightarrow$
$(s \in \{j.\ j \leq (n::nat)\} \to A \wedge f \in \{j.\ j \leq n\} \to$ *carrier M* $\wedge$
$(\forall\, j \leq n.(s\ j) \cdot_s (f\ j) \in N)) \longrightarrow$ *l-comb R M n s f* $\in N$
⟨*proof*⟩

lemma (**in** *Module*) *l-span-sub-submodule*:⟦*ideal R A*; *submodule R M N*; $H \subseteq N$⟧
$\Longrightarrow$
linear-span R M A H $\subseteq N$
⟨*proof*⟩

lemma (**in** *Module*) *linear-span-sub*:⟦*ideal R A*; $H \subseteq$ *carrier M*⟧ $\Longrightarrow$
(*linear-span R M A H*) $\subseteq$ *carrier M*
⟨*proof*⟩

definition
smodule-ideal-coeff :: [('*r*, '*m*) *Ring-scheme*, ('*a*, '*r*, '*m1*) *Module-scheme*,
'*r set*] ⇒ '*a set* **where**
smodule-ideal-coeff R M A = *linear-span R M A* (*carrier M*)

abbreviation
SMLIDEALCOEFF (‹(*3-*/ $\odot$_ -)› [*64*,*64*,*65*]*64*) **where**
$A \odot_R M$ == *smodule-ideal-coeff R M A*

lemma (**in** *Module*) *smodule-ideal-coeff-is-Submodule*:*ideal R A* $\Longrightarrow$
submodule R M ($A \odot_R M$)
⟨*proof*⟩

lemma (**in** *Module*) *mem-smodule-ideal-coeff*:⟦*ideal R A*; $x \in A \odot_R M$⟧ $\Longrightarrow$
$\exists\, n.\ \exists\, s \in \{j.\ j \leq n\} \to A.\ \exists\, g \in \{j.\ j \leq n\} \to$ *carrier M*.
x = *l-comb R M n s g*
⟨*proof*⟩

definition
quotient-of-submodules :: [('*r*, '*m*) *Ring-scheme*, ('*a*, '*r*, '*m1*) *Module-scheme*,
'*a set*, '*a set*] ⇒ '*r set* **where**
quotient-of-submodules R M N P = $\{x \mid x.\ x \in$*carrier R* $\wedge$
(*linear-span R M* (*Rxa R x*) *P*) $\subseteq N\}$

definition
Annihilator :: [('*r*, '*m*) *Ring-scheme*, ('*a*, '*r*, '*m1*) *Module-scheme*]
⇒ '*r set* (‹(*Ann*_ -)› [*82*,*83*]*82*) **where**
$Ann_R\ M$ = *quotient-of-submodules R M* $\{\mathbf{0}_M\}$ (*carrier M*)

abbreviation
QOFSUBMDS (‹(*4*- -‡- -)› [*82*,*82*,*82*,*83*]*82*) **where**
$N\ {}_{R\ddagger M}\ P == quotient\text{-}of\text{-}submodules\ R\ M\ N\ P$

lemma (**in** *Module*) *quotient-of-submodules-inc-0*:
$[\![submodule\ R\ M\ P;\ submodule\ R\ M\ Q]\!] \Longrightarrow \mathbf{0}_R \in (P\ {}_{R\ddagger M}\ Q)$
⟨*proof*⟩

lemma (**in** *Module*) *quotient-of-submodules-is-ideal*:
$[\![submodule\ R\ M\ P;\ submodule\ R\ M\ Q]\!] \Longrightarrow ideal\ R\ (P\ {}_{R\ddagger M}\ Q)$
⟨*proof*⟩

lemma (**in** *Module*) *Ann-is-ideal*:$ideal\ R\ (Ann_R\ M)$
⟨*proof*⟩

lemma (**in** *Module*) *linmap-im-of-lincombTr*:$[\![ideal\ R\ A;\ R\ module\ N;$
$f \in mHom\ R\ M\ N;\ H \subseteq carrier\ M]\!] \Longrightarrow$
$s \in \{j.\ j \leq (n::nat)\} \to A \wedge g \in \{j.\ j \leq n\} \to H \longrightarrow$
$f\ (l\text{-}comb\ R\ M\ n\ s\ g) = l\text{-}comb\ R\ N\ n\ s\ (cmp\ f\ g)$
⟨*proof*⟩

lemma (**in** *Module*) *linmap-im-lincomb*:$[\![ideal\ R\ A;\ R\ module\ N;\ f \in mHom\ R\ M\ N;$
$H \subseteq carrier\ M;\ s \in \{j.\ j \leq (n::nat)\} \to A;\ g \in \{j.\ j \leq n\} \to H\ ]\!] \Longrightarrow$
$f\ (l\text{-}comb\ R\ M\ n\ s\ g) = l\text{-}comb\ R\ N\ n\ s\ (cmp\ f\ g)$
⟨*proof*⟩

lemma (**in** *Module*) *linmap-im-linspan*:$[\![ideal\ R\ A;\ R\ module\ N;\ f \in mHom\ R\ M\ N;$
$H \subseteq carrier\ M;\ s \in \{j.\ j \leq (n::nat)\} \to A;\ g \in \{j.\ j \leq n\} \to H\ ]\!] \Longrightarrow$
$f\ (l\text{-}comb\ R\ M\ n\ s\ g) \in linear\text{-}span\ R\ N\ A\ (f\ `\ H)$
⟨*proof*⟩

lemma (**in** *Module*) *linmap-im-linspan1*:$[\![ideal\ R\ A;\ R\ module\ N;\ f \in mHom\ R\ M\ N;$
$H \subseteq carrier\ M;\ h \in linear\text{-}span\ R\ M\ A\ H]\!] \Longrightarrow$
$f\ h \in linear\text{-}span\ R\ N\ A\ (f\ `\ H)$
⟨*proof*⟩

definition
faithful :: [('*r*, '*m*) *Ring-scheme*, ('*a*, '*r*, '*m1*) *Module-scheme*]
$\Rightarrow$ *bool* **where**
$faithful\ R\ M \longleftrightarrow Ann_R\ M = \{\mathbf{0}_R\}$

5.3 nsum and Generators

definition
generator :: [('*r*, '*m*) *Ring-scheme*, ('*a*, '*r*, '*m1*) *Module-scheme*, '*a set*] ⇒ *bool* **where**
generator R M H == *H* ⊆ *carrier M* ∧ *linear-span R M* (*carrier R*) *H* = *carrier M*

definition
finite-generator :: [('*r*, '*m*) *Ring-scheme*, ('*a*, '*r*, '*m1*) *Module-scheme*, '*a set*] ⇒ *bool* **where**
finite-generator R M H ⟷ *finite H* ∧ *generator R M H*

definition
fGOver :: [('*a*, '*r*, '*m1*) *Module-scheme*, ('*r*, '*m*) *Ring-scheme*] ⇒ *bool* **where**
fGOver M R ⟷ (∃ *H*. *finite-generator R M H*)

abbreviation
FGENOVER (**infixl** ‹*fgover*› 70) **where**
M fgover R == *fGOver M R*

lemma (**in** *Module*) *h-in-linear-span*:⟦*H* ⊆ *carrier M*; *h* ∈ *H*⟧ ⟹ *h* ∈ *linear-span R M* (*carrier R*) *H*
⟨*proof*⟩

lemma (**in** *Module*) *generator-sub-carrier*:*generator R M H* ⟹ *H* ⊆ *carrier M*
⟨*proof*⟩

lemma (**in** *Module*) *lin-span-sub-carrier*:⟦*ideal R A*; *H* ⊆ *carrier M*⟧ ⟹ *linear-span R M A H* ⊆ *carrier M*
⟨*proof*⟩

lemma (**in** *Module*) *lin-span-coeff-mono*:⟦*ideal R A*; *H* ⊆ *carrier M*⟧⟹ *linear-span R M A H* ⊆ *linear-span R M* (*carrier R*) *H*
⟨*proof*⟩

lemma (**in** *Module*) *l-span-sum-closedTr*:⟦*ideal R A*; *H* ⊆ *carrier M*⟧⟹
∀ *s*. ∀ *f*. *s*∈{*j*. *j* ≤ (*n*::*nat*)} → *A* ∧
f ∈ {*j*. *j* ≤ *n*} → *linear-span R M A H* ⟶
(*nsum M* (λ*j*. *s j* ·$_s$ (*f j*)) *n* ∈ *linear-span R M A H*)
⟨*proof*⟩

lemma (**in** *Module*) *l-span-closed*:⟦*ideal R A*; *H* ⊆ *carrier M*;
s ∈ {*j*. *j* ≤ (*n*::*nat*)} → *A*; *f* ∈ {*j*. *j* ≤ *n*} → *linear-span R M A H* ⟧ ⟹
l-comb R M n s f ∈ *linear-span R M A H*
⟨*proof*⟩

lemma (**in** *Module*) *l-span-closed1*:⟦$H \subseteq$ *carrier* M;
$s \in \{j.\ j \leq (n::nat)\} \to$ *carrier* R;
$f \in \{j.\ j \leq n\} \to$ *linear-span* $R\ M$ (*carrier* R) H ⟧ ⟹
$\Sigma_e\ M\ (\lambda j.\ \ s\ j \cdot_s (f\ j))\ n \in$ *linear-span* $R\ M$ (*carrier* R) H
⟨*proof*⟩

lemma (**in** *Module*) *l-span-closed2Tr0*:⟦*ideal* $R\ A$; $H \subseteq$ *carrier* M; *Ring* R; $s \in A$;
$f \in$ *linear-span* $R\ M$ (*carrier* R) H ⟧ ⟹ $s \cdot_s f \in$ *linear-span* $R\ M\ A\ H$
⟨*proof*⟩

lemma (**in** *Module*) *l-span-closed2Tr*:⟦*ideal* $R\ A$; $H \subseteq$ *carrier* M⟧ ⟹
$s \in \{j.\ j \leq (n::nat)\} \to A \wedge$
$f \in \{j.\ j \leq n\} \to$ *linear-span* $R\ M$ (*carrier* R) $H \longrightarrow$
l-comb $R\ M\ n\ s\ f \in$ *linear-span* $R\ M\ A\ H$
⟨*proof*⟩

lemma (**in** *Module*) *l-span-closed2*:⟦*ideal* $R\ A$; $H \subseteq$ *carrier* M;
$s \in \{j.\ j \leq (n::nat)\} \to A$;
$f \in \{j.\ j \leq n\} \to$ *linear-span* $R\ M$ (*carrier* R) H⟧ ⟹
l-comb $R\ M\ n\ s\ f \in$ *linear-span* $R\ M\ A\ H$
⟨*proof*⟩

lemma (**in** *Module*) *l-span-l-span*:$H \subseteq$ *carrier* M ⟹
linear-span $R\ M$ (*carrier* R) (*linear-span* $R\ M$ (*carrier* R) H) =
linear-span $R\ M$ (*carrier* R) H
⟨*proof*⟩

lemma (**in** *Module*) *l-spanA-l-span*:⟦*ideal* $R\ A$; $H \subseteq$ *carrier* M⟧ ⟹
linear-span $R\ M\ A$ (*linear-span* $R\ M$ (*carrier* R) H) =
linear-span $R\ M\ A\ H$
⟨*proof*⟩

lemma (**in** *Module*) *l-span-zero*:*ideal* $R\ A$ ⟹ *linear-span* $R\ M\ A\ \{\mathbf{0}\} = \{\mathbf{0}\}$
⟨*proof*⟩

lemma (**in** *Module*) *l-span-closed3*:⟦*ideal* $R\ A$; *generator* $R\ M\ H$;
$A \odot_R M =$ *carrier* M⟧ ⟹ *linear-span* $R\ M\ A\ H =$ *carrier* M
⟨*proof*⟩

lemma (**in** *Module*) *generator-generator*:⟦*generator* $R\ M\ H$; *H1* $\subseteq$ *carrier* M;
$H \subseteq$ *linear-span* $R\ M$ (*carrier* R) *H1*⟧ ⟹ *generator* $R\ M$ *H1*
⟨*proof*⟩

lemma (**in** *Module*) *generator-elimTr*:
$f \in \{j.\ j \leq (n::nat)\} \to$ *carrier* $M \wedge$ *generator* $R\ M$ (f ‘ $\{j.\ j \leq n\}$) $\wedge$
($\forall\ i \in$*nset* (*Suc 0*) $n.\ f\ i \in$
linear-span $R\ M$ (*carrier* R) (f ‘ $\{j.\ j \leq (i - Suc\ 0)\}$)) ⟶
linear-span $R\ M$ (*carrier* R) $\{f\ 0\} =$ *carrier* M
⟨*proof*⟩

lemma (**in** *Module*) *generator-generator-elim*:
$[\![f \in \{j.\ j \leq (n::nat)\} \to carrier\ M;\ generator\ R\ M\ (f\ `\ \{j.\ j \leq n\});$
$(\forall i{\in}nset\ (Suc\ 0)\ n.\ f\ i \in linear\text{-}span\ R\ M\ (carrier\ R)$
$(f\ `\ \{j.\ j \leq (i - Suc\ 0)\}))]\!] \Longrightarrow$
$linear\text{-}span\ R\ M\ (carrier\ R)\ \{f\ 0\} = carrier\ M$
⟨*proof*⟩

lemma (**in** *Module*) *surjec-generator*:$[\![R\ module\ N;\ f \in mHom\ R\ M\ N;$
$surjec_{M,N}\ f;\ generator\ R\ M\ H]\!] \Longrightarrow generator\ R\ N\ (f\ `\ H)$
⟨*proof*⟩

lemma (**in** *Module*) *surjec-finitely-gen*:$[\![R\ module\ N;\ f \in mHom\ R\ M\ N;$
$surjec_{M,N}\ f;\ M\ fgover\ R]\!] \Longrightarrow N\ fgover\ R$
⟨*proof*⟩

5.3.1 Sum up coefficients

Symbolic calculation.

lemma (**in** *Module*) *similar-termTr*:$[\![ideal\ R\ A;\ a \in A]\!] \Longrightarrow$
$\forall s.\ \forall f.\ s \in \{j.\ j \leq (n::nat)\} \to A \wedge$
$f \in \{j.\ j \leq n\} \to carrier\ M \wedge$
$m \in f\ `\ \{j.\ j \leq n\} \longrightarrow$
$(\exists t{\in}\{j.\ j \leq n\} \to A.\ nsum\ M\ (\lambda j.\ s\ j \cdot_s (f\ j))\ n \pm a \cdot_s m =$
$nsum\ M\ (\lambda j.\ t\ j \cdot_s (f\ j))\ n\ )$
⟨*proof*⟩

lemma (**in** *Module*) *similar-term1*:$[\![ideal\ R\ A;\ a \in A;\ s \in \{j.\ j{\leq}(n::nat)\} \to A;$
$f \in \{j.\ j \leq n\} \to carrier\ M;\ m \in f\ `\ \{j.\ j \leq n\}]\!] \Longrightarrow$
$\exists t{\in}\{j.\ j \leq n\} \to A.\ \Sigma_e\ M\ (\lambda j.\ s\ j \cdot_s (f\ j))\ n \pm a \cdot_s m =$
$\Sigma_e\ M\ (\lambda j.\ \ t\ j \cdot_s (f\ j))\ n$
⟨*proof*⟩

lemma (**in** *Module*) *same-togetherTr*:$[\![ideal\ R\ A;\ H \subseteq carrier\ M\ ]\!] \Longrightarrow$
$\forall s.\ \forall f.\ s{\in}\{j.\ j \leq (n::nat)\} \to A\ \wedge f \in \{j.\ j \leq n\} \to H \longrightarrow$
$(\exists t \in \{j.\ j \leq (card\ (f\ `\ \{j.\ j \leq n\}) - Suc\ 0)\} \to A.$
$\exists g \in \{j.\ j \leq (card\ (f\ `\ \{j.\ j \leq n\}) - Suc\ 0)\} \to f\ `\ \{j.\ j \leq n\}.$
$surj\text{-}to\ g\ \{j.\ j \leq (card\ (f\ `\ \{j.\ j \leq n\}) - Suc\ 0)\}\ (f\ `\ \{j.\ j \leq n\}) \wedge$
$nsum\ M\ (\lambda j.\ s\ j \cdot_s (f\ j))\ n = nsum\ M\ (\lambda k.\ t\ k \cdot_s (g\ k))$
$(card\ (f\ `\ \{j.\ j \leq n\}) - Suc\ 0))$
⟨*proof*⟩

lemma (**in** *Module*) *same-together*:$[\![ideal\ R\ A;\ H \subseteq carrier\ M;$
$s \in \{j.\ j \leq (n::nat)\} \to A;\ f \in \{j.\ j \leq n\} \to H]\!] \Longrightarrow$
$\exists t \in \{j.\ j \leq (card\ (f\ `\ \{j.\ j \leq (n::nat)\}) - Suc\ 0)\} \to A.$
$\exists g \in \{j.\ j \leq (card\ (f\ `\ \{j.\ j \leq n\}) - Suc\ 0)\} \to f\ `\ \{j.\ j \leq n\}.$
$surj\text{-}to\ g\ \{j.\ j \leq (card\ (f\ `\ \{j.\ j \leq n\}) - Suc\ 0)\}\ (f\ `\ \{j.\ j \leq n\}) \wedge$

$\Sigma_e\ M\ (\lambda j.\ s\ j\ \cdot_s\ (f\ j))\ n =$
$\Sigma_e\ M\ (\lambda k.\ t\ k\ \cdot_s\ (g\ k))\ (card\ (f\ `\ \{j.\ j \leq n\})\ -\ Suc\ 0)$
⟨*proof*⟩

lemma (**in** *Module*) *one-last*:⟦*ideal R A*; $H \subseteq$ *carrier M*;
$s \in \{j.\ j \leq (Suc\ n)\} \to A$; $f \in \{j.\ j \leq (Suc\ n)\} \to H$;
bij-to f $\{j.\ j \leq (Suc\ n)\}\ H$; $j \leq (Suc\ n)$; $j \neq (Suc\ n)$⟧ ⟹
$\exists\, t \in \{j.\ j \leq (Suc\ n)\} \to A.\ \exists\, g \in \{j.\ j \leq (Suc\ n)\} \to H.$
$\Sigma_e\ M\ (\lambda k.\ s\ k\ \cdot_s\ (f\ k))\ (Suc\ n) = \Sigma_e\ M\ (\lambda k.\ t\ k\ \cdot_s\ (g\ k))\ (Suc\ n) \wedge$
$g\ (Suc\ n) = f\ j \wedge t\ (Suc\ n) = s\ j \wedge$ *bij-to* $g\ \{j.\ j \leq (Suc\ n)\}\ H$
⟨*proof*⟩

lemma (**in** *Module*) *finite-lin-spanTr1*:⟦*ideal R A*; $z \in$ *carrier M*⟧ ⟹
$h \in \{j.\ j \leq (n::nat)\} \to \{z\} \wedge t \in \{j.\ j \leq n\} \to A\ \longrightarrow$
$(\exists\, s \in \{0::nat\} \to A.\ \Sigma_e\ M\ (\lambda j.\ t\ j\ \cdot_s\ (h\ j))\ n = s\ 0\ \cdot_s\ z)$
⟨*proof*⟩

lemma (**in** *Module*) *single-span*:⟦*ideal R A*; $z \in$ *carrier M*;
$h \in \{j.\ j \leq (n::nat)\} \to \{z\}$; $t \in \{j.\ j \leq n\} \to A$⟧ ⟹
$\exists\, s \in \{0::nat\} \to A.\ \Sigma_e\ M\ (\lambda j.\ t\ j\ \cdot_s\ (h\ j))\ n = s\ 0\ \cdot_s\ z$
⟨*proof*⟩

definition
coeff-at-k :: [('*r*, '*m*) *Ring-scheme*, '*r*, *nat*] ⇒ (*nat* ⇒ '*r*) **where**
coeff-at-k R a k = $(\lambda j.$ *if* $j = k$ *then a else* $(\mathbf{0}_R))$

lemma *card-Nset-im*:$f \in \{j.\ j \leq (n::nat)\} \to A$ ⟹
$(Suc\ 0) \leq card\ (f\ `\{j.\ j \leq n\})$
⟨*proof*⟩

lemma (**in** *Module*) *eSum-changeTr1*:⟦*ideal R A*;
$t \in \{k.\ k \leq (card\ (f\ `\ \{j.\ j \leq (n1::nat)\})\ -\ Suc\ 0)\} \to A$;
$g \in \{k.\ k \leq (card\ (f\ `\ \{j.\ j \leq n1\})\ -\ Suc\ 0)\} \to f\ `\{j.\ j \leq n1\}$;
$Suc\ 0 < card\ (f\ `\{j.\ j \leq n1\})$; $g\ x = h\ (Suc\ n)$; $x = Suc\ n$;
$card\ (f\ `\{j.\ j \leq n1\})\ -\ Suc\ 0 = Suc\ (card\ (f\ `\ \{j.\ j \leq n1\})\ -\ Suc\ 0\ -\ Suc\ 0)$⟧
⟹
$\Sigma_e\ M\ (\lambda k.\ t\ k\ \cdot_s\ (g\ k))\ (card\ (f\ `\ \{j.\ j \leq n1\})\ -\ Suc\ 0) =$
$\Sigma_e\ M\ (\lambda k.\ t\ k\ \cdot_s\ (g\ k))\ (card\ (f\ `\ \{j.\ j \leq n1\})\ -\ Suc\ 0\ -\ Suc\ 0) \pm$
$(t\ (Suc\ (card\ (f\ `\ \{j.\ j \leq n1\})\ -\ Suc\ 0\ -\ Suc\ 0))\ \cdot_s$
$(g\ (\ Suc\ (card\ (f\ `\ \{j.\ j \leq n1\})\ -\ Suc\ 0\ -\ Suc\ 0))))$
⟨*proof*⟩

definition
zeroi :: [('*r*, '*m*) *Ring-scheme*] ⇒ *nat* ⇒ '*r* **where**
zeroi R = $(\lambda j.\ \mathbf{0}_R)$

lemma *zeroi-func*:⟦*Ring R*; *ideal R A*⟧ ⟹ *zeroi R* $\in \{j.\ j \leq 0\} \to A$
⟨*proof*⟩

lemma (**in** *Module*) *prep-arrTr1*:⟦*ideal R A*; $h \in \{j.\ j \leq (Suc\ n)\} \to carrier\ M$;
$f \in \{j.\ j \leq (n1::nat)\} \to h\ `\ \{j.\ j \leq (Suc\ n)\}$; $s \in \{j.\ j \leq n1\} \to A$;
$m = l\text{-}comb\ R\ M\ n1\ s\ f$⟧ ⟹
$\exists\, l \in \{j.\ j \leq (Suc\ n)\}.\ (\exists\, s \in \{j.\ j \leq (l::nat)\} \to A.$
$\exists\, g \in \{j.\ j \leq l\} \to h\ `\{j.\ j \leq (Suc\ n)\}.\ m = l\text{-}comb\ R\ M\ l\ s\ g \wedge$
$bij\text{-}to\ g\ \{j.\ j \leq l\}\ (f\ `\ \{j.\ j \leq n1\}))$
⟨*proof*⟩

lemma *two-func-imageTr*:⟦ $h \in \{j.\ j \leq Suc\ n\} \to B$;
$f \in \{j.\ j \leq (m::nat)\} \to h\ `\ \{j.\ j \leq Suc\ n\}$; $h\ (Suc\ n) \notin f\ `\ \{j.\ j \leq m\}$⟧
⟹ $f \in \{j.\ j \leq m\} \to h\ `\ \{j.\ j \leq n\}$
⟨*proof*⟩

lemma (**in** *Module*) *finite-lin-spanTr3-0*:⟦*bij-to g* $\{j.\ j \leq l\}$ $(g\ `\{j.\ j \leq l\})$;
ideal R A;
$\forall\, na.\ \forall\, s \in \{j.\ j \leq na\} \to A.$
$\forall\, f \in \{j.\ j \leq na\} \to h\ `\ \{j.\ j \leq n\}.$
$\exists\, t \in \{j.\ j \leq n\} \to A.\ l\text{-}comb\ R\ M\ na\ s\ f = l\text{-}comb\ R\ M\ n\ t\ h$;
$h \in \{j.\ j \leq Suc\ n\} \to carrier\ M$; $s \in \{j.\ j \leq m\} \to A$;
$f \in \{j.\ j \leq m\} \to h\ `\ \{j.\ j \leq Suc\ n\}$;
$l \leq Suc\ n$; $sa \in \{j.\ j \leq l\} \to A$; $g \in \{j.\ j \leq l\} \to h\ `\ \{j.\ j \leq Suc\ n\}$;
$0 < l$; $f\ `\ \{j.\ j \leq m\} = g\ `\ \{j.\ j \leq l\}$; $h\ (Suc\ n) = g\ l$⟧
⟹ $\exists\, t \in \{j.\ j \leq Suc\ n\} \to A.\ l\text{-}comb\ R\ M\ l\ sa\ g = l\text{-}comb\ R\ M\ (Suc\ n)\ t\ h$
⟨*proof*⟩

lemma (**in** *Module*) *finite-lin-spanTr3*:*ideal R A* ⟹
$h \in \{j.\ j \leq (n::nat)\} \to carrier\ M \longrightarrow$
$(\forall\, na.\ \forall\, s \in \{j.\ j \leq (na::nat)\} \to A.$
$\forall\, f \in \{j.\ j \leq na\} \to (h\ `\ \{j.\ j \leq n\}).\ (\exists\, t \in \{j.\ j \leq n\} \to A.$
$l\text{-}comb\ R\ M\ na\ s\ f = l\text{-}comb\ R\ M\ n\ t\ h))$
⟨*proof*⟩

lemma (**in** *Module*) *finite-lin-span*:
⟦*ideal R A*; $h \in \{j.\ j \leq (n::nat)\} \to carrier\ M$; $s \in \{j.\ j \leq (n1::nat)\} \to A$;
$f \in \{j.\ j \leq n1\} \to h\ `\ \{j.\ j \leq n\}$⟧ ⟹ $\exists\, t \in \{j.\ j \leq n\} \to A.$
$l\text{-}comb\ R\ M\ n1\ s\ f = l\text{-}comb\ R\ M\ n\ t\ h$
⟨*proof*⟩

5.3.2 Free generators

definition
free-generator :: [('*r*, '*m*) *Ring-scheme*, ('*a*, '*r*, '*m1*) *Module-scheme*, '*a set*]
⇒ *bool* **where**
free-generator R M H ⟷ *generator R M H* ∧
$(\forall\, n.\ (\forall\, s\ f.\ (s \in \{j.\ j \leq (n::nat)\} \to carrier\ R \wedge$
$f \in \{j.\ j \leq n\} \to H \wedge inj\text{-}on\ f\ \{j.\ j \leq n\} \wedge$
$l\text{-}comb\ R\ M\ n\ s\ f = \mathbf{0}_M) \longrightarrow s \in \{j.\ j \leq n\} \to \{\mathbf{0}_R\}))$

lemma (**in** *Module*) *free-generator-generator*:*free-generator R M H* $\Longrightarrow$
generator R M H
⟨*proof*⟩

lemma (**in** *Module*) *free-generator-sub*:*free-generator R M H* $\Longrightarrow$
$H \subseteq$ *carrier M*
⟨*proof*⟩

lemma (**in** *Module*) *free-generator-nonzero*:$[\![\neg$ (*zeroring R*);
free-generator R M H; $h \in H]\!] \Longrightarrow h \neq \mathbf{0}$
⟨*proof*⟩

lemma (**in** *Module*) *has-free-generator-nonzeroring*: $[\![$*free-generator R M H*;
$\exists p \in$ *linear-span R M* (*carrier R*) *H*. $p \neq \mathbf{0}\ ]\!] \Longrightarrow \neg$ *zeroring R*
⟨*proof*⟩

lemma (**in** *Module*) *unique-expression1*:$[\![H \subseteq$ *carrier M*; *free-generator R M H*;
$s \in \{j.\ j \leq (n::nat)\} \to$ *carrier R*; $m \in \{j.\ j \leq n\} \to H$;
inj-on m $\{j.\ j \leq n\}$; *l-comb R M n s m* $= \mathbf{0}]\!] \Longrightarrow$
$\forall j \in \{j.\ j \leq n\}.\ s\ j = \mathbf{0}_R$
⟨*proof*⟩

lemma (**in** *Module*) *free-gen-coeff-zero*:$[\![H \subseteq$ *carrier M*; *free-generator R M H*;
$h \in H$; $a \in$ *carrier R*; $a \cdot_s h = \mathbf{0}]\!] \Longrightarrow a = \mathbf{0}_R$
⟨*proof*⟩

lemma (**in** *Module*) *unique-expression2*:$[\![H \subseteq$ *carrier M*;
$f \in \{j.\ j \leq (n::nat)\} \to H$; $s \in \{j.\ j \leq n\} \to$ *carrier R*$]\!] \Longrightarrow$
$\exists m\ g\ t.\ g \in (\{j.\ j \leq (m::nat)\} \to H) \wedge$
bij-to g $\{j.\ j \leq (m::nat)\}$ $(f\ `\ \{j.\ j \leq n\}) \wedge$
$t \in \{j.\ j \leq m\} \to$ *carrier R* $\wedge$
l-comb R M n s f = *l-comb R M m t g*
⟨*proof*⟩

lemma (**in** *Module*) *unique-expression3-1*:$[\![H \subseteq$ *carrier M*;
$f \in \{l.\ l \leq (Suc\ n)\} \to H$; $s \in \{l.\ l \leq (Suc\ n)\} \to$ *carrier R*;
$(f\ (Suc\ n)) \notin f\ `(\{l.\ l \leq (Suc\ n)\} - \{Suc\ n\})]\!] \Longrightarrow$
$\exists g\ m\ t.\ g \in \{l.\ l \leq (m::nat)\} \to H \wedge$
inj-on g $\{l.\ l \leq (m::nat)\} \wedge$
$t \in \{l.\ l \leq (m::nat)\} \to$ *carrier R* $\wedge$
l-comb R M (*Suc n*) *s f* =
l-comb R M m t g $\wedge$ $t\ m = s\ (Suc\ n) \wedge g\ m = f\ (Suc\ n)$
⟨*proof*⟩

lemma (**in** *Module*) *unique-expression3-2*:$[\![H \subseteq$ *carrier M*;
$f \in \{k.\ k \leq (Suc\ n)\} \to H$; $s \in \{k.\ k \leq (Suc\ n)\} \to$ *carrier R*;
$l \leq (Suc\ n)$; $(f\ l) \notin f\ `(\{k.\ k \leq (Suc\ n)\} - \{l\})$; $l \neq Suc\ n]\!] \Longrightarrow$
$\exists g\ m\ t.\ g \in \{l.\ l \leq (m::nat)\} \to H \wedge$ *inj-on g* $\{l.\ l \leq (m::nat)\} \wedge$

$t \in \{l.\ l \leq m\} \to \mathit{carrier}\ R \wedge$
$\mathit{l\text{-}comb}\ R\ M\ (\mathit{Suc}\ n)\ s\ f = \mathit{l\text{-}comb}\ R\ M\ m\ t\ g \wedge$
$t\ m = s\ l \wedge g\ m = f\ l$
⟨*proof*⟩

lemma (**in** *Module*) *unique-expression3*:
$[\![H \subseteq \mathit{carrier}\ M;\ f \in \{k.\ k \leq (\mathit{Suc}\ n)\} \to H;$
$s \in \{k.\ k \leq (\mathit{Suc}\ n)\} \to \mathit{carrier}\ R;\ l \leq (\mathit{Suc}\ n);$
$(f\ l) \notin f\ `\ (\{k.\ k \leq (\mathit{Suc}\ n)\} - \{l\})]\!] \Longrightarrow$
$\exists g\ m\ t.\ g \in \{k.\ k \leq (m::\mathit{nat})\} \to H \wedge$
$\mathit{inj\text{-}on}\ g\ \{k.\ k \leq m\} \wedge$
$t \in \{k.\ k \leq m\} \to \mathit{carrier}\ R \wedge$
$\mathit{l\text{-}comb}\ R\ M\ (\mathit{Suc}\ n)\ s\ f = \mathit{l\text{-}comb}\ R\ M\ m\ t\ g \wedge t\ m = s\ l \wedge g\ m = f\ l$
⟨*proof*⟩

lemma (**in** *Module*) *unique-expression4*:*free-generator R M H* $\Longrightarrow$
$f \in \{k.\ k \leq (n::\mathit{nat})\} \to H \wedge \mathit{inj\text{-}on}\ f\ \{k.\ k \leq n\} \wedge$
$s \in \{k.\ k \leq n\} \to \mathit{carrier}\ R \wedge \mathit{l\text{-}comb}\ R\ M\ n\ s\ f \neq \mathbf{0}\ \longrightarrow$
$(\exists m\ g\ t.\ (g \in \{k.\ k \leq m\} \to H) \wedge \mathit{inj\text{-}on}\ g\ \{k.\ k \leq m\} \wedge$
$(g\ `\ \{k.\ k \leq m\} \subseteq f\ `\ \{k.\ k \leq n\}) \wedge (t \in \{k.\ k \leq m\} \to \mathit{carrier}\ R) \wedge$
$(\forall l \in \{k.\ k \leq m\}.\ t\ l \neq \mathbf{0}_R) \wedge \mathit{l\text{-}comb}\ R\ M\ n\ s\ f = \mathit{l\text{-}comb}\ R\ M\ m\ t\ g)$
⟨*proof*⟩

lemma (**in** *Module*) *unique-prepression5-0*:[![*free-generator R M H*;
$f \in \{j.\ j \leq n\} \to H;\ \mathit{inj\text{-}on}\ f\ \{j.\ j \leq n\};$
$s \in \{j.\ j \leq n\} \to \mathit{carrier}\ R;\ g \in \{j.\ j \leq m\} \to H;$
$\mathit{inj\text{-}on}\ g\ \{j.\ j \leq m\};\ t \in \{j.\ j \leq m\} \to \mathit{carrier}\ R;$
$\mathit{l\text{-}comb}\ R\ M\ n\ s\ f = \mathit{l\text{-}comb}\ R\ M\ m\ t\ g; \forall j{\leq}n.\ s\ j \neq \mathbf{0}_R;\ \forall k{\leq}m.\ t\ k \neq \mathbf{0}_R;$
$f\ n \notin g\ `\ \{j.\ j \leq m\};\ 0 < n]\!] \Longrightarrow \mathit{False}$
⟨*proof*⟩

lemma (**in** *Module*) *unique-expression5*:[![*free-generator R M H*;
$f \in \{j.\ j \leq (n::\mathit{nat})\} \to H;\ \mathit{inj\text{-}on}\ f\ \{j.\ j \leq n\};$
$s \in \{j.\ j \leq n\} \to \mathit{carrier}\ R;\ g \in \{j.\ j \leq (m::\mathit{nat})\} \to H;$
$\mathit{inj\text{-}on}\ g\ \{j.\ j \leq m\};\ t \in \{j.\ j \leq m\} \to \mathit{carrier}\ R;$
$\mathit{l\text{-}comb}\ R\ M\ n\ s\ f = \mathit{l\text{-}comb}\ R\ M\ m\ t\ g;$
$\forall j \in \{j.\ j \leq n\}.\ s\ j \neq \mathbf{0}_R;\ \forall k \in \{j.\ j \leq m\}.\ t\ k \neq \mathbf{0}_R]\!] \Longrightarrow$
$f\ `\ \{j.\ j \leq n\} \subseteq g\ `\ \{j.\ j \leq m\}$
⟨*proof*⟩

lemma (**in** *Module*) *unique-expression6*:[![*free-generator R M H*;
$f \in \{j.\ j \leq (n::\mathit{nat})\} \to H;\ \mathit{inj\text{-}on}\ f\ \{j.\ j \leq n\};$
$s \in \{j.\ j \leq n\} \to \mathit{carrier}\ R;$
$g \in \{j.\ j \leq (m::\mathit{nat})\} \to H;\ \mathit{inj\text{-}on}\ g\ \{j.\ j \leq m\};$
$t \in \{j.\ j \leq m\} \to \mathit{carrier}\ R;$
$\mathit{l\text{-}comb}\ R\ M\ n\ s\ f = \mathit{l\text{-}comb}\ R\ M\ m\ t\ g;$
$\forall j{\in}\{j.\ j \leq n\}.\ s\ j \neq \mathbf{0}_R;\ \forall k{\in}\ \{j.\ j \leq m\}.\ t\ k \neq \mathbf{0}_R]\!] \Longrightarrow$

f ‘$\{j.\ j \leq n\} = g$ ‘ $\{j.\ j \leq m\}$
⟨*proof*⟩

lemma (**in** *Module*) *unique-expression7-1*:⟦*free-generator R M H*;
$f \in \{j.\ j \leq (n::nat)\} \to H$; *inj-on f* $\{j.\ j \leq n\}$;
$s \in \{j.\ j \leq n\} \to$ *carrier R*;
$g \in \{j.\ j \leq (m::nat)\} \to H$; *inj-on g* $\{j.\ j \leq m\}$;
$t \in \{j.\ j \leq m\} \to$ *carrier R*;
l-comb R M n s f = *l-comb R M m t g*;
$\forall j \in \{j.\ j \leq n\}.\ s\ j \neq \mathbf{0}_R$; $\forall k \in \{j.\ j \leq m\}.\ t\ k \neq \mathbf{0}_R$⟧ $\Longrightarrow n = m$
⟨*proof*⟩

lemma (**in** *Module*) *unique-expression7-2*:⟦*free-generator R M H*;
$f \in \{j.\ j \leq (n::nat)\} \to H$; *inj-on f* $\{j.\ j \leq n\}$;
$s \in \{j.\ j \leq n\} \to$ *carrier R*; $t \in \{j.\ j \leq n\} \to$ *carrier R*;
l-comb R M n s f = *l-comb R M n t f*⟧ $\Longrightarrow (\forall l \in \{j.\ j \leq n\}.\ s\ l = t\ l)$
⟨*proof*⟩

end

theory *Algebra8* **imports** *Algebra7* **begin**

5.4 nsum and Generators (continued)

lemma (**in** *Module*) *unique-expression-last*: ⟦*free-generator R M H*;
$f \in \{j.\ j \leq Suc\ n\} \to H$; $s \in \{j.\ j \leq Suc\ n\} \to$ *carrier R*;
$g \in \{j.\ j \leq Suc\ n\} \to H$; $t \in \{j.\ j \leq Suc\ n\} \to$ *carrier R*;
l-comb R M (*Suc n*) *s f* = *l-comb R M* (*Suc n*) *t g*;
inj-on f $\{j.\ j \leq Suc\ n\}$; *inj-on g* $\{j.\ j \leq Suc\ n\}$;
f (*Suc n*) = *g* (*Suc n*)⟧ $\Longrightarrow$ *s* (*Suc n*) = *t* (*Suc n*)
⟨*proof*⟩

lemma (**in** *Module*) *unique-exprTr7p1*: ⟦*free-generator R M H*;
$\forall f\ s\ g\ t\ m.$
$f \in \{j.\ j \leq n\} \to H \wedge$ *inj-on f* $\{j.\ j \leq n\} \wedge s \in \{j.\ j \leq n\} \to$ *carrier R* $\wedge$
$g \in \{j.\ j \leq m\} \to H \wedge$ *inj-on g* $\{j.\ j \leq m\} \wedge t \in \{j.\ j \leq m\} \to$ *carrier R* $\wedge$
l-comb R M n s f = *l-comb R M m t g* $\wedge$
$(\forall j \leq n.\ s\ j \neq \mathbf{0}_R) \wedge (\forall k \leq m.\ t\ k \neq \mathbf{0}_R) \longrightarrow$
$n = m \wedge$
$(\exists h.\ h \in \{j.\ j \leq n\} \to \{j.\ j \leq n\} \wedge$
$(\forall l \leq n.\ cmp\ f\ h\ l = g\ l \wedge cmp\ s\ h\ l = t\ l))$;
$f \in \{j.\ j \leq Suc\ n\} \to H$; $s \in \{j.\ j \leq Suc\ n\} \to$ *carrier R*;
$g \in \{j.\ j \leq Suc\ n\} \to H$; $t \in \{j.\ j \leq Suc\ n\} \to$ *carrier R*;
l-comb R M (*Suc n*) *s f* = *l-comb R M* (*Suc n*) *t g*; $\forall j \leq Suc\ n.\ s\ j \neq \mathbf{0}_R$;
$\forall k \leq Suc\ n.\ t\ k \neq \mathbf{0}_R$; *inj-on f* $\{j.\ j \leq Suc\ n\}$; *inj-on g* $\{j.\ j \leq Suc\ n\}$;
f (*Suc n*) = *g* (*Suc n*)⟧ $\Longrightarrow \exists h.\ h \in \{j.\ j \leq Suc\ n\} \to \{j.\ j \leq Suc\ n\} \wedge$
$(\forall l \leq Suc\ n.\ cmp\ f\ h\ l = g\ l \wedge cmp\ s\ h\ l = t\ l)$
⟨*proof*⟩

lemma (**in** *Module*) *unique-expression7*:*free-generator R M H* $\Longrightarrow$
$\forall f\ s\ g\ t\ m.\ f \in \{j.\ j \leq (n::nat)\} \to H \wedge$ *inj-on f* $\{j.\ j \leq n\} \wedge$
$s \in \{j.\ j \leq n\} \to$ *carrier R* $\wedge$
$g \in \{j.\ j \leq (m::nat)\} \to H \wedge$ *inj-on g* $\{j.\ j \leq m\} \wedge$
$t \in \{j.\ j \leq m\} \to$ *carrier R* $\wedge$ *l-comb R M n s f* $=$ *l-comb R M m t g* $\wedge$
$(\forall j \in \{j.\ j \leq n\}.\ s\ j \neq \mathbf{0}_R) \wedge (\forall k \in \{j.\ j \leq m\}.\ t\ k \neq \mathbf{0}_R\ ) \longrightarrow n = m \wedge$
$(\exists h.\ h \in \{j.\ j \leq n\} \to \{j.\ j \leq n\} \wedge (\forall l \in \{j.\ j \leq n\}.\ (cmp\ f\ h)\ l = g\ l$
$\wedge\ (cmp\ s\ h)\ l = t\ l))$
⟨*proof*⟩

lemma (**in** *Module*) *gen-mHom-eq*:⟦*R module N*; *generator R M H*; $f \in$ *mHom R M N*;
$g \in$ *mHom R M N*; $\forall h{\in}H.\ f\ h = g\ h\ ⟧ \Longrightarrow f = g$
⟨*proof*⟩

5.5 Existence of homomorphism

definition
fgs :: [('*r*, '*m*) *Ring-scheme*, ('*a*, '*r*, '*m1*) *Module-scheme*, '*a set*] $\Rightarrow$
'*a set* **where**

fgs R M A = *linear-span R M* (*carrier R*) *A*

definition
fsp :: [('*r*, '*m*) *Ring-scheme*, ('*a*, '*r*, '*m1*) *Module-scheme*,
('*b*, '*r*, '*m2*) *Module-scheme*, '*a* $\Rightarrow$ '*b*, '*a set*, '*a set*, '*a* $\Rightarrow$ '*b*] $\Rightarrow$ *bool* **where**
fsp R M N f H A g $\longleftrightarrow$ *g* $\in$ *mHom R* (*mdl M* (*fgs R M A*)) *N* $\wedge$ ($\forall z{\in}A.\ f\ z = g\ z$) $\wedge$ *A* $\subseteq$ *H*

definition
fsps :: [('*r*, '*m*) *Ring-scheme*, ('*a*, '*r*, '*m1*) *Module-scheme*,
('*b*, '*r*, '*m2*) *Module-scheme*, '*a* $\Rightarrow$ '*b*, '*a set*] $\Rightarrow$
(('*a set*) $*$ ('*a* $\Rightarrow$ '*b*)) *set* **where**
fsps R M N f H = {*Z*. $\exists A\ g.\ Z = (A, g) \wedge$ *fsp R M N f H A g*}

definition
od-fm-fun :: [('*r*, '*m*) *Ring-scheme*, ('*a*, '*r*, '*m1*) *Module-scheme*,
('*b*, '*r*, '*m2*) *Module-scheme*, '*a* $\Rightarrow$ '*b*, '*a set*] $\Rightarrow$
(('*a set*) $*$ ('*a* $\Rightarrow$ '*b*)) *Order* **where**
od-fm-fun R M N f H = ⦇*carrier* = *fsps R M N f H*,
rel = {*Y*. *Y* $\in$ (*fsps R M N f H*) $\times$ (*fsps R M N f H*) $\wedge$
(*fst* (*fst Y*)) $\subseteq$ (*fst* (*snd Y*))} ⦈

lemma (**in** *Module*) *od-fm-fun-carrier*:*carrier* (*od-fm-fun R M N f H*) =
fsps R M N f H

⟨*proof*⟩

lemma (**in** *Module*) *fgs-submodule*:$a \subseteq$ *carrier* $M \Longrightarrow$
submodule R M (*fgs* R M a)
⟨*proof*⟩

lemma (**in** *Module*) *fgs-sub-carrier*:$a \subseteq$ *carrier* $M \Longrightarrow$ (*fgs* R M a) $\subseteq$ *carrier* M
⟨*proof*⟩

lemma (**in** *Module*) *elem-fgs*:$[\![a \subseteq$ *carrier* M; $x \in a]\!] \Longrightarrow x \in$ *fgs* R M a
⟨*proof*⟩

lemma (**in** *Module*) *fst-chain-subset*:$[\![R$ *module* N; *free-generator* R M H;
$f \in H \to$ *carrier* N; $C \subseteq$ *fsps* R M N f H; $(a, b) \in C]\!] \Longrightarrow a \subseteq$ *carrier* M
⟨*proof*⟩

lemma (**in** *Module*) *empty-fsp*:$[\![R$ *module* N; *free-generator* R M H;
$f \in H \to$ *carrier* $N]\!] \Longrightarrow (\{\}, (\lambda x \in \{\mathbf{0}_M\}.\ \mathbf{0}_N)) \in$ *fsps* R M N f H
⟨*proof*⟩

lemma (**in** *Module*) *mem-fgs-l-comb*:$[\![K \neq \{\}$; $K \subseteq$ *carrier* M; $x \in$ *fgs* R M $K]\!]$
$\Longrightarrow$
$\exists (n::nat).$
$\exists f \in \{j.\ j \leq (n::nat)\} \to K.\ \exists s \in \{j.\ j \leq n\} \to$ *carrier* R.
$x =$ *l-comb* R M n s f
⟨*proof*⟩

lemma *PairE-lemma*: $\exists x\ y.\ p = (x, y)$ ⟨*proof*⟩

lemma (**in** *Module*) *fsps-chain-boundTr1*:$[\![R$ *module* N; *free-generator* R M H;
$f \in H \to$ *carrier* N; $C \subseteq$ *fsps* R M N f H;
$\forall a \in C.\ \forall b \in C.$ *fst* $a \subseteq$ *fst* $b \vee$ *fst* $b \subseteq$ *fst* a; $\forall a\ b.\ (a, b) \in C \longrightarrow$
$(a, b) \in$ *fsps* R M N f H; $\exists x.\ (\exists b.\ (x, b) \in C) \wedge x \neq \{\}]\!] \Longrightarrow$
$fa \in \{j.\ j \leq (n::nat)\} \to \bigcup \{a.\ \exists b.\ (a, b) \in C\}$
$\longrightarrow (\exists (c, d) \in C.\ fa\ `\ \{j.\ j \leq n\} \subseteq c)$
⟨*proof*⟩

lemma (**in** *Module*) *fsps-chain-boundTr1-1*:$[\![R$ *module* N; *free-generator* R M H;
$f \in H \to$ *carrier* N; $C \subseteq$ *fsps* R M N f H;
$\forall a \in C.\ \forall b \in C.$ *fst* $a \subseteq$ *fst* $b \vee$ *fst* $b \subseteq$ *fst* a;
$\exists x.\ (\exists b.\ (x, b) \in C) \wedge x \neq \{\}$;
$fa \in \{j.\ j \leq (n::nat)\} \to \bigcup \{a.\ \exists b.\ (a, b) \in C\}]\!] \Longrightarrow$
$\exists (c, d) \in C.\ fa\ `\ \{j.\ j \leq n\} \subseteq c$
⟨*proof*⟩

lemma (**in** *Module*) *fsps-chain-boundTr1-2*:$[\![R$ *module* N; *free-generator* R M H;
$f \in H \to$ *carrier* N; $C \subseteq$ *fsps* R M N f H;
$\forall a \in C.\ \forall b \in C.$ *fst* $a \subseteq$ *fst* $b \vee$ *fst* $b \subseteq$ *fst* a;
$\exists x.\ (\exists b.\ (x, b) \in C) \wedge x \neq \{\}$;

$fa \in \{j.\ j \leq (n::nat)\} \to \bigcup \{a.\ \exists\, b.\ (a,\ b) \in C\}\rrbracket \Longrightarrow$
$\exists\, P \in C.\ fa\ `\ \{j.\ j \leq n\} \subseteq fst\ P$
⟨*proof*⟩

lemma (**in** *Module*) *eSum-in-SubmoduleTr*:$\llbracket H \subseteq carrier\ M;\ K \subseteq H\rrbracket \Longrightarrow$
$f \in \{j.\ j \leq (n::nat)\} \to K \wedge s \in \{j.\ j \leq n\} \to carrier\ R \longrightarrow$
$l\text{-}comb\ R\ (mdl\ M\ (fgs\ R\ M\ K))\ n\ s\ f = l\text{-}comb\ R\ M\ n\ s\ f$
⟨*proof*⟩

lemma (**in** *Module*) *eSum-in-Submodule*:$\llbracket H \subseteq carrier\ M;\ K \subseteq H;$
$f \in \{j.\ j \leq (n::nat)\} \to K;\ s \in \{j.\ j \leq n\} \to carrier\ R\rrbracket \Longrightarrow$
$l\text{-}comb\ R\ (mdl\ M\ (fgs\ R\ M\ K))\ n\ s\ f = l\text{-}comb\ R\ M\ n\ s\ f$
⟨*proof*⟩

lemma (**in** *Module*) *fgs-generator*:$H \subseteq carrier\ M \Longrightarrow$
$generator\ R\ (mdl\ M\ (fgs\ R\ M\ H))\ H$
⟨*proof*⟩

lemma (**in** *Module*) *fgs-mono*:$\llbracket free\text{-}generator\ R\ M\ H;\ J \subseteq K;\ K \subseteq H\rrbracket$
$\Longrightarrow fgs\ R\ M\ J \subseteq fgs\ R\ M\ K$
⟨*proof*⟩

lemma (**in** *Module*) *empty-fgs*:$fgs\ R\ M\ \{\} = \{\mathbf{0}\}$
⟨*proof*⟩

lemma (**in** *Module*) *mem-fsps-snd-mHom*:$\llbracket R\ module\ N;\ free\text{-}generator\ R\ M\ H;$
$f \in H \to carrier\ N;\ (a,\ b) \in fsps\ R\ M\ N\ f\ H\rrbracket \Longrightarrow$
$b \in mHom\ R\ (mdl\ M\ (fgs\ R\ M\ a))\ N$
⟨*proof*⟩

lemma (**in** *Module*) *mem-fsps-fst-sub*:$\llbracket R\ module\ N;\ free\text{-}generator\ R\ M\ H;$
$f \in H \to carrier\ N;\ (a,\ b) \in fsps\ R\ M\ N\ f\ H\rrbracket \Longrightarrow a \subseteq H$
⟨*proof*⟩

lemma (**in** *Module*) *mem-fsps-fst-sub1*:$\llbracket R\ module\ N;\ free\text{-}generator\ R\ M\ H;$
$f \in H \to carrier\ N;\ (a,\ b) \in fsps\ R\ M\ N\ f\ H\rrbracket \Longrightarrow a \subseteq carrier\ M$
⟨*proof*⟩

lemma (**in** *Module*) *Order-od-fm-fun*:$\llbracket R\ module\ N;\ free\text{-}generator\ R\ M\ H;$
$f \in H \to carrier\ N\rrbracket \Longrightarrow Order\ (od\text{-}fm\text{-}fun\ R\ M\ N\ f\ H)$
⟨*proof*⟩

lemma (**in** *Module*) *fsps-chain-boundTr2-1*:$\llbracket R\ module\ N;$
$free\text{-}generator\ R\ M\ H;\ f \in H \to carrier\ N;\ C \subseteq fsps\ R\ M\ N\ f\ H;$
$(a,\ b) \in C;\ (aa,\ ba) \in C;\ x \in fgs\ R\ M\ a;\ x \in fgs\ R\ M\ aa;\ a \subseteq aa\rrbracket$
$\Longrightarrow b\ x = ba\ x$
⟨*proof*⟩

lemma (**in** *Module*) *fsps-chain-boundTr2-2*:⟦*R module N*; *free-generator R M H*;
$f \in H \to$ *carrier N*; $C \subseteq$ *fsps R M N f H*;
$\forall\, a{\in}C.\ \forall\, b{\in}C.$ *fst a* $\subseteq$ *fst b* $\vee$ *fst b* $\subseteq$ *fst a*; $C \neq \{\}$; $(a,\ b) \in C$;
$x \in$ *fgs R M a*; $(a1,\ b1) \in C$; $x \in$ *fgs R M a1*⟧ $\Longrightarrow$ $b\ x = b1\ x$
⟨*proof*⟩

lemma (**in** *Module*) *fsps-chain-boundTr2*:$\bigwedge x.$ ⟦*R module N*; *free-generator R M H*;
$f \in H \to$ *carrier N*; $C \subseteq$ *fsps R M N f H*;
$\forall\, a{\in}C.\ \forall\, b{\in}C.$ *fst a* $\subseteq$ *fst b* $\vee$ *fst b* $\subseteq$ *fst a*;
$x \in$ (*fgs R M* ($\bigcup\{a.\ \exists\, b.\ (a,\ b) \in C\}$)); $C \neq \{\}$⟧ $\Longrightarrow$
(*THE y.* $y \in$ *carrier N* $\wedge$ ($\exists\, a\ b.\ (a,\ b) \in C \wedge x \in$ *fgs R M a* $\wedge\ y = b\ x$)) $\in$
(*carrier N*) $\wedge$
($\exists\, a1\ b1.\ (a1,\ b1) \in C \wedge x \in$ *fgs R M a1* $\wedge$
(*THE y.* $y \in$ *carrier N* $\wedge$ ($\exists\, a\ b.\ (a,\ b) \in C \wedge x \in$ *fgs R M a* $\wedge\ y = b\ x$)) =
b1 x)
⟨*proof*⟩

lemma (**in** *Module*) *Un-C-submodule*:⟦*R module N*; *free-generator R M H*;
$f \in H \to$ *carrier N*; $C \subseteq$ *fsps R M N f H*; $C \neq \{\}$;
$\forall\, a{\in}C.\ \forall\, b{\in}C.$ *fst a* $\subseteq$ *fst b* $\vee$ *fst b* $\subseteq$ *fst a*⟧ $\Longrightarrow$
submodule R M (*fgs R M* ($\bigcup\{a.\ \exists\, b.\ (a,\ b) \in C\}$))
⟨*proof*⟩

lemma (**in** *Module*) *Un-C-fgs-sub*:⟦*R module N*; *free-generator R M H*;
$f \in H \to$ *carrier N*; $C \subseteq$ *fsps R M N f H*; $C \neq \{\}$;
$\forall\, a{\in}C.\ \forall\, b{\in}C.$ *fst a* $\subseteq$ *fst b* $\vee$ *fst b* $\subseteq$ *fst a*⟧ $\Longrightarrow$
$\bigcup\{a.\ \exists\, b.\ (a,\ b) \in C\} \subseteq$ *fgs R M* ($\bigcup\{a.\ \exists\, b.\ (a,\ b) \in C\}$)
⟨*proof*⟩

lemma (**in** *Module*) *Chain-3-supset*:⟦*R module N*; *free-generator R M H*;
$f \in H \to$ *carrier N*; $C \subseteq$ *fsps R M N f H*; $C \neq \{\}$;
$\forall\, a{\in}C.\ \forall\, b{\in}C.$ *fst a* $\subseteq$ *fst b* $\vee$ *fst b* $\subseteq$ *fst a*; $(a1,\ b1) \in C$; $(a2,\ b2) \in C$;
$(a3,\ b3) \in C$⟧ $\Longrightarrow$ $\exists\, (g,\ h){\in}C.\ a1 \subseteq g \wedge a2 \subseteq g \wedge a3 \subseteq g$
⟨*proof*⟩

lemma (**in** *Module*) *fsps-chain-bound1*:⟦*R module N*; *free-generator R M H*;
$f \in H \to$ *carrier N*; $C \subseteq$ *fsps R M N f H*;
$\forall\, a{\in}C.\ \forall\, b{\in}C.$ *fst a* $\subseteq$ *fst b* $\vee$ *fst b* $\subseteq$ *fst a*; $C \neq \{\}$⟧ $\Longrightarrow$
($\lambda x{\in}$(*fgs R M* ($\bigcup\{a.\ \exists\, b.\ (a,\ b) \in C\}$)). (*THE y.* $y \in$ *carrier N* $\wedge$
($\exists\, a\ b.\ (a,\ b) \in C \wedge x \in$ *fgs R M a* $\wedge\ y = b\ x$))) $\in$
mHom R (*mdl M* (*fgs R M* ($\bigcup\{a.\ \exists\, b.\ (a,\ b) \in C\}$))) *N*
⟨*proof*⟩

lemma (**in** *Module*) *fsps-chain-bound2*:⟦*R module N*; *free-generator R M H*;
$f \in H \to$ *carrier N*; $C \subseteq$ *fsps R M N f H*; $C \neq \{\}$;
$\forall\, a{\in}C.\ \forall\, b{\in}C.$ *fst a* $\subseteq$ *fst b* $\vee$ *fst b* $\subseteq$ *fst a*⟧ $\Longrightarrow$
$\forall\, y{\in}(\bigcup\{a.\ \exists\, b.\ (a,\ b) \in C\}).\ (\lambda x{\in}(\textit{fgs R M}\ (\bigcup\{a.\ \exists\, b.\ (a,\ b) \in C\})).$
(*THE y.* $y \in$ *carrier N* $\wedge$ ($\exists\, a\ b.\ (a,\ b) \in C \wedge x \in$ *fgs R M a* $\wedge\ y = b\ x$))) $y =$

$f\ y$
⟨*proof*⟩

lemma (**in** *Module*) *od-fm-fun-Chain*:⟦*R module N*; *free-generator R M H*;
$f \in H \to$ *carrier N*; *Algebra2.Chain* (*od-fm-fun R M N f H*) *C*; $C \neq \{\}$⟧ ⟹
$\forall\, a{\in}C.\ \forall\, b{\in}C.$ *fst a* ⊆ *fst b* ∨ *fst b* ⊆ *fst a*
⟨*proof*⟩

lemma (**in** *Module*) *od-fm-fun-inPr0*:⟦*R module N*; *free-generator R M H*;
$f \in H \to$ *carrier N*; *Algebra2.Chain* (*od-fm-fun R M N f H*) *C*; $C \neq \{\}$;
$\exists\, b.\ (y,\ b) \in C$; $z \in y$⟧ ⟹ $z \in$ *fgs R M* ($\bigcup\{a.\ \exists\, b.\ (a,\ b) \in C\}$)
⟨*proof*⟩

lemma (**in** *Module*) *od-fm-fun-indPr1*: ⟦*R module N*; *free-generator R M H*;
$f \in H \to$ *carrier N*; *Algebra2.Chain* (*od-fm-fun R M N f H*) *C*; $C \neq \{\}$⟧ ⟹
($\bigcup\{a.\ \exists\, b.\ (a,\ b) \in C\}$,
$\lambda x \in$ *fgs R M* ($\bigcup\{a.\ \exists\, b.\ (a,\ b) \in C\}$). *THE y*. $y \in$ *carrier N* ∧
($\exists\, a\ b.\ (a,\ b) \in C \wedge x \in$ *fgs R M a* ∧ $y = b\ x$)) ∈ *fsps R M N f H*
⟨*proof*⟩

lemma (**in** *Module*) *od-fm-fun-indPr2*: ⟦*R module N*; *free-generator R M H*;
$f \in H \to$ *carrier N*; *Algebra2.Chain* (*od-fm-fun R M N f H*) *C*; $C \neq \{\}$⟧ ⟹
$^{ub}_{\textit{od-fm-fun R M N f H}}\ C$ ($\bigcup\{a.\ \exists\, b.\ (a,\ b) \in C\}$,
$\lambda x \in$ *fgs R M* ($\bigcup\{a.\ \exists\, b.\ (a,\ b) \in C\}$). *THE y*. $y \in$ *carrier N* ∧
($\exists\, a\ b.\ (a,\ b) \in C \wedge x \in$ *fgs R M a* ∧ $y = b\ x$))
⟨*proof*⟩

lemma (**in** *Module*) *od-fm-fun-inductive*:⟦*R module N*; *free-generator R M H*;
$f \in H \to$ *carrier N*⟧ ⟹ *inductive-set* (*od-fm-fun R M N f H*)
⟨*proof*⟩

lemma (**in** *Module*) *sSum-eq*:⟦*R module N*; *free-generator R M H*; *H1* ⊆ *H*;
$h \in H - H1$⟧ ⟹ (*fgs R M H1*) ∓ (*fgs R M* {*h*}) = *fgs R M* (*H1* ∪ {*h*})
⟨*proof*⟩

definition
monofun :: [('*r*, '*m*) *Ring-scheme*, ('*a*, '*r*, '*m1*) *Module-scheme*,
('*b*, '*r*, '*m2*) *Module-scheme*, '*a* ⇒ '*b*, '*a set*, '*a*] ⇒ ('*a* ⇒ '*b*) **where**
monofun R M N f H h = ($\lambda x{\in}$*fgs R M* {*h*}.
(*THE y*. ($\exists\, r{\in}$*carrier R*. $x = r \cdot_{s M} h \wedge\ \ y = r \cdot_{s N} (f\ h)$)))

lemma (**in** *Module*) *fgs-single-span*:⟦$h \in$ *carrier M*; $x \in$ (*fgs R M* {*h*})⟧ ⟹
$\exists\, a{\in}$*carrier R*. $x = a \cdot_s h$
⟨*proof*⟩

lemma (**in** *Module*) *monofun-mHomTr*:⟦$h \in H$; *free-generator R M H*;
$a \in$ *carrier R*; $r \in$ *carrier R*; $a \cdot_s h = r \cdot_s h$⟧ ⟹ $a = r$
⟨*proof*⟩

lemma (**in** *Module*) *monofun-mhomTr1*:⟦*R module N*; $h \in H$; *free-generator R M H*;
$f \in H \to$ *carrier N*; $a \in$ *carrier R*⟧ $\Longrightarrow$
monofun R M N f H h $(a \cdot_s h) = a \cdot_{s_N} (f\ h)$
⟨*proof*⟩

lemma (**in** *Module*) *monofun-mem*:⟦*R module N*; $h \in H$; *free-generator R M H*;
$x \in$ *fgs R M* $\{h\}$; $f \in H \to$ *carrier N*⟧ $\Longrightarrow$
monofun R M N f H h x $\in$ *carrier N*
⟨*proof*⟩

lemma (**in** *Module*) *monofun-eq-f*:⟦*R module N*; $h \in H$; *free-generator R M H*;
$f \in H \to$ *carrier N*⟧ $\Longrightarrow$ *monofun R M N f H h h* $= f\ h$
⟨*proof*⟩

lemma (**in** *Module*) *sSum-unique*:⟦*R module N*; *free-generator R M H*; $H1 \subseteq H$;
$h \in H - H1$; *x1* $\in$(*fgs R M H1*); *x2* $\in$ (*fgs R M H1*);
y1 $\in$ (*fgs R M* $\{h\}$); *y2* $\in$ (*fgs R M* $\{h\}$); *x1* $\pm$ *y1* $=$ *x2* $\pm$ *y2*⟧ $\Longrightarrow$
x1 $=$ *x2* $\wedge$ *y1* $=$ *y2*
⟨*proof*⟩

lemma (**in** *Module*) *ex-extensionTr*:⟦*R module N*; *free-generator R M H*;
$f \in H \to$ *carrier N*; $H1 \subseteq H$; $h \in H$; $h \notin H1$;
$g \in$ *mHom R* (*mdl M* (*fgs R M H1*)) *N*;
x $\in$ *fgs R M H1* $\mp$ (*fgs R M* $\{h\}$)⟧ $\Longrightarrow$
$\exists\, k1 \in$ *fgs R M H1*. $\exists\, k2 \in$*fgs R M* $\{h\}$. *x* $=$ *k1* $\pm$ *k2* $\wedge$
(*THE y*. $\exists\, h1 \in$*fgs R M H1*. $\exists\, h2 \in$*fgs R M* $\{h\}$. *x* $=$ *h1* $\pm$ *h2* $\wedge$ *y* $=$ *g h1* $\pm_N$
(*monofun R M N f H h h2*)) $=$ *g k1* $\pm_N$ (*monofun R M N f H h k2*)
⟨*proof*⟩

lemma (**in** *Module*) *monofun-add*:⟦*R module N*; *free-generator R M H*;
$f \in H \to$ *carrier N*; $h \in H$; *x* $\in$ *fgs R M* $\{h\}$; *y* $\in$ *fgs R M* $\{h\}$⟧ $\Longrightarrow$
monofun R M N f H h $(x \pm y) =$
monofun R M N f H h x $\pm_N$ (*monofun R M N f H h y*)
⟨*proof*⟩

lemma (**in** *Module*) *monofun-sprod*:⟦*R module N*; *free-generator R M H*;
$f \in H \to$ *carrier N*; $h \in H$; *x* $\in$ *fgs R M* $\{h\}$; $a \in$ *carrier R*⟧ $\Longrightarrow$
monofun R M N f H h $(a \cdot_s x) = a \cdot_{s_N}$ (*monofun R M N f H h x*)
⟨*proof*⟩

lemma (**in** *Module*) *monofun-0*:⟦*R module N*; *free-generator R M H*;
$f \in H \to$ *carrier N*; $h \in H$⟧ $\Longrightarrow$ *monofun R M N f H h* $\mathbf{0} = \mathbf{0}_N$
⟨*proof*⟩

lemma (**in** *Module*) *ex-extension*:⟦*R module N*; *free-generator R M H*;
$f \in H \to$ *carrier N*; $H1 \subseteq H$; $h \in H - H1$; $(H1, g) \in$ *fsps R M N f H*⟧ $\Longrightarrow$
$\exists\, k.\ ((H1 \cup \{h\}), k) \in$ *fsps R M N f H*
⟨*proof*⟩

lemma (**in** *Module*) *mHom-mHom*:⟦*R module N*; *g* ∈ *mHom R* (*mdl M* (*carrier M*)) *N*⟧
⟹ *g* ∈ *mHom R M N*
⟨*proof*⟩

lemma (**in** *Module*) *exist-extension-mhom*:⟦*R module N*; *free-generator R M H*; *f* ∈ *H* → *carrier N*⟧ ⟹ ∃ *g*∈*mHom R M N*. ∀ *x*∈*H*. *g x* = *f x*
⟨*proof*⟩

5.6 Nakayama lemma

definition
Lcg :: [('*r*, '*m*) *Ring-scheme*, ('*a*, '*r*, '*m1*) *Module-scheme*, *nat*] ⇒ *bool* **where**
Lcg R M j ⟷ (∃ *H*. *finite-generator R M H* ∧ *j* = *card H*)

lemma (**in** *Module*) *NAKTr1*:*M fgover R* ⟹
∃ *H*. *finite-generator R M H* ∧ (*LEAST j*.
∃ *L*. *finite-generator R M L* ∧ *j* = *card L*) = *card H*
⟨*proof*⟩

lemma (**in** *Module*) *NAKTr2*:⟦*Lcg R M j*; *k* < (*LEAST j*. *Lcg R M j*)⟧ ⟹
¬ *Lcg R M k*
⟨*proof*⟩

lemma (**in** *Module*) *NAKTr3*:⟦*M fgover R*; *H* ⊆ *carrier M*; *finite H*;
card H < (*LEAST j*. ∃ *L*. *finite-generator R M L* ∧ *j* = *card L*)⟧ ⟹
¬ *finite-generator R M H*
⟨*proof*⟩

lemma (**in** *Module*) *finite-gen-over-ideal*:⟦*ideal R A*; *h* ∈ {*j*. *j* ≤ (*n*::*nat*)} →
carrier M; *generator R M* (*h* ' {*j*. *j* ≤ *n*}); *A* ⊙$_R$ *M* = *carrier M*;
m ∈ *carrier M* ⟧ ⟹ ∃ *s*∈{*j*. *j* ≤ *n*} → *A*. *m* = *l-comb R M n s h*
⟨*proof*⟩

lemma (**in** *Module*) *NAKTr4*:⟦*ideal R A*; *h* ∈ {*j*. *j* ≤ (*k*::*nat*)} → *carrier M*;
0 < *k*; *h* ' {*j*. *j* ≤ *k*} ⊆ *carrier M*; *s* ∈ {*j*. *j* ≤ *k*} → *A*;
h k = Σ$_e$ *M* (λ*j*. *s j* ·$_s$ (*h j*)) (*k* − *Suc 0*) ± (*s k* ·$_s$ (*h k*))⟧ ⟹
(*1*$_{r\,R}$ ±$_R$ (−$_{a\,R}$ (*s k*))) ·$_s$ (*h k*) = Σ$_e$ *M* (λ*j*. *s j* ·$_s$ (*h j*)) (*k* − *Suc 0*)
⟨*proof*⟩

lemma (**in** *Module*) *NAKTr5*:⟦¬ *zeroring R*; *ideal R A*; *A* ⊆ *J-rad R*;
A ⊙$_R$ *M* = *carrier M*; *finite-generator R M H*; *card H* = *Suc k*; *0* < *k*⟧ ⟹
∃ *h* ∈ {*j*. *j* ≤ *k*} → *carrier M*. *H* = *h* ' {*j*. *j* ≤ *k*} ∧
h k ∈ *linear-span R M A* (*h* ' {*j*. *j* ≤ (*k* − *Suc 0*)})
⟨*proof*⟩

lemma (**in** *Module*) *NAK*:⟦¬ *zeroring R*; *M fgover R*; *ideal R A*; $A \subseteq$ *J-rad R*;
$A \odot_R M = carrier\ M$ ⟧ ⟹ $carrier\ M = \{\mathbf{0}\}$
⟨*proof*⟩

lemma (**in** *Module*) *fg-qmodule*:⟦ *M fgover R*; *submodule R M N*⟧ ⟹
$(M\ /_m\ N)$ *fgover R*
⟨*proof*⟩

lemma (**in** *Module*) *NAK1*:⟦¬ *zeroring R*; *M fgover R*; *submodule R M N*;
ideal R A; $A \subseteq$ *J-rad R*; $carrier\ M = A \odot_R M \mp N$ ⟧ ⟹ $carrier\ M = N$
⟨*proof*⟩

5.7 Direct sum and direct products of modules

definition
prodM-sprod :: [('*r*, '*m*) *Ring-scheme*, '*i set*,
'*i* ⇒ ('*a*, '*r*, '*m1*) *Module-scheme*] ⇒ '*r* ⇒ ('*i* ⇒ '*a*) ⇒ ('*i* ⇒ '*a*) **where**
prodM-sprod R I A = (λ*a*∈*carrier R*. λ*g*∈*carr-prodag I A*.
(λ*j*∈*I*. *a* ·$_{s(A\ j)}$ (*g j*)))

definition
prodM :: [('*r*, '*m*) *Ring-scheme*, '*i set*, '*i* ⇒ ('*a*, '*r*, '*m1*) *Module-scheme*] ⇒
(| *carrier*:: ('*i* ⇒ '*a*) *set*,
pop::['*i* ⇒ '*a*, '*i* ⇒ '*a*] ⇒ ('*i* ⇒ '*a*),
mop:: ('*i* ⇒ '*a*) ⇒ ('*i* ⇒ '*a*), *zero*::('*i* ⇒ '*a*),
sprod :: ['*r*, '*i* ⇒ '*a*] ⇒ ('*i* ⇒ '*a*) |) **where**
prodM R I A = (|*carrier* = *carr-prodag I A*,
pop = *prod-pOp I A*, *mop* = *prod-mOp I A*,
zero = *prod-zero I A*, *sprod* = *prodM-sprod R I A* |)

definition
mProject :: [('*r*, '*m*) *Ring-scheme*, '*i set*,
'*i* ⇒ ('*a*, '*r*, '*more*) *Module-scheme*, '*i*] ⇒ ('*i* ⇒ '*a*) ⇒ '*a* **where**
mProject R I A j = (λ*f*∈*carr-prodag I A*. *f j*)

abbreviation
PRODMODULES (‹(*3m*Π_ _ -)› [*72*,*72*,*73*]*72*) **where**
*m*Π$_{R\ I}$ *A* == *prodM R I A*

lemma (**in** *Ring*) *prodM-carr*:⟦∀ *i*∈*I*. (*R module* (*M i*))⟧ ⟹
carrier (*prodM R I M*) = *carr-prodag I M*
⟨*proof*⟩

lemma (**in** *Ring*) *prodM-mem-eq*:⟦∀ *i*∈*I*. (*R module* (*M i*));
m1 ∈ *carrier* (*prodM R I M*); *m2* ∈ *carrier* (*prodM R I M*);
∀ *i*∈*I*. *m1 i* = *m2 i* ⟧ ⟹ *m1* = *m2*
⟨*proof*⟩

lemma (**in** *Ring*) *prodM-sprod-mem*:⟦∀ *i*∈*I*. (*R module* (*M i*)); *a* ∈ *carrier R*;

$m \in$ *carr-prodag I M*$\rrbracket \Longrightarrow$ *prodM-sprod R I M a m* $\in$ *carr-prodag I M*
⟨*proof*⟩

lemma (**in** *Ring*) *prodM-sprod-val*:$\llbracket \forall\, i{\in}I.\ (R$ *module* $(M\ i));\ a \in$ *carrier* R;
$m \in$ *carr-prodag I M*; $j \in I\rrbracket \Longrightarrow$ (*prodM-sprod R I M a m*) $j = a \cdot_{s(M\ j)} (m\ j)$
⟨*proof*⟩

lemma (**in** *Ring*) *prodM-Module*:$\forall\, i{\in}I.\ (R$ *module* $(M\ i)) \Longrightarrow$
R module (*prodM R I M*)
⟨*proof*⟩

definition
dsumM :: [('*r*, '*m*) *Ring-scheme*, '*i set*, '*i* ⇒ ('*a*, '*r*, '*more*) *Module-scheme*]
⇒ ⦇ *carrier*:: ('*i* ⇒ '*a*) *set*,
pop::['*i* ⇒ '*a*, '*i* ⇒ '*a*] ⇒ ('*i* ⇒ '*a*),
mop:: ('*i* ⇒ '*a*) ⇒ ('*i* ⇒ '*a*),
zero::('*i* ⇒ '*a*),
sprod :: ['*r*, '*i* ⇒ '*a*] ⇒ ('*i* ⇒ '*a*) ⦈ **where**

dsumM R I A = ⦇ *carrier* = *carr-dsumag I A*,
pop = *prod-pOp I A*, *mop* = *prod-mOp I A*,
zero = *prod-zero I A*, *sprod* = *prodM-sprod R I A*⦈

abbreviation
DSUMMOD (‹(*3*_Σ_{d}_ -)› [*72*,*72*,*73*]*72*) **where**
${}_R\Sigma_{d\,I}\ A == dsumM\ R\ I\ A$

lemma (**in** *Ring*) *dsumM-carr*:*carrier* (*dsumM R I M*) = *carr-dsumag I M*
⟨*proof*⟩

lemma (**in** *Ring*) *dsum-sprod-mem*:$\llbracket \forall\, i{\in}I.\ R$ *module* $M\ i$; $a \in$ *carrier* R;
$b \in$ *carr-dsumag I M*$\rrbracket \Longrightarrow$ *prodM-sprod R I M a b* $\in$ *carr-dsumag I M*
⟨*proof*⟩

lemma (**in** *Ring*) *carr-dsum-prod*:*carr-dsumag I M* $\subseteq$ *carr-prodag I M*
⟨*proof*⟩

lemma (**in** *Ring*) *carr-dsum-prod1*:
$\forall\, x.\ x \in$ *carr-dsumag I M* $\longrightarrow x \in$ *carr-prodag I M*
⟨*proof*⟩

lemma (**in** *Ring*) *carr-dsumM-mem-eq*:$\llbracket \forall\, i{\in}I.\ R$ *module* $M\ i$; $x \in$ *carr-dsumag I M*;
$y \in$ *carr-dsumag I M*; $\forall\, j{\in}I.\ x\ j = y\ j\rrbracket \Longrightarrow x = y$
⟨*proof*⟩

lemma (**in** *Ring*) *dsumM-Module*:$\forall\, i{\in}I.\ R$ *module* $(M\ i) \Longrightarrow R$ *module* $({}_R\Sigma_{d\,I}\ M)$
⟨*proof*⟩

definition
ringModule :: ('r, 'b) *Ring-scheme* ⇒ ('r, 'r) *Module*
(‹(M_-)› [998]999) **where**
M_R = (|*carrier* = *carrier R*, *pop* = *pop R*, *mop* = *mop R*,
zero = *zero R*, *sprod* = *tp R*|)

lemma (**in** *Ring*) *ringModule-Module*:*R module* M_R
⟨*proof*⟩

definition
dsumMhom :: ['i *set*, 'i ⇒ ('a, 'r, 'm) *Module-scheme*,
'i ⇒ ('b, 'r, 'm1) *Module-scheme*, 'i ⇒ ('a ⇒ 'b)] ⇒ ('i ⇒ 'a) ⇒
('i ⇒ 'b) **where**

dsumMhom I A B S = (λ*f*∈*carr-dsumag I A*. (λ*k*∈*I*. (*S k*) (*f k*)))

lemma (**in** *Ring*) *dsumMhom-mem*:⟦∀ *i*∈*I*. *R module M i*; ∀ *i*∈*I*. *R module N i*;
∀ *i*∈*I*. *S i* ∈ *mHom R* (*M i*) (*N i*); *x* ∈ *carr-dsumag I M*⟧
⟹ *dsumMhom I M N S x* ∈ *carr-dsumag I N*
⟨*proof*⟩

lemma (**in** *Ring*) *dsumMhom-mHom*:⟦∀ *i*∈*I*. (*R module* (*M i*));
∀ *i*∈*I*. (*R module* (*N i*)); ∀ *i*∈*I*. ((*S i*) ∈ *mHom R* (*M i*) (*N i*))⟧ ⟹
dsumMhom I M N S ∈ *mHom R* (*dsumM R I M*) (*dsumM R I N*)
⟨*proof*⟩

end

theory *Algebra9* **imports** *Algebra8* **begin**

5.8 Exact sequence

definition
Zm :: [('r, 'm) *Ring-scheme*, 'a] ⇒ ('a, 'r) *Module* **where**
Zm R e = (| *carrier* = {*e*}, *pop* = λ*x*∈{*e*}. λ*y*∈{*e*}. *e*, *mop* =
λ*x*∈{*e*}. *e*, *zero* = *e*, *sprod* = λ*r*∈*carrier R*. λ*x*∈{*e*}. *e*|)

lemma (**in** *Ring*) *Zm-Module*:*R module* (*Zm R e*)
⟨*proof*⟩

lemma (**in** *Ring*) *Zm-carrier*:*carrier* (*Zm R e*) = {*e*}
⟨*proof*⟩

lemma (**in** *Ring*) *Zm-to-M-0*:⟦*R module M*; *f* ∈ *mHom R* (*Zm R e*) *M*⟧ ⟹
f e = $\mathbf{0}_M$

⟨*proof*⟩

lemma (**in** *Ring*) *Z-to-M*:⟦*R module M*; $f \in$ *mHom R* (*Zm R e*) *M*;
$g \in$ *mHom R* (*Zm R e*) *M* ⟧ $\Longrightarrow f = g$
⟨*proof*⟩

lemma (**in** *Ring*) *mzeromap-mHom*:⟦*R module M*; *R module N*⟧ $\Longrightarrow$
mzeromap M N $\in$ *mHom R M N*
⟨*proof*⟩

lemma (**in** *Ring*) *HOM-carrier*:*carrier* (HOM_R *M N*) = *mHom R M N*
⟨*proof*⟩

lemma (**in** *Ring*) *mHom-Z-M*:*R module M* $\Longrightarrow$
mHom R (*Zm R e*) *M* = {*mzeromap* (*Zm R e*) *M*}
⟨*proof*⟩

lemma (**in** *Module*) *Modules-single-carrier-isom*:⟦*R module N*; *carrier M* = {**0**};
carrier N = {$\mathbf{0}_N$}⟧ $\Longrightarrow M \cong_R N$
⟨*proof*⟩

lemma (**in** *Ring*) *Zm-isom*:(*Zm R* (*e*::′*a*)) $\cong_R$ (*Zm R* (*u*::′*b*))
⟨*proof*⟩

lemma (**in** *Ring*) *HOM-Z-M-0*:*R module M* $\Longrightarrow$ HOM_R (*Zm R e*) *M* $\cong_R$ (*Zm R e*)
⟨*proof*⟩

lemma (**in** *Ring*) *M-to-Z*:⟦*R module M*; $f \in$ *mHom R M* (*Zm R e*);
$g \in$ *mHom R M* (*Zm R e*)⟧ $\Longrightarrow f = g$
⟨*proof*⟩

lemma (**in** *Ring*) *mHom-to-zero*:*R module M* $\Longrightarrow$ *mHom R M* (*Zm R e*) =
{*mzeromap M* (*Zm R e*)}
⟨*proof*⟩

lemma (**in** *Ring*) *carrier-HOM-M-Z*:*R module M* $\Longrightarrow$
carrier (HOM_R *M* (*Zm R e*)) = {*mzeromap M* (*Zm R e*)}
⟨*proof*⟩

lemma (**in** *Ring*) *HOM-M-Z-0*:*R module M* $\Longrightarrow$ HOM_R *M* (*Zm R e*) $\cong_R$ (*Zm R e*)
⟨*proof*⟩

lemma (**in** *Ring*) *M-to-Z-0*:⟦*R module M*; $f \in$ *mHom R M* (*Zm R e*)⟧ $\Longrightarrow$
$ker_{M,(Zm\ R\ e)}\ f$ = *carrier M*
⟨*proof*⟩

definition

exact3 :: [('r, 'm) *Ring-scheme*, ('a, 'r, 'm1) *Module-scheme*, 'a ⇒ 'b,
('b, 'r, 'm1) *Module-scheme*, 'b ⇒ 'c, ('c, 'r, 'm1) *Module-scheme*] ⇒ *bool*
where
exact3 R L0 h0 L1 h1 L2 == *h0* ‘ (*carrier L0*) = $ker_{(L1),(L2)}$ *h1*

definition
exact4 :: [('r, 'm) *Ring-scheme*, ('a0, 'r, 'm1) *Module-scheme*, 'a0 ⇒ 'a1,
('a1, 'r, 'm1) *Module-scheme*, 'a1 ⇒ 'a2, ('a2, 'r, 'm1) *Module-scheme*,
'a2 ⇒ 'a3, ('a3, 'r, 'm1) *Module-scheme*] ⇒ *bool* **where**
exact4 R L0 h0 L1 h1 L2 h2 L3 ⟷ *h0* ‘ (*carrier L0*) = $ker_{(L1),(L2)}$ *h1* ∧
h1 ‘ (*carrier L1*) = $ker_{(L2),(L3)}$ *h2*

definition
exact5 :: [('r, 'm) *Ring-scheme*, ('a0, 'r, 'm1) *Module-scheme*, 'a0 ⇒ 'a1,
('a1, 'r, 'm1) *Module-scheme*, 'a1 ⇒ 'a2, ('a2, 'r, 'm1) *Module-scheme*,
'a2 ⇒ 'a3, ('a3, 'r, 'm1) *Module-scheme*, 'a3 ⇒ 'a4,
('a4, 'r, 'm1) *Module-scheme*] ⇒ *bool* **where**
exact5 R L0 h0 L1 h1 L2 h2 L3 h3 L4 == *h0* ‘ (*carrier L0*) = $ker_{(L1),(L2)}$ *h1*
∧
h1 ‘ (*carrier L1*) = $ker_{(L2),(L3)}$ *h2* ∧ *h2* ‘(*carrier L2*) = $ker_{(L3),(L4)}$ *h3*

definition
exact8 :: [('r, 'm) *Ring-scheme*, ('a0, 'r, 'm1) *Module-scheme*, 'a0 ⇒ 'a1,
('a1, 'r, 'm1) *Module-scheme*, 'a1 ⇒ 'a2, ('a2, 'r, 'm1) *Module-scheme*,
'a2 ⇒ 'a3, ('a3, 'r, 'm1) *Module-scheme*, 'a3 ⇒ 'a4,
('a4, 'r, 'm1) *Module-scheme*, 'a4 ⇒ 'a5, ('a5, 'r, 'm1) *Module-scheme*,
'a5 ⇒ 'a6, ('a6, 'r, 'm1) *Module-scheme*, 'a6 ⇒ 'a7,
('a7, 'r, 'm1) *Module-scheme*] ⇒ *bool* **where**
exact8 R L0 h0 L1 h1 L2 h2 L3 h3 L4 h4 L5 h5 L6 h6 L7 ⟷
h0 ‘ (*carrier L0*) = $ker_{(L1),(L2)}$ *h1* ∧ *h1* ‘ (*carrier L1*) = $ker_{(L2),(L3)}$ *h2* ∧
h2 ‘ (*carrier L2*) = $ker_{(L3),(L4)}$ *h3* ∧ *h3* ‘ (*carrier L3*) = $ker_{(L4),(L5)}$ *h4* ∧
h4 ‘ (*carrier L4*) = $ker_{(L5),(L6)}$ *h5* ∧ *h5* ‘ (*carrier L5*) = $ker_{(L6),(L7)}$ *h6*

lemma (**in** *Ring*) *exact3-comp-0*:⟦*R module L*; *R module M*; *R module N*;
f ∈ *mHom R L M*; *g* ∈ *mHom R M N*; *exact3 R L f M g N*⟧ ⟹
compos L g f = *mzeromap L N*
⟨*proof*⟩

lemma (**in** *Ring*) *exact-im-sub-kern*:⟦*R module L*; *R module M*; *R module N*;
f ∈ *mHom R L M*; *g* ∈ *mHom R M N*; *exact3 R L f M g N*⟧ ⟹
f ‘ (*carrier L*) ⊆ $ker_{M,N}$ *g*
⟨*proof*⟩

lemma (**in** *Ring*) *mzero-im-sub-ker*:⟦*R module L*; *R module M*; *R module N*;
f ∈ *mHom R L M*; *g* ∈ *mHom R M N*; *compos L g f* = *mzeromap L N*⟧ ⟹
f ‘ (*carrier L*) ⊆ $ker_{M,N}$ *g*
⟨*proof*⟩

lemma (**in** *Ring*) *left-exact-injec*:⟦*R module M*; *R module N*;

$z \in mHom\ R\ (Zm\ R\ e)\ M;\ f \in mHom\ R\ M\ N;\ exact3\ R\ (Zm\ R\ e)\ z\ M\ f\ N\rrbracket$
$\Longrightarrow$
$injec_{M,N}\ f$
⟨*proof*⟩

lemma (**in** *Ring*) *injec-left-exact*:⟦R *module* M; R *module* N;
$z \in mHom\ R\ (Zm\ R\ e)\ M;\ f \in mHom\ R\ M\ N;\ injec_{M,N}\ f\rrbracket \Longrightarrow$
$exact3\ R\ (Zm\ R\ e)\ z\ M\ f\ N$
⟨*proof*⟩

lemma (**in** *Ring*) *injec-mHom-image*:⟦R *module* N; R *module* $M1$; R *module* $M2$;

$x \in mHom\ R\ N\ M2;\ f \in mHom\ R\ M1\ M2;\ x\ `\ (carrier\ N) \subseteq f\ `\ (carrier\ M1);$
$injec_{M1,M2}\ f\rrbracket \Longrightarrow$
$(\lambda n \in (carrier\ N).\ (SOME\ m.\ (m \in carrier\ M1 \wedge x\ n = f\ m))) \in mHom\ R\ N\ M1 \wedge$
$compos\ N\ f\ (\lambda n \in (carrier\ N).\ (SOME\ m.\ m \in carrier\ M1 \wedge x\ n = f\ m)) = x$
⟨*proof*⟩

lemma (**in** *Ring*) *right-exact-surjec*:⟦R *module* M; R *module* N; $f \in mHom\ R\ M\ N$;
$p \in mHom\ R\ N\ (Zm\ R\ e);\ exact3\ R\ M\ f\ N\ p\ (Zm\ R\ e)\rrbracket \Longrightarrow surjec_{M,N}\ f$
⟨*proof*⟩

lemma (**in** *Ring*) *surjec-right-exact*:⟦R *module* M; R *module* N; $f \in mHom\ R\ M\ N$;
$p \in mHom\ R\ N\ (Zm\ R\ e);\ surjec_{M,N}\ f\rrbracket \Longrightarrow exact3\ R\ M\ f\ N\ p\ (Zm\ R\ e)$
⟨*proof*⟩

lemma (**in** *Ring*) *exact4-exact3*:⟦R *module* M; R *module* N; $z \in mHom\ R\ (Zm\ R\ e)\ M$;
$f \in mHom\ R\ M\ N;\ z1 \in mHom\ R\ N\ (Zm\ R\ e);$
$exact4\ R\ (Zm\ R\ e)\ z\ M\ f\ N\ z1\ (Zm\ R\ e)\ \rrbracket \Longrightarrow$
$exact3\ R\ (Zm\ R\ e)\ z\ M\ f\ N \wedge exact3\ R\ M\ f\ N\ z1\ (Zm\ R\ e)$
⟨*proof*⟩

lemma (**in** *Ring*) *exact4-bijec*:⟦R *module* M; R *module* N; $z \in mHom\ R\ (Zm\ R\ e)\ M$;
$f \in mHom\ R\ M\ N;\ z1 \in mHom\ R\ N\ (Zm\ R\ e);$
$exact4\ R\ (Zm\ R\ e)\ z\ M\ f\ N\ z1\ (Zm\ R\ e)\rrbracket \Longrightarrow bijec_{M,N}\ f$
⟨*proof*⟩

lemma (**in** *Ring*) *exact-im-sub-ker*:⟦R *module* L; R *module* M; R *module* N;
$f \in mHom\ R\ L\ M;\ g \in mHom\ R\ M\ N;\ z1 \in mHom\ R\ N\ (Zm\ R\ e);\ R$ *module* Z;
$exact4\ R\ L\ f\ M\ g\ N\ z1\ (Zm\ R\ e);\ x \in mHom\ R\ M\ Z;\ compos\ L\ x\ f = mzeromap\ L\ Z\rrbracket$

$\Longrightarrow (\lambda z{\in}(carrier\ N).\ x\ (SOME\ y.\ y \in carrier\ M \wedge g\ y = z)) \in mHom\ R\ N\ Z$
⟨*proof*⟩

lemma (**in** *Ring*) *exact-im-sub-ker1*:⟦*R module L*; *R module M*; *R module N*;
f ∈ *mHom R L M*; *g* ∈ *mHom R M N*; *z1* ∈ *mHom R N* (*Zm R e*); *R module Z*;
exact4 R L f M g N z1 (*Zm R e*); *x* ∈ *mHom R M Z*;
compos L x f = *mzeromap L Z* ⟧ ⟹
compos M (λ*z*∈(*carrier N*). *x* (*SOME y*. *y* ∈ *carrier M* ∧ *g y* = *z*)) *g* = *x*
⟨*proof*⟩

definition
module-iota :: [('*r*, '*m*) *Ring-scheme*, ('*a*, '*r*, '*m1*) *Module-scheme*] ⇒
'*a* ⇒ '*a* (‹(*mι*_ -)› [*92*, *93*]*92*) **where**
$m\iota_R\ M = (\lambda x{\in}carrier\ M.\ x)$

lemma (**in** *Ring*) *short-exact-sequence*:⟦*R module M*; *submodule R M N*;
z ∈ *mHom R* (*Zm R e*) (*mdl M N*); *z1* ∈ *mHom R* (*M* /$_m$ *N*) (*Zm R e*)⟧ ⟹
exact5 R (*Zm R e*) *z* (*mdl M N*)($m\iota_R$ (*mdl M N*)) *M* (*mpj M N*) (*M* /$_m$ *N*) *z1* (*Zm R e*)
⟨*proof*⟩

lemma (**in** *Ring*) *rexact4-lexact4-HOM*:⟦*R module M1*; *R module M2*; *R module M3*;
f ∈ *mHom R M1 M2*; *g* ∈ *mHom R M2 M3*; *z1* ∈ *mHom R M3* (*Zm R e*);
exact4 R M1 f M2 g M3 z1 (*Zm R e*)⟧ ⟹
∀ *N*. *R module N* ⟶
exact4 R (HOM_R (*Zm R e*) *N*) (*sup-sharp R M3* (*Zm R e*) *N z1*) (HOM_R *M3 N*)
(*sup-sharp R M2 M3 N g*) (HOM_R *M2 N*) (*sup-sharp R M1 M2 N f*) (HOM_R *M1 N*)

⟨*proof*⟩

lemma *exact-HOM-exactTr*:⟦*Ring* (*R*::('*r*, '*m1*) *Ring-scheme*); *f* ∈ *mHom R M1 M2*;
g ∈ *mHom R M2 M3*; *z1* ∈ *mHom R M3* (*Zm R e*); *R module NV*;
∀(*N*::('*a*, '*r*, '*m*) *Module-scheme*). *R module N* ⟶
exact4 R (HOM_R (*Zm R e*) *N*)(*sup-sharp R M3* (*Zm R e*) *N z1*)
(HOM_R *M3 N*) (*sup-sharp R M2 M3 N g*) (HOM_R *M2 N*) (*sup-sharp R M1 M2 N f*)
(HOM_R *M1 N*); *R module* (*L*::('*a*, '*r*, '*m*) *Module-scheme*)⟧ ⟹
exact4 R (HOM_R (*Zm R e*) *L*) (*sup-sharp R M3* (*Zm R e*) *L z1*)
(HOM_R *M3 L*) (*sup-sharp R M2 M3 L g*) (HOM_R *M2 L*) (*sup-sharp R M1 M2 L f*)
(HOM_R *M1 L*)

⟨*proof*⟩

lemma *lexact4-rexact4-HOM*:⟦*Ring R*; *R module M1*; *R module M2*; *R module M3*;
f ∈ *mHom R M1 M2*; *g* ∈ *mHom R M2 M3*; *z* ∈ *mHom R* (*Zm R e*) *M1*;
exact4 R (*Zm R e*) *z M1 f M2 g M3* ⟧ ⟹
∀ *N*. *R module N* ⟶ *exact4 R* (*HOM*$_R$ *N* (*Zm R e*)) (*sub-sharp R N* (*Zm R e*) *M1 z*)
(*HOM*$_R$ *N M1*) (*sub-sharp R N M1 M2 f*) (*HOM*$_R$ *N M2*) (*sub-sharp R N M2 M3 g*)
(*HOM*$_R$ *N M3*)

⟨*proof*⟩

5.9 Tensor product

definition
prod-carr :: [('*a*, '*r*, '*m*) *Module-scheme*, ('*b*, '*r*, '*m*) *Module-scheme*]
⇒ ('*a* ∗ '*b*) *set* (**infixl** ‹×$_c$› *100*) **where**
M ×$_c$ *N* = *carrier M* × *carrier N*

definition
bilinear-map :: ['*a* ∗ '*b* ⇒ '*c*, ('*r*, '*m*) *Ring-scheme*,
('*a*, '*r*, '*m1*) *Module-scheme*, ('*b*, '*r*, '*m1*) *Module-scheme*,
('*c*, '*r*, '*m1*) *Module-scheme*] ⇒ *bool* **where**
bilinear-map f R M1 M2 N ⟷ *f* ∈ *M1* ×$_c$ *M2* → *carrier N* ∧
f ∈ *extensional* (*M1* ×$_c$ *M2*) ∧
(∀ *x1* ∈ *carrier M1*. ∀ *x2* ∈ *carrier M1*.
∀ *y*∈*carrier M2*.(*f* (*x1* ±$_{M1}$ *x2*, *y*) = *f* (*x1*, *y*) ±$_N$ (*f* (*x2*, *y*)))) ∧
(∀ *x*∈*carrier M1*. ∀ *y1*∈*carrier M2*.
∀ *y2*∈*carrier M2*. *f* (*x*, *y1* ±$_{M2}$ *y2*) = *f* (*x*, *y1*) ±$_N$ (*f* (*x*, *y2*))) ∧
(∀ *x*∈*carrier M1*. ∀ *y*∈*carrier M2*.
∀ *r*∈*carrier R*. *f* (*r* ·$_{s M1}$ *x*, *y*) = *r* ·$_{s N}$ (*f* (*x*, *y*)) ∧
f (*x*, *r* ·$_{s M2}$ *y*) = *r* ·$_{s N}$ (*f* (*x*, *y*)))

lemma (**in** *Ring*) *prod-carr-mem*:⟦*R module M*; *R module N*; *m* ∈ *carrier M*;
n ∈ *carrier N*⟧ ⟹ (*m*, *n*) ∈ *M* ×$_c$ *N*
⟨*proof*⟩

lemma (**in** *Ring*) *bilinear-func*:*bilinear-map f R M N Z* ⟹
f ∈ *M* ×$_c$ *N* → *carrier Z*
⟨*proof*⟩

lemma (**in** *Ring*) *bilinear-mem*:⟦*R module M1*; *R module M2*; *R module N*;
m1 ∈ *carrier M1*; *m2* ∈ *carrier M2*; *bilinear-map f R M1 M2 N*⟧ ⟹
f (*m1*, *m2*) ∈ *carrier N*

⟨*proof*⟩

lemma (**in** *Ring*) *bilinear-l-add*:⟦*R module M1*; *R module M2*; *R module N*;
m11 ∈ *carrier M1*; *m12* ∈ *carrier M1*; *m2* ∈ *carrier M2*;
bilinear-map f R M1 M2 N⟧ ⟹
$f\ (m11 \pm_{M1} m12,\ m2) = f\ (m11,\ m2) \pm_N (f\ (m12,\ m2))$
⟨*proof*⟩

lemma (**in** *Ring*) *bilinear-l-add1*:⟦*R module M1*; *R module M2*; *R module N*;
m11 ∈ *carrier M1*; *m12* ∈ *carrier M1*; *m2* ∈ *carrier M2*;
bilinear-map f R M1 M2 N⟧ ⟹
$f\ (m11 \pm_{M1} m12,\ m2) \pm_N -_{aN} (f\ (m11,\ m2) \pm_N (f\ (m12,\ m2))) = \mathbf{0}_N$
⟨*proof*⟩

lemma (**in** *Ring*) *bilinear-r-add*:⟦*R module M1*; *R module M2*; *R module N*;
m ∈ *carrier M1*; *m21* ∈ *carrier M2*; *m22* ∈ *carrier M2*;
bilinear-map f R M1 M2 N⟧ ⟹
$f\ (m,\ m21 \pm_{M2} m22) = f\ (m,\ m21) \pm_N (f\ (m,\ m22))$
⟨*proof*⟩

lemma (**in** *Ring*) *bilinear-r-add1*:⟦*R module M1*; *R module M2*; *R module N*;
m ∈ *carrier M1*; *m21* ∈ *carrier M2*; *m22* ∈ *carrier M2*;
bilinear-map f R M1 M2 N⟧ ⟹
$f\ (m,\ m21 \pm_{M2} m22) \pm_N -_{aN} (f\ (m,\ m21) \pm_N (f\ (m,\ m22))) = \mathbf{0}_N$
⟨*proof*⟩

lemma (**in** *Ring*) *bilinear-l-lin*:⟦*R module M1*; *R module M2*; *R module N*;
m1 ∈ *carrier M1*; *m2* ∈ *carrier M2*; *r* ∈ *carrier R*;
bilinear-map f R M1 M2 N⟧ ⟹ $f\ (r \cdot_{sM1} m1,\ m2) = r \cdot_{sN} (f\ (m1,\ m2))$
⟨*proof*⟩

lemma (**in** *Ring*) *bilinear-l-lin1*:⟦*R module M1*; *R module M2*; *R module N*;
m1 ∈ *carrier M1*; *m2* ∈ *carrier M2*; *r* ∈ *carrier R*;
bilinear-map f R M1 M2 N⟧ ⟹
$f\ (r \cdot_{sM1} m1,\ m2) \pm_N -_{aN} (r \cdot_{sN} (f\ (m1,\ m2))) = \mathbf{0}_N$
⟨*proof*⟩

lemma (**in** *Ring*) *bilinear-r-lin*:⟦*R module M1*; *R module M2*; *R module N*;
m1 ∈ *carrier M1*; *m2* ∈ *carrier M2*; *r* ∈ *carrier R*;
bilinear-map f R M1 M2 N⟧ ⟹ $f\ (m1,\ r \cdot_{sM2} m2) = r \cdot_{sN} (f\ (m1,\ m2))$
⟨*proof*⟩

lemma (**in** *Ring*) *bilinear-r-lin1*:⟦*R module M1*; *R module M2*; *R module N*;
m1 ∈ *carrier M1*; *m2* ∈ *carrier M2*; *r* ∈ *carrier R*;
bilinear-map f R M1 M2 N⟧ ⟹
$f\ (m1,\ r \cdot_{sM2} m2)\ \pm_N -_{aN} (r \cdot_{sN} (f\ (m1,\ m2))) = \mathbf{0}_N$
⟨*proof*⟩

lemma (**in** *Ring*) *bilinear-l-0*:⟦*R module M1*; *R module M2*; *R module N*;

$m2 \in carrier\ M2$; $bilinear\text{-}map\ f\ R\ M1\ M2\ N\rrbracket \Longrightarrow f\ (\mathbf{0}_{M1},\ m2) = \mathbf{0}_N$
⟨*proof*⟩

lemma (**in** *Ring*) *bilinear-r-0*:⟦$R\ module\ M1$; $R\ module\ M2$; $R\ module\ N$;
$m1 \in carrier\ M1$; $bilinear\text{-}map\ f\ R\ M1\ M2\ N\rrbracket \Longrightarrow f\ (m1,\ \mathbf{0}_{M2}) = \mathbf{0}_N$
⟨*proof*⟩

definition
universal-property :: [('*r*, '*m*) *Ring-scheme*, ('*a*, '*r*, '*m1*) *Module-scheme*,
('*b*, '*r*, '*m1*) *Module-scheme*, ('*c*, '*r*, '*m1*) *Module-scheme*,
'*a* ∗ '*b* ⇒'*c*] ⇒ *bool* **where**
universal-property (*R*::('*r*, '*m*) *Ring-scheme*) (*M*::('*a*, '*r*, '*m1*) *Module-scheme*)
(*N*:: ('*b*, '*r*, '*m1*) *Module-scheme*) (*MN*::('*c*, '*r*, '*m1*) *Module-scheme*)
(*f*:: '*a* ∗ '*b* ⇒ '*c*) ⟷ (*bilinear-map f R M N MN*) ∧
(∀(*Z* :: ('*c*, '*r*, '*m1*) *Module-scheme*). ∀(*g* :: '*a* ∗ '*b* ⇒ '*c*). (*R module Z*) ∧
(*bilinear-map g R M N Z*) ⟶ ((∃!*h*. (*h* ∈ *mHom R MN Z*) ∧
(*compose* (*M* ×$_c$ *N*) *h f* = *g*))))

lemma *tensor-prod-uniqueTr*:⟦*Ring R*; *R module* (*M*::('*a*, '*r*, '*m1*) *Module-scheme*);

R module (*N*:: ('*b*, '*r*, '*m1*) *Module-scheme*);
R module (*MN*:: ('*c*, '*r*, '*m1*) *Module-scheme*);
R module (*MN1*::('*c*, '*r*, '*m1*) *Module-scheme*);
universal-property R M N MN f; *universal-property R M N MN1 g*⟧ ⟹
∃!*k*. *k* ∈ *mHom R MN1 MN* ∧ *compose* (*M* ×$_c$ *N*) *k g* = *f*
⟨*proof*⟩

lemma *tensor-prod-unique*:⟦*Ring* (*R*:: ('*r*, '*m*) *Ring-scheme*);
R module (*M* :: ('*a*, '*r*, '*m1*) *Module-scheme*);
R module (*N*:: ('*b*, '*r*, '*m1*) *Module-scheme*);
R module (*MN*:: ('*c*, '*r*, '*m1*) *Module-scheme*);
R module (*MN1*::('*c*, '*r*, '*m1*) *Module-scheme*);
universal-property R M N MN f; *universal-property R M N MN1 g*⟧ ⟹
MN ≅$_R$ *MN1*
⟨*proof*⟩

Chapter 6

Construction of an abelian group

6.1 Free generated abelian group I, direct sum and direct product 2

definition
bpp :: [$'a \Rightarrow 'a \Rightarrow 'a$, $'a$, $'a$] $\Rightarrow 'a$ **where**
bpp f a b = *f a b*

definition
ipp :: [$'a \Rightarrow 'a$, $'a$] $\Rightarrow 'a$ (‹(_−/ -)› [64,65]64) **where**
$_i-$ *a* == *i a*

definition
sop :: [$'r \Rightarrow 'a \Rightarrow 'a$, $'r$, $'a$] $\Rightarrow 'a$ **where**
sop s r a = *s r a*

abbreviation
BOP :: [$'a$, $'a \Rightarrow 'a \Rightarrow 'a$, $'a$] $\Rightarrow 'a$
(‹(3-/ _+/ -)› [62,62,63]62) **where**
a $_f+$ *b* == *bpp f a b*

abbreviation
SOP :: [$'r$, $'r \Rightarrow 'a \Rightarrow 'a$, $'a$] $\Rightarrow 'a$
(‹(3-/ _· -)› [68,68,69]68) **where**
r $_s\cdot$ *a* == *sop s r a*

definition
minus-set :: [$'a \Rightarrow 'a$, $'a$ *set*] $\Rightarrow 'a$ *set* **where**
minus-set i A = $\{x.\ \exists\, y{\in}A.\ x = {}_i- y\}$

definition
pm-set :: [$'a \Rightarrow 'a$, $'a$ *set*] $\Rightarrow 'a$ *set* **where**

pm-set i A = A ∪ (minus-set i A)

definition
s-set :: [('r, 'm) *Ring-scheme*, 'r ⇒ 'a ⇒ 'a, 'a *set*] ⇒ 'a *set* **where**
s-set R s A = {x. ∃ r∈*carrier R*. ∃ a∈A. x = r $_s$· a} ∪ A

primrec *add-set* :: ['a ⇒ 'a ⇒ 'a, 'a *set*] ⇒ *nat* ⇒ 'a *set*
where
add-set-0 : *add-set f A 0* = A
| *add-set-Suc*: *add-set f A* (*Suc n*) =
{x. ∃ s∈ (*add-set f A n*). ∃ t∈ A. x = s $_f$+ t}

definition
aug-pm-set :: ['a, 'a ⇒ 'a, 'a *set*] ⇒ 'a *set* **where**
aug-pm-set z i A = {z} ∪ A ∪ (*minus-set i A*)

definition
addition-set :: ['a ⇒ 'a ⇒ 'a, 'a *set*] ⇒ 'a *set* **where**
addition-set f A = ⋃{*add-set f A n* | n. (0::*nat*)≤ n}

definition
assoc-bpp :: ['a *set*, 'a ⇒ 'a ⇒ 'a] ⇒ *bool* **where**
assoc-bpp A f ⟷
(∀ a∈(*addition-set f A*). ∀ b∈(*addition-set f A*). ∀ c∈(*addition-set f A*). (a $_f$+ b) $_f$+ c = a $_f$+ (b $_f$+ c))

definition
commute-bpp :: ['a ⇒ 'a ⇒ 'a, 'a *set*] ⇒ *bool* **where**
commute-bpp f A ⟷ (∀ x∈*addition-set f A*. ∀ y∈*addition-set f A*. x $_f$+ y = y $_f$+ x)

definition
zeroA :: ['a, 'a ⇒ 'a, 'a ⇒ 'a ⇒ 'a, 'a *set*] ⇒ 'a ⇒ *bool* **where**
zeroA z i f A z1 ⟷ (∀ x ∈ *addition-set f* (*aug-pm-set z i A*). *z1* $_f$+ x = x)

definition
inv-ipp :: ['a, 'a ⇒ 'a, 'a ⇒ 'a ⇒ 'a, 'a *set*] ⇒ *bool* **where**
inv-ipp z i f A ⟷ (∀ a∈*addition-set f* (*aug-pm-set z i A*). *zeroA z i f A* (($_i$− a) $_f$+ a))

definition
ipp-cond1 :: ['a *set*, 'a ⇒ 'a] ⇒ *bool* **where**
ipp-cond1 A i ⟷ (∀ x∈A. $_i$− ($_i$− x) = x)

definition
ipp-cond2 :: ['a, 'a *set*, 'a ⇒ 'a, 'a ⇒ 'a ⇒ 'a] ⇒ *bool* **where**
ipp-cond2 z A i f == ∀ x∈(*addition-set f* (*aug-pm-set z i A*)).
∀ y∈ (*addition-set f* (*aug-pm-set z i A*)). $_i$−(x $_f$+ y) = $_i$− y $_f$+ ($_i$− x)

definition
ipp-cond3 :: $['a, 'a \Rightarrow 'a] \Rightarrow bool$ **where**
ipp-cond3 z i $\longleftrightarrow_i -$ *z* = *z*

lemma *add-set-mono*:$A \subseteq B \Longrightarrow$ *add-set f A n* $\subseteq$ *add-set f B n*
⟨*proof*⟩

lemma *addition-inc-add*:*add-set f A n* $\subseteq$ *addition-set f A*
⟨*proof*⟩

lemma *addition-inc-add0*: *A* $\subseteq$ *addition-set f A*
⟨*proof*⟩

lemma *addition-set-mono*:$A \subseteq B \Longrightarrow$ *addition-set f A* $\subseteq$ *addition-set f B*
⟨*proof*⟩

lemma *a-in-aug-pm-set*:$a \in A \Longrightarrow a \in$ *aug-pm-set z i A*
⟨*proof*⟩

lemma *A-sub-aug-pm-set*:*A* $\subseteq$ *aug-pm-set z i A*
⟨*proof*⟩

lemma *addition-sub-aug-pm-addition*:
addition-set f A $\subseteq$ *addition-set f* (*aug-pm-set z i A*)
⟨*proof*⟩

lemma *assoc-bpp-restrict*:$[\![$ *A* $\subseteq$ *B*; *assoc-bpp B f*$]\!] \Longrightarrow$ *assoc-bpp A f*
⟨*proof*⟩

lemma *addition-assoc*:$[\![$*assoc-bpp A f*; *x* $\in$ *addition-set f A*;
y $\in$ *addition-set f A*; *z* $\in$ *addition-set f A*$]\!] \Longrightarrow$
(*x* $_f+$ *y*) $_f+$ *z* = *x* $_f+$ (*y* $_f+$ *z*)
⟨*proof*⟩

lemma *bpp-closedTr*:*assoc-bpp A f* $\Longrightarrow$
$\forall$ *x y*. *x* $\in$ *add-set f A n* $\wedge$ *y* $\in$ *add-set f A m* $\longrightarrow$
x $_f+$ *y* $\in$ *add-set f A* (*n* + *m* + *Suc 0*)
⟨*proof*⟩

lemma *bpp-closed1*:$[\![$*assoc-bpp A f*; *x* $\in$ *add-set f A n*; *y* $\in$ *add-set f A m*$]\!] \Longrightarrow$
x $_f+$ *y* $\in$ *add-set f A* (*n* + *m* + *Suc 0*)
⟨*proof*⟩
lemma *bpp-closed*:$[\![$*assoc-bpp A f*; *x* $\in$ *addition-set f A*; *y* $\in$ *addition-set f A*$]\!]$
$\Longrightarrow$ *x* $_f+$ *y* $\in$ *addition-set f A*
⟨*proof*⟩

lemma *aug-addition-inc-z*: *z* $\in$ *addition-set f* (*aug-pm-set z i A*)
⟨*proof*⟩

lemma *aug-bpp-closed*:⟦*assoc-bpp* (*aug-pm-set z i A*) *f*;
x ∈ *addition-set f* (*aug-pm-set z i A*);
y ∈ *addition-set f* (*aug-pm-set z i A*)⟧ ⟹
x $_f$+ *y* ∈ *addition-set f* (*aug-pm-set z i A*)
⟨*proof*⟩

lemma *aug-commute*:⟦*commute-bpp f* (*aug-pm-set z i A*);
x ∈ *addition-set f* (*aug-pm-set z i A*);
y ∈ *addition-set f* (*aug-pm-set z i A*)⟧ ⟹ *x* $_f$+ *y* = *y* $_f$+ *x*
⟨*proof*⟩

lemma *addition-set-inc-z*:*z* ∈ *addition-set f* (*aug-pm-set z i A*)
⟨*proof*⟩

lemma *aug-ipp-closed0*:⟦*commute-bpp f* (*aug-pm-set z i A*);
assoc-bpp (*aug-pm-set z i A*) *f*; *ipp-cond1 A i*; *ipp-cond2 z A i f*;
ipp-cond3 z i; *x* ∈ *add-set f* (*aug-pm-set z i A*) *0*⟧ ⟹
$_i$− *x* ∈ *add-set f* (*aug-pm-set z i A*) *0*
⟨*proof*⟩

lemma *aug-ipp-closedTr*:⟦*commute-bpp f* (*aug-pm-set z i A*);
assoc-bpp (*aug-pm-set z i A*) *f*; *ipp-cond1 A i*; *ipp-cond2 z A i f*;
ipp-cond3 z i⟧ ⟹
∀ *x*. *x* ∈ *add-set f* (*aug-pm-set z i A*) *n* ⟶
$_i$− *x* ∈ *add-set f* (*aug-pm-set z i A*) *n*
⟨*proof*⟩

lemma *aug-ipp-closedTr2*:⟦*commute-bpp f* (*aug-pm-set z i A*);
assoc-bpp (*aug-pm-set z i A*) *f*; *ipp-cond1 A i*; *ipp-cond2 z A i f*;
ipp-cond3 z i; *x* ∈ *add-set f* (*aug-pm-set z i A*) *n*⟧ ⟹
$_i$− *x* ∈ *add-set f* (*aug-pm-set z i A*) *n*
⟨*proof*⟩

lemma *aug-ipp-closed*:⟦*commute-bpp f* (*aug-pm-set z i A*);
assoc-bpp (*aug-pm-set z i A*) *f*; *ipp-cond1 A i*; *ipp-cond2 z A i f*;
ipp-cond3 z i; *x* ∈ *addition-set f* (*aug-pm-set z i A*)⟧ ⟹
$_i$− *x* ∈ *addition-set f* (*aug-pm-set z i A*)
⟨*proof*⟩

lemma *aug-zero-unique*:⟦*commute-bpp f* (*aug-pm-set z i A*);
z1 ∈ *addition-set f* (*aug-pm-set z i A*); *zeroA z i f A z*;
zeroA z i f A z1⟧ ⟹ *z* = *z1*
⟨*proof*⟩

lemma *inv-aug-addition*:⟦*commute-bpp f* (*aug-pm-set z i A*);
assoc-bpp (*aug-pm-set z i A*) *f*; *ipp-cond1 A i*; *ipp-cond2 z A i f*;
ipp-cond3 z i; *inv-ipp z i f A*; *commute-bpp f* (*aug-pm-set z i A*);
zeroA z i f A z⟧ ⟹
∀ *a*∈*addition-set f* (*aug-pm-set z i A*). ($_i$−*a*) $_f$+ *a* = *z*

⟨*proof*⟩

definition
fag-gen-by :: [*'a set*, *'a* ⇒ *'a* ⇒ *'a*, *'a* ⇒ *'a*, *'a*] ⇒ *'a aGroup* **where**
fag-gen-by A f i z = (|*carrier* = *addition-set f* (*aug-pm-set z i A*),
pop = λ*x*∈(*addition-set f* (*aug-pm-set z i A*)).
λ*y*∈(*addition-set f* (*aug-pm-set z i A*)). *x* $_f$+ *y*,
mop = λ*x*∈(*addition-set f* (*aug-pm-set z i A*)). $_i$− *x*, *zero* = *z*|)

lemma *fag-gen-carrier*:⟦*commute-bpp f* (*aug-pm-set z i A*);
assoc-bpp (*aug-pm-set z i A*) *f*; *ipp-cond1 A i*; *ipp-cond2 z A i f*;
ipp-cond3 z i; *inv-ipp z i f A*; *commute-bpp f* (*aug-pm-set z i A*);
zeroA z i f A z⟧ ⟹
carrier (*fag-gen-by A f i z*) = *addition-set f* (*aug-pm-set z i A*)
⟨*proof*⟩

lemma *addition-set-sub-fag-gen-carrier*:⟦*commute-bpp f* (*aug-pm-set z i A*);
assoc-bpp (*aug-pm-set z i A*) *f*; *ipp-cond1 A i*; *ipp-cond2 z A i f*;
ipp-cond3 z i; *inv-ipp z i f A*; *commute-bpp f* (*aug-pm-set z i A*);
zeroA z i f A z⟧ ⟹ *addition-set f A* ⊆ *carrier* (*fag-gen-by A f i z*)
⟨*proof*⟩

lemma *fag-aGroup*:⟦*commute-bpp f* (*aug-pm-set z i A*);
assoc-bpp (*aug-pm-set z i A*) *f*; *ipp-cond1 A i*; *ipp-cond2 z A i f*;
ipp-cond3 z i; *inv-ipp z i f A*; *commute-bpp f* (*aug-pm-set z i A*);
zeroA z i f A z⟧ ⟹ *aGroup* (*fag-gen-by A f i z*)
⟨*proof*⟩

6.2 Abelian group generated by a singleton (constructive)

definition
fag-single :: [*'a*, *'a* ⇒ *'a* ⇒ *'a*, *'a* ⇒ *'a*, *'a*] ⇒ *'a aGroup* **where**
fag-single a f i z = *fag-gen-by* {*a*} *f i z*

lemma *aug-pm-aug-pm-minus*:*ipp-cond1* {*a*} *i* ⟹
aug-pm-set z i {*a*} = *aug-pm-set z i* {$_i$− *a*}
⟨*proof*⟩

lemma *ipp-cond1-minus*:*ipp-cond1* {*a*} *i* ⟹ *ipp-cond1* {$_i$− *a*} *i*
⟨*proof*⟩

lemma *ipp-cond2-minus*:⟦*ipp-cond1* {*a*} *i*; *ipp-cond2 z* {*a*} *i f*⟧ ⟹
ipp-cond2 z {$_i$− *a*} *i f*
⟨*proof*⟩

lemma *zeroA-minus*:⟦*ipp-cond1* {*a*} *i*; *zeroA z i f* {*a*} *z1*⟧ ⟹

$$zeroA\ z\ i\ f\ \{_i-\ a\}\ z1$$

⟨*proof*⟩

lemma *inv-ipp-minus*:⟦*ipp-cond1* $\{a\}$ i; *inv-ipp* $z\ i\ f\ \{a\}$⟧ ⟹
$$inv\text{-}ipp\ z\ i\ f\ \{_i-\ a\}$$
⟨*proof*⟩

lemma *fag-single-additionTr1*:⟦*commute-bpp* f (*aug-pm-set* $z\ i\ \{a\}$);
assoc-bpp (*aug-pm-set* $z\ i\ \{a\}$) f; *ipp-cond1* $\{a\}$ i; *ipp-cond2* $z\ \{a\}\ i\ f$;
ipp-cond3 $z\ i$; *inv-ipp* $z\ i\ f\ \{a\}$; *commute-bpp* f (*aug-pm-set* $z\ i\ \{a\}$);
zeroA $z\ i\ f\ \{a\}\ z$⟧ ⟹
$\forall\, s.\ s \in add\text{-}set\ f\ \{a\}\ (Suc\ n) \longrightarrow s\ {}_f+\ {}_i-\ a \in add\text{-}set\ f\ \{a\}\ n$
⟨*proof*⟩

lemma *fag-single-additionTr2*:⟦*commute-bpp* f (*aug-pm-set* $z\ i\ \{a\}$);
assoc-bpp (*aug-pm-set* $z\ i\ \{a\}$) f; *ipp-cond1* $\{a\}$ i; *ipp-cond2* $z\ \{a\}\ i\ f$;
ipp-cond3 $z\ i$; *inv-ipp* $z\ i\ f\ \{a\}$; *commute-bpp* f (*aug-pm-set* $z\ i\ \{a\}$);
zeroA $z\ i\ f\ \{a\}\ z$; $s \in add\text{-}set\ f\ \{a\}\ 0$⟧ ⟹ $s\ {}_f+\ {}_i-\ a = z$
⟨*proof*⟩

lemma *ipp-conditions*:⟦*assoc-bpp* (*aug-pm-set* $z\ i\ \{a\}$) f; *ipp-cond1* $\{a\}$ i;
ipp-cond2 $z\ \{a\}\ i\ f$; *ipp-cond3* $z\ i$; *inv-ipp* $z\ i\ f\ \{a\}$;
commute-bpp f (*aug-pm-set* $z\ i\ \{a\}$); *zeroA* $z\ i\ f\ \{a\}\ z$⟧ ⟹
assoc-bpp (*aug-pm-set* $z\ i\ \{\ _i-\ a\}$) $f \wedge$ *ipp-cond1* $\{\ _i-\ a\}\ i \wedge$
ipp-cond2 $z\ \{\ _i-\ a\}\ i\ f \wedge$ *inv-ipp* $z\ i\ f\ \{\ _i-\ a\} \wedge$
commute-bpp f (*aug-pm-set* $z\ i\ \{\ _i-\ a\}$) $\wedge$ *zeroA* $z\ i\ f\ \{\ _i-\ a\}\ z$
⟨*proof*⟩

lemma *fag-single-additionTr3*:⟦*commute-bpp* f (*aug-pm-set* $z\ i\ \{a\}$);
assoc-bpp (*aug-pm-set* $z\ i\ \{a\}$) f; *ipp-cond1* $\{a\}$ i; *ipp-cond2* $z\ \{a\}\ i\ f$;
ipp-cond3 $z\ i$; *inv-ipp* $z\ i\ f\ \{a\}$; *commute-bpp* f (*aug-pm-set* $z\ i\ \{a\}$);
zeroA $z\ i\ f\ \{a\}\ z$; $s \in add\text{-}set\ f\ \{_i-\ a\}\ n$⟧ ⟹
$s\ {}_f+\ {}_i-\ a \in add\text{-}set\ f\ \{_i-\ a\}\ (Suc\ n)$
⟨*proof*⟩

lemma *fag-single-elemTr*:⟦*commute-bpp* f (*aug-pm-set* $z\ i\ \{a\}$);
assoc-bpp (*aug-pm-set* $z\ i\ \{a\}$) f; *ipp-cond1* $\{a\}$ i; *ipp-cond2* $z\ \{a\}\ i\ f$;
ipp-cond3 $z\ i$; *inv-ipp* $z\ i\ f\ \{a\}$; *commute-bpp* f (*aug-pm-set* $z\ i\ \{a\}$);
zeroA $z\ i\ f\ \{a\}\ z$⟧ ⟹
$\forall\, x.\ x \in add\text{-}set\ f\ (aug\text{-}pm\text{-}set\ z\ i\ \{a\})\ n \longrightarrow$
$(\exists\, n1.\ x \in add\text{-}set\ f\ \{a\}\ n1) \vee (\exists\, m1.\ x \in add\text{-}set\ f\ \{_i-\ a\}\ m1) \vee x = z$
⟨*proof*⟩

lemma *fag-single-elem*:⟦*commute-bpp* f (*aug-pm-set* $z\ i\ \{a\}$);
assoc-bpp (*aug-pm-set* $z\ i\ \{a\}$) f; *ipp-cond1* $\{a\}$ i; *ipp-cond2* $z\ \{a\}\ i\ f$;
ipp-cond3 $z\ i$; *inv-ipp* $z\ i\ f\ \{a\}$; *commute-bpp* f (*aug-pm-set* $z\ i\ \{a\}$);
zeroA $z\ i\ f\ \{a\}\ z$; $x \in addition\text{-}set\ f\ (aug\text{-}pm\text{-}set\ z\ i\ \{a\})$⟧ ⟹
$(\exists\, n1.\ x \in add\text{-}set\ f\ \{a\}\ n1) \vee (\exists\, m1.\ x \in add\text{-}set\ f\ \{_i-\ a\}\ m1) \vee x = z$

⟨*proof*⟩

lemma *add-set-single1Tr*:⟦*commute-bpp f* (*aug-pm-set z i* {*a*});
assoc-bpp (*aug-pm-set z i* {*a*}) *f*; *ipp-cond1* {*a*} *i*; *ipp-cond2 z* {*a*} *i f*;
ipp-cond3 z i; *inv-ipp z i f* {*a*}; *commute-bpp f* (*aug-pm-set z i* {*a*});
zeroA z i f {*a*} *z*⟧ ⟹
∀ *x y*. *x* ∈ *add-set f* {*a*} *n* ∧ *y* ∈ *add-set f* {*a*} *n* ⟶ *x* = *y*
⟨*proof*⟩

lemma *add-set-single-nonempty1*:⟦*commute-bpp f* (*aug-pm-set z i* {*a*});
assoc-bpp (*aug-pm-set z i* {*a*}) *f*; *ipp-cond1* {*a*} *i*; *ipp-cond2 z* {*a*} *i f*;
ipp-cond3 z i; *inv-ipp z i f* {*a*}; *commute-bpp f* (*aug-pm-set z i* {*a*});
zeroA z i f {*a*} *z*⟧ ⟹ ∃ *x*. *x*∈*add-set f* {*a*} *n*
⟨*proof*⟩

lemma *add-set-single-nonempty2*:⟦*commute-bpp f* (*aug-pm-set z i* {*a*});
assoc-bpp (*aug-pm-set z i* {*a*}) *f*; *ipp-cond1* {*a*} *i*; *ipp-cond2 z* {*a*} *i f*;
ipp-cond3 z i; *inv-ipp z i f* {*a*}; *commute-bpp f* (*aug-pm-set z i* {*a*});
zeroA z i f {*a*} *z*⟧ ⟹ ∃ *x*. *x*∈*add-set f* {$_i-$ *a*} *n*
⟨*proof*⟩

lemma *add-set-single1*:⟦*commute-bpp f* (*aug-pm-set z i* {*a*});
assoc-bpp (*aug-pm-set z i* {*a*}) *f*; *ipp-cond1* {*a*} *i*; *ipp-cond2 z* {*a*} *i f*;
ipp-cond3 z i; *inv-ipp z i f* {*a*}; *commute-bpp f* (*aug-pm-set z i* {*a*});
zeroA z i f {*a*} *z*; *x* ∈ *add-set f* {*a*} *n*; *y* ∈ *add-set f* {*a*} *n*⟧ ⟹ *x* = *y*
⟨*proof*⟩

lemma *add-set-single2*:⟦*commute-bpp f* (*aug-pm-set z i* {*a*});
assoc-bpp (*aug-pm-set z i* {*a*}) *f*; *ipp-cond1* {*a*} *i*; *ipp-cond2 z* {*a*} *i f*;
ipp-cond3 z i; *inv-ipp z i f* {*a*}; *commute-bpp f* (*aug-pm-set z i* {*a*});
zeroA z i f {*a*} *z*; *x* ∈ *add-set f* {$_i-$ *a*} *n*; *y* ∈ *add-set f* {$_i-$ *a*} *n*⟧ ⟹
x = *y*
⟨*proof*⟩

lemma *fag-single-additionTr4*:⟦*commute-bpp f* (*aug-pm-set z i* {*a*});
assoc-bpp (*aug-pm-set z i* {*a*}) *f*; *ipp-cond1* {*a*} *i*; *ipp-cond2 z* {*a*} *i f*;
ipp-cond3 z i; *inv-ipp z i f* {*a*}; *commute-bpp f* (*aug-pm-set z i* {*a*});
zeroA z i f {*a*} *z* ⟧ ⟹
∀ *s t*. *s* ∈ *add-set f* {*a*} *n* ∧ *t* ∈ *add-set f* {$_i-$ *a*} *n*⟶ *s* $_f+$ *t* = *z*
⟨*proof*⟩

lemma *fag-single-additionTr4-1*:⟦*commute-bpp f* (*aug-pm-set z i* {*a*});
assoc-bpp (*aug-pm-set z i* {*a*}) *f*; *ipp-cond1* {*a*} *i*; *ipp-cond2 z* {*a*} *i f*;
ipp-cond3 z i; *inv-ipp z i f* {*a*}; *commute-bpp f* (*aug-pm-set z i* {*a*});
zeroA z i f {*a*} *z*;*s* ∈ *add-set f* {*a*} *n*; *t* ∈ *add-set f* {$_i-$ *a*} *n* ⟧ ⟹
s $_f+$ *t* = *z*
⟨*proof*⟩

lemma *fag-single-additionTr5*:⟦*assoc-bpp* (*aug-pm-set z i* {*a*}) *f*;

ipp-cond1 $\{a\}$ *i*; *ipp-cond2 z* $\{a\}$ *i f*; *ipp-cond3 z i*; *inv-ipp z i f* $\{a\}$;
commute-bpp f (*aug-pm-set z i* $\{a\}$); *zeroA z i f* $\{a\}$ *z*$\rrbracket \Longrightarrow$
$\forall$ *m*. *m* < *Suc n* $\longrightarrow$ (*THE x*. *x* $\in$ *add-set f* $\{a\}$ (*Suc n*)) $_f+$
(*THE x*. *x* $\in$ *add-set f* $\{_i-$ *a*$\}$ *m*) = (*THE x*. *x* $\in$ *add-set f* $\{a\}$ (*n* − *m*))
⟨*proof*⟩

lemma *fag-single-additionTr5-1*:$\llbracket$*assoc-bpp* (*aug-pm-set z i* $\{a\}$) *f*;
ipp-cond1 $\{a\}$ *i*; *ipp-cond2 z* $\{a\}$ *i f*; *ipp-cond3 z i*; *inv-ipp z i f* $\{a\}$;
commute-bpp f (*aug-pm-set z i* $\{a\}$); *zeroA z i f* $\{a\}$ *z*; *m* < *Suc n*$\rrbracket \Longrightarrow$
(*THE x*. *x* $\in$ *add-set f* $\{a\}$ (*Suc n*)) $_f+$ (*THE x*. *x* $\in$ *add-set f* $\{_i-$ *a*$\}$ *m*) =
(*THE x*. *x* $\in$ *add-set f* $\{a\}$ (*n* − *m*))
⟨*proof*⟩

lemma *fag-single-additionTr5-2*:$\llbracket$*assoc-bpp* (*aug-pm-set z i* $\{a\}$) *f*;
ipp-cond1 $\{a\}$ *i*; *ipp-cond2 z* $\{a\}$ *i f*; *ipp-cond3 z i*; *inv-ipp z i f* $\{a\}$;
commute-bpp f (*aug-pm-set z i* $\{a\}$); *zeroA z i f* $\{a\}$ *z*; *n* < *Suc m*$\rrbracket \Longrightarrow$
(*THE x*. *x* $\in$ *add-set f* $\{_i-$ *a*$\}$ (*Suc m*)) $_f+$ (*THE x*. *x* $\in$ *add-set f* $\{a\}$ *n*) =
(*THE x*. *x* $\in$ *add-set f* $\{_i-$ *a*$\}$ (*m* − *n*))
⟨*proof*⟩

definition
free-gen-condition :: [*'a* ⇒ *'a* ⇒ *'a*, *'a* ⇒ *'a*, *'a*, *'a*] ⇒ *bool* **where**
free-gen-condition f i a z ⟷ (∀ *n*. *z* ∉ *add-set f* $\{a\}$ *n*)

definition
fg-elem-single :: [*'a* ⇒ *'a* ⇒ *'a*, *'a* ⇒ *'a*, *'a*, *'a*] ⇒ *int* ⇒ *'a* **where**
fg-elem-single f i a z n = (*if 0* = *n then z else*
(*if 0* < *n then* (*THE x*. *x* ∈ (*add-set f* $\{a\}$ (*nat* (*n* − *1*))))
else (*THE x*. *x* ∈ (*add-set f* $\{_i-$ *a*$\}$ (*nat* (− *n* − *1*))))))

abbreviation
FGELEMSNGLE (‹(*5-*⊙ *-*,*-*,*-*)› [*99*,*98*,*98*,*98*,*98*]*99*) **where**
n⊙ $a_{f,i,z}$ == *fg-elem-single f i a z n*

lemma *single-addition-pm-mem*:$\llbracket$*assoc-bpp* (*aug-pm-set z i* $\{a\}$) *f*;
ipp-cond1 $\{a\}$ *i*; *ipp-cond2 z* $\{a\}$ *i f*; *ipp-cond3 z i*; *inv-ipp z i f* $\{a\}$;
commute-bpp f (*aug-pm-set z i* $\{a\}$); *zeroA z i f* $\{a\}$ *z*$\rrbracket \Longrightarrow$
(*n*⊙ $a_{f,i,z}$) ∈ *addition-set f* (*aug-pm-set z i* $\{a\}$)
⟨*proof*⟩

lemma *assoc-aug-assoc*:*assoc-bpp* (*aug-pm-set z i* $\{a\}$) *f* ⟹ *assoc-bpp* $\{a\}$ *f*
⟨*proof*⟩

lemma *single-addition-posTr*:$\llbracket$*commute-bpp f* (*aug-pm-set z i* $\{$(*a*::*'a*)$\}$);
assoc-bpp (*aug-pm-set z i* $\{a\}$) *f*; *ipp-cond1* $\{a\}$ *i*; *ipp-cond2 z* $\{a\}$ *i f*;
ipp-cond3 z i; *inv-ipp z i f* $\{a\}$; *commute-bpp f* (*aug-pm-set z i* $\{a\}$);
zeroA z i f $\{a\}$ *z*; *0* < (*n*::*int*); *0* < (*m*::*int*)$\rrbracket \Longrightarrow$
(*THE x*. *x* $\in$ *add-set f* $\{a\}$ (*nat* (*n* − *1*))) $_f+$

$(THE\ x.\ x \in add\text{-}set\ f\ \{a\}\ (nat\ (m - 1))) =$
$(THE\ x.\ x \in add\text{-}set\ f\ \{a\}\ (nat\ (n + m - 1)))$

⟨*proof*⟩

lemma *single-addition-pos*:⟦*commute-bpp f* (*aug-pm-set z i* {(*a*::′*a*)});
assoc-bpp (*aug-pm-set z i* {*a*}) *f*; *ipp-cond1* {*a*} *i*; *ipp-cond2 z* {*a*} *i f*;
ipp-cond3 z i; *inv-ipp z i f* {*a*}; *commute-bpp f* (*aug-pm-set z i* {*a*});
zeroA z i f {*a*} *z*; *0* < (*n*::*int*); *0* < (*m*::*int*)⟧ ⟹
$(n\bigodot a_{f,i,z})\ {}_f{+}\ (m\bigodot a_{f,i,z}) = (n + m)\bigodot a_{f,i,z}$

⟨*proof*⟩

lemma *single-addition-neg*:⟦*commute-bpp f* (*aug-pm-set z i* {(*a*::′*a*)});
assoc-bpp (*aug-pm-set z i* {*a*}) *f*; *ipp-cond1* {*a*} *i*; *ipp-cond2 z* {*a*} *i f*;
ipp-cond3 z i; *inv-ipp z i f* {*a*}; *commute-bpp f* (*aug-pm-set z i* {*a*});
zeroA z i f {*a*} *z*; (*n*::*int*) < *0*; (*m*::*int*) < *0* ⟧ ⟹
$(n\bigodot a_{f,i,z})\ {}_f{+}\ (m\bigodot a_{f,i,z}) = (n + m)\bigodot a_{f,i,z}$

⟨*proof*⟩

lemma *single-addition-zero*:⟦*commute-bpp f* (*aug-pm-set z i* {(*a*::′*a*)});
assoc-bpp (*aug-pm-set z i* {*a*}) *f*; *ipp-cond1* {*a*} *i*; *ipp-cond2 z* {*a*} *i f*;
ipp-cond3 z i; *inv-ipp z i f* {*a*}; *commute-bpp f* (*aug-pm-set z i* {*a*});
zeroA z i f {*a*} *z*⟧ ⟹ $0\bigodot a_{f,i,z} = z$

⟨*proof*⟩

lemma *s-a-p-1*:⟦*assoc-bpp* (*aug-pm-set z i* {*a*}) *f*; *ipp-cond1* {*a*} *i*;
ipp-cond2 z {*a*} *i f*; *ipp-cond3 z i*; *inv-ipp z i f* {*a*};
commute-bpp f (*aug-pm-set z i* {*a*}); *zeroA z i f* {*a*} *z*;
m < *0*; *0* < *n*⟧ ⟹ $(n\bigodot a_{f,i,z})\ {}_f{+}\ (m\bigodot a_{f,i,z}) = (n + m)\bigodot a_{f,i,z}$

⟨*proof*⟩

lemma *single-addition-pm*:⟦*commute-bpp f* (*aug-pm-set z i* {(*a*::′*a*)});
assoc-bpp (*aug-pm-set z i* {*a*}) *f*; *ipp-cond1* {*a*} *i*; *ipp-cond2 z* {*a*} *i f*;
ipp-cond3 z i; *inv-ipp z i f* {*a*}; *commute-bpp f* (*aug-pm-set z i* {*a*});
zeroA z i f {*a*} *z*⟧ ⟹ $(n\bigodot a_{f,i,z})\ {}_f{+}\ (m\bigodot a_{f,i,z}) = (n + m)\bigodot a_{f,i,z}$

⟨*proof*⟩

lemma *single-inv*:⟦*commute-bpp f* (*aug-pm-set z i* {(*a*::′*a*)});
assoc-bpp (*aug-pm-set z i* {*a*}) *f*; *ipp-cond1* {*a*} *i*; *ipp-cond2 z* {*a*} *i f*;
ipp-cond3 z i; *inv-ipp z i f* {*a*}; *commute-bpp f* (*aug-pm-set z i* {*a*});
zeroA z i f {*a*} *z*⟧ ⟹ $\ {}_i{-}\ (m\bigodot a_{f,i,z}) = (-m)\bigodot a_{f,i,z}$

⟨*proof*⟩

lemma *free-ag-single*:⟦*commute-bpp f* (*aug-pm-set z i* {*a*});
assoc-bpp (*aug-pm-set z i* {*a*}) *f*; *ipp-cond1* {*a*} *i*; *ipp-cond2 z* {*a*} *i f*;
ipp-cond3 z i; *inv-ipp z i f* {*a*}; *commute-bpp f* (*aug-pm-set z i* {*a*});
zeroA z i f {*a*} *z*; *free-gen-condition f i a z*; *n* ≠ *m*⟧ ⟹
$(n\bigodot a_{f,i,z}) \neq (m\bigodot a_{f,i,z})$

⟨*proof*⟩

definition
fags-cond :: [$'a \Rightarrow 'a \Rightarrow 'a$, $'a$, $'a \Rightarrow 'a$, $'a$] $\Rightarrow$ *bool* **where**
fags-cond f z i a $\longleftrightarrow$ *commute-bpp f* (*aug-pm-set z i* {*a*}) $\wedge$
assoc-bpp (*aug-pm-set z i* {*a*}) *f* $\wedge$ *ipp-cond1* {*a*} *i* $\wedge$
ipp-cond2 z {*a*} *i f* $\wedge$ *ipp-cond3 z i* $\wedge$ *inv-ipp z i f* {*a*} $\wedge$
commute-bpp f (*aug-pm-set z i* {*a*}) $\wedge$ *zeroA z i f* {*a*} *z* $\wedge$
free-gen-condition f i a z

lemma *fag-single-free*:$[\![$*fags-cond f z i a*; $n \neq m]\!] \Longrightarrow (n\bigodot a_{f,i,z}) \neq (m\bigodot a_{f,i,z})$
⟨*proof*⟩

lemma *fag-single-free1*:$[\![$*fags-cond f z i a*;$(n\bigodot a_{f,i,z}) = (m\bigodot a_{f,i,z})]\!] \Longrightarrow n = m$
⟨*proof*⟩

definition
fags-carr :: [$'a \Rightarrow 'a \Rightarrow 'a$, $'a$, $'a \Rightarrow 'a$, $'a$] $\Rightarrow$ $'a$ *set* **where**
fags-carr f z i a = $\{x.\ \exists\, n.\ x = n\bigodot a_{f,i,z}\}$

definition
fags-bpp :: [$'a \Rightarrow 'a \Rightarrow 'a$, $'a$, $'a \Rightarrow 'a$, $'a$] $\Rightarrow 'a \Rightarrow 'a \Rightarrow 'a$ **where**
fags-bpp f z i a = ($\lambda x{\in}$(*fags-carr f z i a*). $\lambda y{\in}$(*fags-carr f z i a*).
((*THE n*. $x = n\bigodot a_{f,i,z}$) + (*THE m*. $y = m\bigodot a_{f,i,z}$))$\bigodot a_{f,i,z}$)

definition
fags-ipp :: [$'a \Rightarrow 'a \Rightarrow 'a$, $'a$, $'a \Rightarrow 'a$, $'a$] $\Rightarrow 'a \Rightarrow 'a$ **where**
fags-ipp f z i a = ($\lambda x{\in}$(*fags-carr f z i a*).
(− (*THE n*. $x = n\bigodot a_{f,i,z}$))$\bigodot a_{f,i,z}$)

lemma *fags-mem*:*fags-cond f z i a* $\Longrightarrow (n\bigodot a_{f,i,z}) \in$ *fags-carr f z i a*
⟨*proof*⟩

lemma *fags-ippTr*:*fags-cond f z i a* $\Longrightarrow$
fags-ipp f z i a $(n\bigodot a_{f,i,z}) = (-\ n)\bigodot a_{f,i,z}$
⟨*proof*⟩

lemma *fags-bppTr*:*fags-cond f z i a* $\Longrightarrow$
fags-bpp f z i a $(n\bigodot a_{f,i,z})$ $(m\bigodot a_{f,i,z}) = (n + m)\bigodot a_{f,i,z}$
⟨*proof*⟩

definition
fags :: [$'a \Rightarrow 'a \Rightarrow 'a$, $'a$, $'a \Rightarrow 'a$, $'a$] $\Rightarrow$ $'a$ *aGroup* **where**
fags f z i a = (|*carrier* = *fags-carr f z i a*,
pop = *fags-bpp f z i a*,
mop = *fags-ipp f z i a*, *zero* = *z*|)

lemma *fags-ag*:*fags-cond f z i a* $\Longrightarrow$ *aGroup* (*fags f z i a*)
⟨*proof*⟩

6.3 Abelian Group generated by one element II (nonconstructive)

definition
ag-single-gen :: $[('a, 'm)\ aGroup\text{-}scheme, 'a] \Rightarrow bool$ **where**
ag-single-gen $A\ a \longleftrightarrow aGroup\ A \wedge carrier\ A = \bigcap \{H.\ asubGroup\ A\ H \wedge a \in H\}$

primrec *aSum* :: $[('a, 'm)\ aGroup\text{-}scheme, nat, 'a] \Rightarrow 'a$ **where**
aSum-0: $aSum\ A\ 0\ a = \mathbf{0}_A$
| *aSum-Suc*: $aSum\ A\ (Suc\ n)\ a = aSum\ A\ n\ a \pm_A a$

definition
sprod-n-a ::$[('a, 'm)\ aGroup\text{-}scheme, int, 'a] \Rightarrow 'a$ **where**
sprod-n-a $A\ n\ x = (if\ 0 \leq n\ then\ (aSum\ A\ (nat\ n)\ x)$
$else\ (aSum\ A\ (nat\ (-\ n))\ (-_{a\,A}\ x)))$

abbreviation
SPRODNA $(\langle(3\text{-}\triangleright\text{-}\_)\rangle\ [95,95,96]95)$ **where**
$n \triangleright a_A == sprod\text{-}n\text{-}a\ A\ n\ a$

lemma (**in** *aGroup*) *asum-mem*:$a \in carrier\ A \Longrightarrow aSum\ A\ n\ a \in carrier\ A$
⟨*proof*⟩

lemma (**in** *aGroup*) *nt-mem0*:$a \in carrier\ A \Longrightarrow n \triangleright a_A \in carrier\ A$
⟨*proof*⟩

lemma (**in** *aGroup*) *nt-zero0*:$a \in carrier\ A \Longrightarrow 0 \triangleright a_A = \mathbf{0}$
⟨*proof*⟩

lemma (**in** *aGroup*) *nt-1*:$a \in carrier\ A \Longrightarrow 1 \triangleright a_A = a$
⟨*proof*⟩

lemma (**in** *aGroup*) *asumTr*:$a \in carrier\ A \Longrightarrow$
$aSum\ A\ (n + m)\ a = aSum\ A\ n\ a \pm (aSum\ A\ m\ a)$
⟨*proof*⟩

lemma (**in** *aGroup*) *aSum-zero*:$a \in carrier\ A \Longrightarrow aSum\ A\ n\ \mathbf{0} = \mathbf{0}$
⟨*proof*⟩

lemma (**in** *aGroup*) *agsum-add1p*:$[\![\ a \in carrier\ A;\ 0 \leq n;\ 0 \leq m]\!] \Longrightarrow$
$(n + m) \triangleright a_A = n \triangleright a_A \pm (m \triangleright a_A)$
⟨*proof*⟩

lemma (**in** *aGroup*) *agsum-add1m*:$[\![\ a \in carrier\ A;\ n < 0;\ m < 0]\!] \Longrightarrow$
$(n + m) \triangleright a_A = n \triangleright a_A \pm (m \triangleright a_A)$
⟨*proof*⟩

lemma (**in** *aGroup*) *agsum-add2Tr*:$a \in carrier\ A \Longrightarrow$
$\mathbf{0} = aSum\ A\ n\ a \pm (aSum\ A\ n\ (-_a\ a))$
⟨*proof*⟩

lemma (**in** *aGroup*) *agsum-add2p*:$[\![a \in carrier\ A;\ 0 \leq n]\!] \Longrightarrow$
$\mathbf{0} = n{\triangleright}a_A \pm ((-n){\triangleright}a_A)$
⟨*proof*⟩

lemma (**in** *aGroup*) *agsum-add2m*:$[\![a \in carrier\ A;\ n < 0]\!] \Longrightarrow$
$\mathbf{0} = n{\triangleright}a_A \pm ((-n){\triangleright}a_A)$
⟨*proof*⟩

lemma (**in** *aGroup*) *agsum-add3pm*:$[\![a \in carrier\ A;\ 0 < n;\ m < 0]\!] \Longrightarrow$
$(n + m){\triangleright}a_A = n{\triangleright}a_A \pm (m{\triangleright}a_A)$
⟨*proof*⟩

lemma (**in** *aGroup*) *agsum-add3mp*:$[\![a \in carrier\ A;\ n < 0;\ 0 < m]\!] \Longrightarrow$
$(n + m){\triangleright}a_A = n{\triangleright}a_A \pm (m{\triangleright}a_A)$
⟨*proof*⟩

lemma (**in** *aGroup*) *nt-sum0*:$[\![a \in carrier\ A]\!] \Longrightarrow (n + m){\triangleright}a_A = n{\triangleright}a_A \pm (m{\triangleright}a_A)$
⟨*proof*⟩

lemma (**in** *aGroup*) *nt-inv0*:$a \in carrier\ A \Longrightarrow -_a\ (n{\triangleright}a_A) = (-\ n){\triangleright}a_A$
⟨*proof*⟩

lemma (**in** *aGroup*) *m-x-asum*:$[\![a \in carrier\ A;\ b \in carrier\ A]\!]$
$\Longrightarrow aSum\ A\ m\ (a \pm b) = (aSum\ A\ m\ a) \pm (aSum\ A\ m\ b)$
⟨*proof*⟩

lemma (**in** *aGroup*) *asum-multTr-pp*:$a \in carrier\ A \Longrightarrow$
$aSum\ A\ m\ (aSum\ A\ n\ a) = aSum\ A\ (m * n)\ a$
⟨*proof*⟩

lemma (**in** *aGroup*) *nt-mult-pp*:$[\![a \in carrier\ A;\ 0 \leq m;\ 0 \leq n]\!]$
$\Longrightarrow m{\triangleright}(n{\triangleright}a_A)_A = (m * n){\triangleright}a_A$
⟨*proof*⟩

lemma (**in** *aGroup*) *asum-multTr-pm*:$[\![a \in carrier\ A;\ 0 \leq m;\ n < 0]\!] \Longrightarrow$
$aSum\ A\ (nat\ m)\ (aSum\ A\ (nat\ (-\ n))\ (-_a\ a)) =$
$aSum\ A\ (nat\ (m * (-\ n)))\ (-_a\ a)$
⟨*proof*⟩

lemma (**in** *aGroup*) *nt-mult-pm*:$[\![a \in carrier\ A;\ 0 \leq m;\ n < 0]\!] \Longrightarrow$
$m{\triangleright}(n{\triangleright}a_A)_A = (m * n){\triangleright}a_A$
⟨*proof*⟩

lemma (**in** *aGroup*) *asum-multTr-mp*:$[\![a \in carrier\ A;\ m < 0;\ 0 \leq n]\!] \Longrightarrow$
$aSum\ A\ (nat\ (-m))(-_a\ (aSum\ A\ (nat\ n)\ a)) = aSum\ A\ (nat\ ((-\ m) * n))\ (-_a$

a)
⟨*proof*⟩

lemma (**in** *aGroup*) *nt-mult-mp*:⟦$a \in$ *carrier* A; $m < 0$; $0 \leq n$⟧ ⟹
$m{\triangleright}(n{\triangleright}a_A)_A = (m * n){\triangleright}a_A$
⟨*proof*⟩

lemma (**in** *aGroup*) *asum-multTr-mm*:⟦$a \in$ *carrier* A; $m < 0$; $n < 0$⟧ ⟹
aSum A (*nat* $(-m)$)$(-_a$ (*aSum* A (*nat* $(-\ n)$) $(-_a\ a)))$ =
aSum A (*nat* $((-\ m) * (-\ n))$) a
⟨*proof*⟩

lemma (**in** *aGroup*) *nt-mult-mm*:⟦ $a \in$ *carrier* A; $m < 0$; $n < 0$⟧ ⟹
$m{\triangleright}(n{\triangleright}a_A)_A = (m * n){\triangleright}a_A$
⟨*proof*⟩

lemma (**in** *aGroup*) *nt-mult-assoc0*:$a \in$ *carrier* $A \Longrightarrow m{\triangleright}n{\triangleright}a_{A\,A} = (m * n){\triangleright}a_A$
⟨*proof*⟩

lemma (**in** *aGroup*) *single-gen-carrTr*:$a \in$ *carrier* A ⟹
asubGroup A $\{x.\ \exists\, n.\ x = (n{\triangleright}a_A)\}$
⟨*proof*⟩

lemma (**in** *aGroup*) *ag-single-inc-a*:*ag-single-gen* A $a \Longrightarrow a \in$ *carrier* A
⟨*proof*⟩

lemma (**in** *aGroup*) *single-gen*:*ag-single-gen* A a ⟹
carrier $A = \{g.\ \exists\, n.\ g = (n{\triangleright}a_A)\}$
⟨*proof*⟩

definition
single-gen-free :: [('a, 'm) *aGroup-scheme*, 'a] ⇒ *bool* **where**
single-gen-free A a == $\forall\, n.\ n \neq 0 \longrightarrow \mathbf{0}_A \neq n{\triangleright}a_A$

definition
sfg :: [('a, 'm) *aGroup-scheme*, 'a] ⇒ *bool* **where**
sfg A $a \longleftrightarrow$ *ag-single-gen* A $a \wedge$ *single-gen-free* A a

lemma (**in** *aGroup*) *single-gen-free-neg*:⟦*sfg* A a; $n{\triangleright}a_A = \mathbf{0}$⟧ ⟹ $n = 0$
⟨*proof*⟩

lemma (**in** *aGroup*) *sfg-G-inc-a*:*sfg* A $a \Longrightarrow a \in$ *carrier* A
⟨*proof*⟩

lemma *sfg-agroup*:*sfg* A $a \Longrightarrow$ *aGroup* A
⟨*proof*⟩

lemma (**in** *aGroup*) *mem-G-nt*:⟦*sfg* A a; $x \in$ *carrier* A⟧ ⟹ $\exists\, n.\ x = n{\triangleright}a_A$

⟨*proof*⟩

lemma (**in** *aGroup*) *nt-mem*:*sfg A a* ⟹ $n \triangleright a_A \in$ *carrier A*
⟨*proof*⟩

lemma (**in** *aGroup*) *nt-zero*:*sfg A a* ⟹ $0 \triangleright a_A = \mathbf{0}$
⟨*proof*⟩

lemma (**in** *aGroup*) *nt-sum*:*sfg A a* ⟹ $(n + m) \triangleright a_A = n \triangleright a_A \pm (m \triangleright a_A)$
⟨*proof*⟩

lemma (**in** *aGroup*) *nt-inv*:*sfg A a* ⟹ $-_a(n \triangleright a_A) = (-\ n) \triangleright a_A$
⟨*proof*⟩

lemma (**in** *aGroup*) *nt-mult-assoc*:*sfg A a* ⟹ $m \triangleright n \triangleright a_{A\,A} = (m * n) \triangleright a_A$
⟨*proof*⟩

lemma (**in** *aGroup*) *sfg-free*:⟦*sfg A a*; $n \neq m$ ⟧ ⟹ $n \triangleright a_A \neq (m \triangleright a_A)$
⟨*proof*⟩

lemma (**in** *aGroup*) *sfg-free-inj*:⟦*sfg A a*; $n \triangleright a_A = (m \triangleright a_A)$ ⟧ ⟹ $n = m$
⟨*proof*⟩

6.4 Free Generated Modules (constructive)

definition
sop-one::[('*r*, '*m*) *Ring-scheme*, '*r* ⇒ '*a* ⇒ '*a*, '*a set*] ⇒ *bool* **where**
sop-one R s A ⟷ (∀ *x*∈*A*. $(1_{r\,R})\ _s\cdot\ x = x$)

definition
sop-assoc :: [('*r*, '*m*) *Ring-scheme*, '*r* ⇒ '*a* ⇒ '*a*, '*a set*] ⇒ *bool* **where**
sop-assoc R s A ⟷ (∀ *a*∈*carrier R*. ∀ *b*∈*carrier R*. ∀ *x*∈*A*.
$(a\ \cdot_{r\,R}\ b)\ _s\cdot\ x = a\ _s\cdot\ (b\ _s\cdot\ x)$)

definition
sop-inv :: [('*r*, '*m*) *Ring-scheme*, '*r* ⇒ '*a* ⇒ '*a*, '*a* ⇒ '*a*, '*a set*]
⇒ *bool* **where**
sop-inv R s i A ⟷ (∀ *r*∈*carrier R*. ∀ *x*∈*A*. $r\ _s\cdot\ (_i-\ x) = (-_{a\,R}\ r)\ _s\cdot\ x$)

definition
sop-distr1 :: [('*r*, '*m*) *Ring-scheme*, '*r* ⇒ '*a* ⇒ '*a*, '*a* ⇒ '*a* ⇒ '*a*,
'*a* ⇒ '*a*, '*a set*, '*a*] ⇒ *bool* **where**
sop-distr1 R s f i A z ⟷ (∀ *a*∈*carrier R*. ∀ *b*∈*carrier R*.
∀ *x*∈(*aug-pm-set z i A*). $(a \pm_R b)\ _s\cdot\ x = (a\ _s\cdot\ x)\ _f+\ (b\ _s\cdot\ x)$)

definition
sop-distr2 :: [('*r*, '*m*) *Ring-scheme*, '*r* ⇒ '*a* ⇒ '*a*, '*a* ⇒ '*a* ⇒ '*a*,
'*a* ⇒ '*a*, '*a set*, '*a*] ⇒ *bool* **where**
sop-distr2 R s f i A z ⟷ (∀ *a*∈*carrier R*.

$\forall\, x{\in}addition\text{-}set\ f\ (aug\text{-}pm\text{-}set\ z\ i\ A).$
$\forall\, y{\in}addition\text{-}set\ f\ (aug\text{-}pm\text{-}set\ z\ i\ A).$
$a\ {}_s{\cdot}\ (x\ {}_f{+}\ y) = (a\ {}_s{\cdot}\ x)\ {}_f{+}\ (a\ {}_s{\cdot}\ y))$

definition
$sop\text{-}z :: [('r, 'm)\ Ring\text{-}scheme, 'r \Rightarrow 'a \Rightarrow 'a, 'a] \Rightarrow bool$ **where**
$sop\text{-}z\ R\ s\ z \longleftrightarrow (\forall\, r{\in}carrier\ R.\ r\ {}_s{\cdot}\ z = z)$

definition
$fgmodule :: [('r, 'm)\ Ring\text{-}scheme, 'a\ set, 'a, 'a \Rightarrow 'a, 'a \Rightarrow 'a \Rightarrow 'a,$
$'r \Rightarrow 'a \Rightarrow 'a] \Rightarrow ('a, 'r)\ Module$ **where**
$fgmodule\ R\ A\ z\ i\ f\ s =$
$(\!|carrier = addition\text{-}set\ f\ (aug\text{-}pm\text{-}set\ z\ i\ (s\text{-}set\ R\ s\ A)),$
$pop = \lambda x{\in}addition\text{-}set\ f\ (aug\text{-}pm\text{-}set\ z\ i\ (s\text{-}set\ R\ s\ A)).$
$\lambda y{\in}addition\text{-}set\ f\ (aug\text{-}pm\text{-}set\ z\ i\ (s\text{-}set\ R\ s\ A)).\ x\ {}_f{+}\ y,$
$mop = \lambda x{\in}addition\text{-}set\ f\ (aug\text{-}pm\text{-}set\ z\ i\ (s\text{-}set\ R\ s\ A)).\ {}_i{-}\ x,$
$zero = z,$
$sprod = \lambda r{\in}carrier\ R.$
$\lambda x{\in}addition\text{-}set\ f\ (aug\text{-}pm\text{-}set\ z\ i\ (s\text{-}set\ R\ s\ A)).\ r\ {}_s{\cdot}\ x\ |\!)$

lemma $fgmodule\text{-}carr{:}carrier\ (fgmodule\ R\ A\ z\ i\ f\ s) =$
$addition\text{-}set\ f\ (aug\text{-}pm\text{-}set\ z\ i\ (s\text{-}set\ R\ s\ A))$
⟨*proof*⟩

lemma $a\text{-}in\text{-}s\text{-}set{:}a \in A \Longrightarrow a \in s\text{-}set\ R\ s\ A$
⟨*proof*⟩

lemma (**in** *Ring*) $ra\text{-}in\text{-}s\text{-}set{:}[\![r \in carrier\ R;\ a \in A]\!] \Longrightarrow r\ {}_s{\cdot}\ a \in s\text{-}set\ R\ s\ A$
⟨*proof*⟩

lemma $in\text{-}aug\text{-}pm\text{-}set{:}$
$x \in aug\text{-}pm\text{-}set\ z\ i\ A = (x = z \vee x \in A \vee x \in minus\text{-}set\ i\ A)$
⟨*proof*⟩

lemma (**in** *Ring*) $in\text{-}s\text{-}set{:}x \in s\text{-}set\ R\ s\ A \Longrightarrow (\exists\, r \in carrier\ R.\ \exists\, a \in A.$
$x = r\ {}_s{\cdot}\ a\ ) \vee x \in A$
⟨*proof*⟩

lemma (**in** *Ring*) $sop\text{-}closedTr0{:}[\![ipp\text{-}cond1\ (s\text{-}set\ R\ s\ A)\ i;$
$ipp\text{-}cond2\ z\ (s\text{-}set\ R\ s\ A)\ i\ f;\ ipp\text{-}cond3\ z\ i;$
$inv\text{-}ipp\ z\ i\ f\ (s\text{-}set\ R\ s\ A);\ zeroA\ z\ i\ f\ (s\text{-}set\ R\ s\ A)\ z;$
$sop\text{-}distr2\ R\ s\ f\ i\ (s\text{-}set\ R\ s\ A)\ z;$
$sop\text{-}assoc\ R\ s\ (aug\text{-}pm\text{-}set\ z\ i\ (s\text{-}set\ R\ s\ A));$
$sop\text{-}inv\ R\ s\ i\ (s\text{-}set\ R\ s\ A);$
$sop\text{-}one\ R\ s\ (aug\text{-}pm\text{-}set\ z\ i\ (s\text{-}set\ R\ s\ A));\ sop\text{-}z\ R\ s\ z;$
$r \in carrier\ R;\ x \in aug\text{-}pm\text{-}set\ z\ i\ (s\text{-}set\ R\ s\ A)]\!] \Longrightarrow$
$r\ {}_s{\cdot}\ x \in aug\text{-}pm\text{-}set\ z\ i\ (s\text{-}set\ R\ s\ A)$
⟨*proof*⟩

lemma (**in** *Ring*) *sop-closedTr*:⟦*ipp-cond1* (*s-set R s A*) *i*;
ipp-cond2 z (*s-set R s A*) *i f*; *ipp-cond3 z i*;
inv-ipp z i f (*s-set R s A*); *zeroA z i f* (*s-set R s A*) *z*;
sop-distr2 R s f i (*s-set R s A*) *z*;
sop-assoc R s (*aug-pm-set z i* (*s-set R s A*));
sop-inv R s i (*s-set R s A*);
sop-one R s (*aug-pm-set z i* (*s-set R s A*)); *sop-z R s z*⟧ ⟹
∀ *r*∈*carrier R*. ∀ *x*∈*add-set f* (*aug-pm-set z i* (*s-set R s A*)) *n*.
r $_s\cdot$ *x* ∈ *add-set f* (*aug-pm-set z i* (*s-set R s A*)) *n*
⟨*proof*⟩

lemma (**in** *Ring*) *sop-closed*:⟦*ipp-cond1* (*s-set R s A*) *i*;
ipp-cond2 z (*s-set R s A*) *i f*; *ipp-cond3 z i*;
inv-ipp z i f (*s-set R s A*); *zeroA z i f* (*s-set R s A*) *z*;
sop-distr2 R s f i (*s-set R s A*) *z*;
sop-assoc R s (*aug-pm-set z i* (*s-set R s A*));
sop-inv R s i (*s-set R s A*);
sop-one R s (*aug-pm-set z i* (*s-set R s A*)); *sop-z R s z*⟧ ⟹
∀ *r*∈*carrier R*. ∀ *x*∈*addition-set f* (*aug-pm-set z i* (*s-set R s A*)).
r $_s\cdot$ *x* ∈ *addition-set f* (*aug-pm-set z i* (*s-set R s A*))
⟨*proof*⟩

lemma (**in** *Ring*) *sop-oneTr*:⟦*commute-bpp f* (*aug-pm-set z i* (*s-set R s A*));
assoc-bpp (*aug-pm-set z i* (*s-set R s A*)) *f*;
ipp-cond1 (*s-set R s A*) *i*; *ipp-cond2 z* (*s-set R s A*) *i f*;
ipp-cond3 z i; *inv-ipp z i f* (*s-set R s A*); *zeroA z i f* (*s-set R s A*) *z*;
sop-distr2 R s f i (*s-set R s A*) *z*;
sop-assoc R s (*aug-pm-set z i* (*s-set R s A*));
sop-one R s (*aug-pm-set z i* (*s-set R s A*))⟧ ⟹
∀ *x*∈*add-set f* (*aug-pm-set z i* (*s-set R s A*)) *n*. (*1* $_r$) $_s\cdot$ *x* = *x*
⟨*proof*⟩

lemma (**in** *Ring*) *sop-one*:⟦*commute-bpp f* (*aug-pm-set z i* (*s-set R s A*));
assoc-bpp (*aug-pm-set z i* (*s-set R s A*)) *f*; *ipp-cond1* (*s-set R s A*) *i*;
ipp-cond2 z (*s-set R s A*) *i f*; *ipp-cond3 z i*;
inv-ipp z i f (*s-set R s A*); *zeroA z i f* (*s-set R s A*) *z*;
sop-distr2 R s f i (*s-set R s A*) *z*;
sop-assoc R s (*aug-pm-set z i* (*s-set R s A*));
sop-one R s (*aug-pm-set z i* (*s-set R s A*))⟧ ⟹
∀ *x*∈*addition-set f* (*aug-pm-set z i* (*s-set R s A*)). (*1* $_r$) $_s\cdot$ *x* = *x*
⟨*proof*⟩

lemma (**in** *Ring*) *sop-assocTr*:⟦*ipp-cond1* (*s-set R s A*) *i*;
ipp-cond2 z (*s-set R s A*) *i f*; *ipp-cond3 z i*;
inv-ipp z i f (*s-set R s A*); *zeroA z i f* (*s-set R s A*) *z*;
sop-distr2 R s f i (*s-set R s A*) *z*;
sop-assoc R s (*aug-pm-set z i* (*s-set R s A*));
sop-inv R s i (*s-set R s A*);
sop-one R s (*aug-pm-set z i* (*s-set R s A*)); *sop-z R s z*⟧ ⟹

$\forall\, a{\in}carrier\ R.\ \forall\, b{\in}carrier\ R.$
$\forall\, x{\in}add\text{-}set\ f\ (aug\text{-}pm\text{-}set\ z\ i\ (s\text{-}set\ R\ s\ A))\ n.$
$a\ {}_s{\cdot}\ (\ b\ {}_s{\cdot}\ x) = (a\ {\cdot}_r\ b)\ {}_s{\cdot}\ x$

⟨*proof*⟩

lemma (**in** *Ring*) *sop-assoc*:⟦*ipp-cond1* (*s-set R s A*) *i*;
ipp-cond2 z (*s-set R s A*) *i f*; *ipp-cond3 z i*;
inv-ipp z i f (*s-set R s A*); *zeroA z i f* (*s-set R s A*) *z*;
sop-distr2 R s f i (*s-set R s A*) *z*;
sop-assoc R s (*aug-pm-set z i* (*s-set R s A*));
sop-inv R s i (*s-set R s A*); *sop-z R s z*;
sop-one R s (*aug-pm-set z i* (*s-set R s A*))⟧ ⟹
$\forall\, a{\in}carrier\ R.\ \forall\, b{\in}carrier\ R.$
$\forall\, x{\in}addition\text{-}set\ f\ (aug\text{-}pm\text{-}set\ z\ i\ (s\text{-}set\ R\ s\ A)).$
$a\ {}_s{\cdot}\ (b\ {}_s{\cdot}\ x) = (\ a\ {\cdot}_r\ b)\ {}_s{\cdot}\ x$

⟨*proof*⟩

lemma (**in** *Ring*) *s-set-commute*:⟦*commute-bpp f* (*aug-pm-set z i* (*s-set R s A*));
$x \in addition\text{-}set\ f\ (aug\text{-}pm\text{-}set\ z\ i\ (s\text{-}set\ R\ s\ A))$;
$y \in addition\text{-}set\ f\ (aug\text{-}pm\text{-}set\ z\ i\ (s\text{-}set\ R\ s\ A))$⟧ ⟹
$x\ {}_f{+}\ y = y\ {}_f{+}\ x$

⟨*proof*⟩

lemma (**in** *Ring*) *add-s-set-inc-add-set*:
$add\text{-}set\ f\ (aug\text{-}pm\text{-}set\ z\ i\ A)\ n \subseteq$
$add\text{-}set\ f\ (aug\text{-}pm\text{-}set\ z\ i\ (s\text{-}set\ R\ s\ A))\ n$

⟨*proof*⟩

lemma (**in** *Ring*) *sop-distr1Tr*:⟦*commute-bpp f* (*aug-pm-set z i* (*s-set R s A*));
assoc-bpp (*aug-pm-set z i* (*s-set R s A*)) *f*; *ipp-cond1* (*s-set R s A*) *i*;
ipp-cond2 z (*s-set R s A*) *i f*; *ipp-cond3 z i*;
inv-ipp z i f (*s-set R s A*); *zeroA z i f* (*s-set R s A*) *z*;
sop-distr1 R s f i (*s-set R s A*) *z*;
sop-distr2 R s f i (*s-set R s A*) *z*;
sop-assoc R s (*aug-pm-set z i* (*s-set R s A*));
sop-inv R s i (*s-set R s A*);
sop-one R s (*aug-pm-set z i* (*s-set R s A*)); *sop-z R s z*⟧ ⟹
$\forall\, a{\in}carrier\ R.\ \forall\, b{\in}carrier\ R.\ \forall\, x{\in}\ add\text{-}set\ f\ (aug\text{-}pm\text{-}set\ z\ i\ (s\text{-}set\ R\ s\ A))\ n.$
$(a \pm b)\ {}_s{\cdot}\ x = a\ {}_s{\cdot}\ x\ {}_f{+}\ (b\ {}_s{\cdot}\ x)$

⟨*proof*⟩

lemma (**in** *Ring*) *sop-distr1*:⟦*commute-bpp f* (*aug-pm-set z i* (*s-set R s A*));
assoc-bpp (*aug-pm-set z i* (*s-set R s A*)) *f*; *ipp-cond1* (*s-set R s A*) *i*;
ipp-cond2 z (*s-set R s A*) *i f*; *ipp-cond3 z i*;
inv-ipp z i f (*s-set R s A*); *zeroA z i f* (*s-set R s A*) *z*;
sop-distr1 R s f i (*s-set R s A*) *z*;
sop-distr2 R s f i (*s-set R s A*) *z*;
sop-assoc R s (*aug-pm-set z i* (*s-set R s A*));
sop-inv R s i (*s-set R s A*);

sop-one R s (aug-pm-set z i (s-set R s A)); sop-z R s z⟧ ⟹
∀ *a*∈*carrier R*. ∀ *b*∈*carrier R*.
∀ *x*∈ *addition-set f* (*aug-pm-set z i* (*s-set R s A*)).
(*a* ± *b*) $_s\cdot$ *x* = *a* $_s\cdot$ *x* $_f+$ (*b* $_s\cdot$ *x*)

⟨*proof*⟩

definition
fgmodule-condition ::[('*r*, '*m*) *Ring-scheme*, '*a* ⇒ '*a* ⇒ '*a*, '*a* ⇒ '*a*,
'*r* ⇒ '*a* ⇒ '*a*, '*a set*, '*a*] ⇒ *bool* **where**
fgmodule-condition R f i s A z ⟷
commute-bpp f (*aug-pm-set z i* (*s-set R s A*)) ∧
assoc-bpp (*aug-pm-set z i* (*s-set R s A*)) *f* ∧
ipp-cond1 (*s-set R s A*) *i* ∧ *ipp-cond2 z* (*s-set R s A*) *i f* ∧
ipp-cond3 z i ∧ *inv-ipp z i f* (*s-set R s A*) ∧
zeroA z i f (*s-set R s A*) *z* ∧ *sop-distr1 R s f i* (*s-set R s A*) *z* ∧
sop-distr2 R s f i (*s-set R s A*) *z* ∧
sop-assoc R s (*aug-pm-set z i* (*s-set R s A*)) ∧
sop-inv R s i (*s-set R s A*) ∧
sop-one R s (*aug-pm-set z i* (*s-set R s A*)) ∧ *sop-z R s z*

lemma (**in** *Ring*) *sop-closed1*:⟦*fgmodule-condition R f i s A z*; *r* ∈ *carrier R*;
x ∈ *addition-set f* (*aug-pm-set z i* (*s-set R s A*))⟧ ⟹
r $_s\cdot$ *x* ∈ *addition-set f* (*aug-pm-set z i* (*s-set R s A*))

⟨*proof*⟩

lemma (**in** *Ring*) *fgmodule-is-module*:*fgmodule-condition R f i s A z*
⟹ *R module* (*fgmodule R A z i f s*)

⟨*proof*⟩

lemma (**in** *Ring*) *a-in-carr-fgmodule*:*a* ∈ *A*
⟹ *a* ∈ *carrier* (*fgmodule R A z i f s*)

⟨*proof*⟩

6.5 A fgmodule and a free module

lemma (**in** *Ring*) *fg-zeroTr*:⟦*fgmodule-condition R f i s A z*; *a* ∈ *A*⟧ ⟹
0 $_s\cdot$ *a* = *z*

⟨*proof*⟩

lemma (**in** *Ring*) *fg-genTr0*:⟦*fgmodule-condition R f i s A z*;
x ∈ *aug-pm-set z i* (*s-set R s A*)⟧ ⟹
x ∈ *linear-span R* (*fgmodule R A z i f s*) (*carrier R*) *A*

⟨*proof*⟩

lemma (**in** *Ring*) *fg-genTr*:*fgmodule-condition R f i s A z* ⟹
∀ *x*. *x* ∈ (*add-set f* (*aug-pm-set z i* (*s-set R s A*)) *n*) ⟶
x ∈ *linear-span R* (*fgmodule R A z i f s*) (*carrier R*) *A*

⟨*proof*⟩

lemma (**in** *Ring*) *generator-of-fgm*:*fgmodule-condition R f i s A z* $\Longrightarrow$
generator R (*fgmodule R A z i f s*) *A*
⟨*proof*⟩

lemma (**in** *Ring*) *fg-freeTr1*:⟦*R module M*; *free-generator R M A*;
R module fgmodule R A z i f s; *free-generator R* (*fgmodule R A z i f s*) *A*;
g ∈ *mHom R M* (*fgmodule R A z i f s*); ∀ *x*∈*A*. *g x* = *x*⟧ $\Longrightarrow$
∀ *fa sa*. *fa* ∈ {*j*. *j* ≤ (*n*::*nat*)} → *A* ∧ *sa* ∈ {*j*. *j* ≤ *n*} → *carrier R* ⟶
l-comb R (*fgmodule R A z i f s*) *n sa* (*cmp g fa*) =
l-comb R (*fgmodule R A z i f s*) *n sa fa*
⟨*proof*⟩

lemma (**in** *Ring*) *fg-freeTr*:⟦*R module M*; *free-generator R M A*;
R module fgmodule R A z i f s;
free-generator R (*fgmodule R A z i f s*) *A*;
g ∈ *mHom R M* (*fgmodule R A z i f s*); ∀ *x*∈*A*. *g x* = *x*;
fa ∈ {*j*. *j* ≤ (*n*::*nat*)} → *A*; *sa* ∈ {*j*. *j* ≤ *n*} → *carrier R*⟧ $\Longrightarrow$
l-comb R (*fgmodule R A z i f s*) *n sa* (*cmp g fa*) =
l-comb R (*fgmodule R A z i f s*) *n sa fa*
⟨*proof*⟩

lemma (**in** *Ring*) *fg-free1*:⟦ *A* ≠ {}; *fgmodule-condition R f i s A z*;
free-generator R (*fgmodule R A z i f s*) *A*; *R module M*;
free-generator R M A⟧ $\Longrightarrow$ *M* $\cong_R$ (*fgmodule R A z i f s*)
⟨*proof*⟩

lemma (**in** *Ring*) *fg-free*:⟦*fgmodule-condition R f i s A z*;
free-generator R (*fgmodule R A z i f s*) *A*; *R module M*;
free-generator R M A⟧ $\Longrightarrow$ *M* $\cong_R$ (*fgmodule R A z i f s*)
⟨*proof*⟩

6.6 Direct sum, again

definition
miota :: [('*r*, '*m*) *Ring-scheme*, ('*a*, '*r*, '*m1*) *Module-scheme*,
('*a*, '*r*, '*m1*) *Module-scheme*] ⇒ '*a* ⇒ '*a* **where**
miota R M1 M = (λ*x*∈*carrier M1*. *x*)

definition
msubmodule ::[('*r*, '*m*) *Ring-scheme*, ('*a*, '*r*, '*m1*) *Module-scheme*,
('*a*, '*r*, '*m1*) *Module-scheme*] ⇒ *bool* **where**
msubmodule R M M1 ⟷ *miota R M1 M* ∈ *mHom R M1 M* ∧
(*carrier M1*) ⊆ (*carrier M*)

definition
ds2 :: [('*r*, '*m*) *Ring-scheme*, ('*a*, '*r*, '*m1*) *Module-scheme*,
('*a*, '*r*, '*m1*) *Module-scheme*, ('*a*, '*r*, '*m1*) *Module-scheme*] ⇒ *bool* **where**

ds2 R M M1 M2 $\longleftrightarrow$ *R module M* $\wedge$ *msubmodule R M M1* $\wedge$ *msubmodule R M M2* $\wedge$
$(\forall\, x{\in}carrier\ M.\ \exists\, m1{\in}carrier\ M1.\ \exists\, m2{\in}carrier\ M2.\ x = m1\ \pm_M\ m2) \wedge$

$(carrier\ M1) \cap (carrier\ M2) = \{\mathbf{0}_M\}$

abbreviation
DS2 $(\langle(4\text{-}/\ \bigoplus_{\text{-},\text{-}}\ \text{-})\rangle\ [92,93,92,92]92)$ **where**
$M1 \bigoplus_{R,M} M2 == ds2\ R\ M\ M1\ M2$

lemma (**in** *Ring*) *ds2-commute*:$[\![R\ module\ M1;\ R\ module\ M2;\ R\ module\ M;$
$M1 \bigoplus_{R,M} M2]\!] \Longrightarrow M2 \bigoplus_{R,M} M1$
⟨*proof*⟩

lemma (**in** *Ring*) *msub-addition*:$[\![R\ module\ M;\ R\ module\ M1;\ msubmodule\ R\ M\ M1;$
$x \in carrier\ M1;\ y \in carrier\ M1]\!] \Longrightarrow x \pm_{M1} y = x \pm_M y$
⟨*proof*⟩

lemma (**in** *Ring*) *msub-mOp*:$[\![R\ module\ M;\ R\ module\ M1;\ msubmodule\ R\ M\ M1;$
$x \in carrier\ M1]\!] \Longrightarrow -_{a\,M1}\ x\ = -_{a\,M}\ x$
⟨*proof*⟩

lemma (**in** *Ring*) *msub-sprod*:$[\![R\ module\ M;\ R\ module\ M1;\ msubmodule\ R\ M\ M1;$
$a \in carrier\ R;\ x \in carrier\ M1]\!] \Longrightarrow a \cdot_{s\,M1} x = a \cdot_{s\,M} x$
⟨*proof*⟩

lemma (**in** *Ring*) *msub-submodule*:$[\![R\ module\ M;\ R\ module\ M1;\ msubmodule\ R\ M\ M1]\!]$
$\Longrightarrow\ submodule\ R\ M\ (carrier\ M1)$
⟨*proof*⟩

lemma (**in** *Ring*) *ds2-unique*:$[\![R\ module\ M;\ R\ module\ M1;\ R\ module\ M2;$
$ds2\ R\ M\ M1\ M2;\ \ m1 \in carrier\ M1;\ m1' \in carrier\ M1;$
$m2 \in carrier\ M2;\ m2' \in carrier\ M2;$
$m1 \pm_M m2 = m1' \pm_M m2']\!] \Longrightarrow m1 = m1' \wedge m2 = m2'$
⟨*proof*⟩

lemma (**in** *Ring*) *miota-injec*:$[\![R\ module\ M;\ R\ module\ M1;\ R\ module\ M2;$
$ds2\ R\ M\ M1\ M2;\ msubmodule\ R\ M\ M1]\!] \Longrightarrow$
$miota\ R\ M1\ M \in mHom\ R\ M1\ M \wedge injec_{M1,M}\ (miota\ R\ M1\ M)$
⟨*proof*⟩

definition
mproj1 :: $[('r,\ 'm)\ Ring\text{-}scheme,\ ('a,\ 'r,\ 'm1)\ Module\text{-}scheme,$
$('a,\ 'r,\ 'm1)\ Module\text{-}scheme,\ ('a,\ 'r,\ 'm1)\ Module\text{-}scheme] \Rightarrow 'a \Rightarrow 'a$ **where**
$mproj1\ R\ M1\ M2\ M = (\lambda x{\in}carrier\ M.\ THE\ x1.\ x1 \in carrier\ M1 \wedge$
$(x \pm_M (-_{a\,M}\ x1)) \in carrier\ M2)$

definition
mproj2 :: [('r, 'm) *Ring-scheme*, ('a, 'r, 'm1) *Module-scheme*,
('a, 'r, 'm1) *Module-scheme*, ('a, 'r, 'm1) *Module-scheme*] ⇒ 'a ⇒ 'a **where**
mproj2 R M1 M2 M = *mproj1 R M2 M1 M*

lemma (**in** *Ring*) *ds2-components*:⟦*R module M1*; *R module M2*; *R module M*;
M1 $\bigoplus_{R,M}$ *M2*; *a* ∈ *carrier M*⟧ ⟹
∃ *a1*∈*carrier M1*. ∃ *a2*∈*carrier M2*. *a* = *a1* $\pm_M$ *a2*
⟨*proof*⟩

lemma (**in** *Ring*) *ds2-components1*:⟦*R module M1*; *R module M2*; *R module M*;
M1 $\bigoplus_{R,M}$ *M2*; *a* ∈ *carrier M*⟧ ⟹
∃ *a1*∈*carrier M1*. *a* $\pm_M$ $-_{a\,M}$ *a1* ∈ *carrier M2*
⟨*proof*⟩

lemma (**in** *Ring*) *mprojTr1*:⟦*R module M1*; *R module M2*; *R module M*; *ds2 R M M1 M2*;
x ∈ *carrier M* ⟧ ⟹ ∃!*x1*. *x1* ∈ *carrier M1* ∧ (*x* $\pm_M$ $-_{a\,M}$ *x1*) ∈ *carrier M2*
⟨*proof*⟩

lemma (**in** *Ring*) *mprojTr2*:⟦*R module M1*; *R module M2*; *R module M*; *ds2 R M M1 M2*;
x ∈ *carrier M*; *x1* ∈ *carrier M1*; (*x* $\pm_M$ ($-_{a\,M}$ *x1*)) ∈ *carrier M2*;
y1 ∈ *carrier M1*;(*x* $\pm_M$ ($-_{a\,M}$ *y1*)) ∈ *carrier M2* ⟧ ⟹ *x1* = *y1*
⟨*proof*⟩

lemma (**in** *Ring*) *mprojTr3*:⟦*R module M1*; *R module M2*; *R module M*; *ds2 R M M1 M2*;
a ∈ *carrier M*; *a1* ∈ *carrier M1*; (*a* $\pm_M$ ($-_{a\,M}$ *a1*)) ∈ *carrier M2*⟧ ⟹
(*THE x1*. *x1* ∈ *carrier M1* ∧ *a* $\pm_M$ $-_{a\,M}$ *x1* ∈ *carrier M2*) = *a1*
⟨*proof*⟩

lemma (**in** *Ring*) *mproj*:⟦*R module M1*; *R module M2*; *R module M*; *ds2 R M M1 M2*⟧
⟹ *mproj1 R M1 M2 M* ∈ *mHom R M M1*
⟨*proof*⟩

lemma (**in** *Ring*) *mproj2*:⟦*R module M1*; *R module M2*; *R module M*; *M1* $\bigoplus_{R,M}$ *M2*⟧
⟹ *mproj2 R M1 M2 M* ∈ *mHom R M M2*
⟨*proof*⟩

6.6.1 Existence of the tensor product

definition
fm-gen-by-prod :: [('r, 'm) *Ring-scheme*, (('a ∗ 'b), 'r, 'm1) *Module-scheme*,
('a, 'r, 'm1) *Module-scheme*, ('b, 'r, 'm1) *Module-scheme*] ⇒ *bool*

(‹(4FM_-/ - - -)› [100,100,101]100) **where**
$FM_R\ P\ M\ N \longleftrightarrow R$ *module* $P \wedge$ *free-generator* $R\ P\ (M \times_c N)$

lemma (**in** *Ring*) *free-gen-gen*:$FM_R\ P\ M\ N \Longrightarrow$ *generator* $R\ P\ (M \times_c N)$
⟨*proof*⟩

lemma (**in** *Ring*) *free-gen-mem*:$[\![FM_R\ P\ M\ N;\ a \in (M \times_c N)]\!] \Longrightarrow\ a \in$ *carrier* P
⟨*proof*⟩

lemma (**in** *Ring*) *mHom-lin-nsumTr*:$[\![R$ *module* $M;\ R$ *module* $N;\ t \in$ *mHom* $R\ M\ N]\!] \Longrightarrow$
$f \in \{j.\ j \leq (n::nat)\} \to$ *carrier* $M\ \longrightarrow\ t\ (nsum\ M\ f\ n) = nsum\ N\ (cmp\ t\ f)\ n$
⟨*proof*⟩

lemma (**in** *Ring*) *mHom-lin-nsum*:$[\![R$ *module* $M;\ R$ *module* $N;\ t \in$ *mHom* $R\ M\ N;$
$f \in \{j.\ j \leq (n::nat)\} \to$ *carrier* $M]\!] \Longrightarrow$
$t\ (nsum\ M\ f\ n) = nsum\ N\ (cmp\ t\ f)\ n$
⟨*proof*⟩

lemma (**in** *Ring*) *module-over-zeroring*:$[\![$*zeroring* $R;\ R$ *module* $M]\!] \Longrightarrow$
carrier $M = \{\mathbf{0}_M\}$
⟨*proof*⟩

lemma (**in** *Ring*) *submodule-over-zeroring*:$[\![$*zeroring* $R;\ R$ *module* $M;$
submodule $R\ M\ N]\!] \Longrightarrow\ N = \{\mathbf{0}_M\}$
⟨*proof*⟩

definition
Least-submodule :: [('*r*, '*m*) *Ring-scheme*, ('*a*, '*r*, '*m1*) *Module-scheme*,
'*a set*] ⇒ '*a set*
(‹(3LSM_-/ -/ -)› [100,100,101]100) **where**
$LSM_R\ M\ T = \bigcap \{N.$ *submodule* $R\ M\ N \wedge\ T \subseteq N\}$

lemma (**in** *Ring*) *LSM-mem*:$[\![R$ *module* $M;\ T \subseteq$ *carrier* $M;\ t \in T]\!] \Longrightarrow$
$t \in (LSM_R\ M\ T)$
⟨*proof*⟩

lemma (**in** *Ring*) *LSM-sub-M*:$[\![R$ *module* $M;\ T \subseteq$ *carrier* $M]\!] \Longrightarrow$
$(LSM_R\ M\ T) \subseteq$ *carrier* M
⟨*proof*⟩

lemma (**in** *Ring*) *LSM-sub-submodule*:$[\![R$ *module* $M;\ T \subseteq$ *carrier* $M;$
submodule $R\ M\ N;\ T \subseteq N\ ]\!] \Longrightarrow (LSM_R\ M\ T) \subseteq N$
⟨*proof*⟩

lemma (**in** *Ring*) *LSM-inc-T*:$[\![R$ *module* $M;\ T \subseteq$ *carrier* $M]\!] \Longrightarrow T \subseteq (LSM_R\ M$

T)
⟨*proof*⟩

lemma (**in** *Ring*) *LSM-submodule*:⟦*R module M*; $T \subseteq$ *carrier M*⟧ ⟹
submodule R M (*LSM*$_R$ *M T*)
⟨*proof*⟩

lemma (**in** *Ring*) *linear-comb-memTr*:⟦*R module M*; *submodule R M N*; $T \subseteq N$⟧
⟹
$\forall f\ s.\ f \in \{j.\ j \leq (n::nat)\} \to T \wedge s \in \{j.\ j \leq n\} \to$ *carrier R* ⟶
l-comb R M n s f $\in N$
⟨*proof*⟩

lemma (**in** *Ring*) *linear-comb-mem*:⟦*R module M*; *submodule R M N*; $T \subseteq N$;
$f \in \{j.\ j \leq (n::nat)\} \to T$; $s \in \{j.\ j \leq n\} \to$ *carrier R*⟧ ⟹
l-comb R M n s f $\in N$
⟨*proof*⟩

lemma (**in** *Ring*) *LSM-eq-linear-span*:⟦*R module M*; $T \subseteq$ *carrier M*⟧ ⟹
(*LSM*$_R$ *M T*) = *linear-span R M* (*carrier R*) *T*
⟨*proof*⟩

lemma (**in** *Ring*) *LSM-sub-ker*:⟦*R module M*; *R module N*; $T \subseteq$ *carrier M*;
$f \in$ *mHom R M N*; $T \subseteq$ *ker*$_{M,N}$ *f*⟧ ⟹ *LSM*$_R$ *M T* $\subseteq$ *ker*$_{M,N}$ *f*
⟨*proof*⟩

definition
tensor-relations1 :: [('*r*, '*m*) *Ring-scheme*, ('*a*, '*r*, '*m1*) *Module-scheme*,
('*b*, '*r*, '*m1*) *Module-scheme*, (('*a* ∗ '*b*), '*r*, '*m1*) *Module-scheme*] ⇒
('*a* ∗ '*b*) *set*
(‹(*4TR1*/ -/ -/ -/ -)› [*100*,*100*,*100*,*101*]*100*) **where**
TR1 R M N MN = {*x*. ∃ *m1*∈*carrier M*. ∃ *m2*∈*carrier M*. ∃ *n*∈*carrier N*.
x = (*m1* ±$_M$ *m2*, *n*) ±$_{MN}$ (−$_{aMN}$ ((*m1*, *n*) ±$_{MN}$ (*m2*, *n*)))}

definition
tensor-relations2 :: [('*r*, '*m*) *Ring-scheme*, ('*a*, '*r*, '*m1*) *Module-scheme*,
('*b*, '*r*, '*m1*) *Module-scheme*, (('*a* ∗ '*b*), '*r*, '*m1*) *Module-scheme*] ⇒
('*a* ∗ '*b*) *set*
(‹(*4TR2*/ -/ -/ -/ -)› [*100*,*100*,*100*, *101*]*100*) **where**
TR2 R M N MN = {*x*. ∃ *m*∈*carrier M*. ∃ *n1*∈*carrier N*. ∃ *n2*∈*carrier N*.
x = (*m*, *n1* ±$_N$ *n2*) ±$_{MN}$ (−$_{aMN}$ ((*m*, *n1*) ±$_{MN}$ (*m*, *n2*)))}

definition
tensor-relations3 :: [('*r*, '*m*) *Ring-scheme*, ('*a*, '*r*, '*m1*) *Module-scheme*,
('*b*, '*r*, '*m1*) *Module-scheme*, (('*a* ∗ '*b*), '*r*, '*m1*) *Module-scheme*] ⇒
('*a* ∗ '*b*) *set*
(‹(*4TR3*/ -/ -/ -/ -)› [*100*,*100*,*100*,*101*]*100*) **where**
TR3 R M N P = {*x*. ∃ *m*∈*carrier M*. ∃ *n*∈*carrier N*. ∃ *a*∈*carrier R*.

$x = (a \cdot_{s M} m,\ n) \pm_P (-_{a P}\ (a \cdot_{s P} (m,\ n)))\}$

definition
tensor-relations4 :: [('r, 'm) *Ring-scheme*, ('a, 'r, 'm1) *Module-scheme*,
('b, 'r, 'm1) *Module-scheme*, (('a * 'b), 'r, 'm1) *Module-scheme*] ⇒
('a * 'b) *set*
(‹(*4TR4*/ -/ -/ -/ -)› [*100*,*100*,*100*,*101*]*100*) **where**
TR4 R M N MN = {*x*. ∃ *m*∈*carrier M*. ∃ *n*∈*carrier N*. ∃ *a*∈*carrier R*.
x = (*m*, *a* ·$_{s N}$ *n*) ±$_{MN}$ (−$_{a MN}$ (*a* ·$_{s\ MN}$ (*m*, *n*)))}

definition
tensor-relations :: [('r, 'm) *Ring-scheme*, ('a, 'r, 'm1) *Module-scheme*,
('b, 'r, 'm1) *Module-scheme*, (('a * 'b), 'r, 'm1) *Module-scheme*] ⇒
('a * 'b) *set*
(‹(*4TR*- -/ -/ -)› [*100*,*100*,*101*]*100*) **where**
TR_R *M N MN* = LSM_R *MN* ((*TR1 R M N MN*) ∪ (*TR2 R M N MN*) ∪
(*TR3 R M N MN*) ∪ (*TR4 R M N MN*))

definition
tensor-product :: [('r, 'm) *Ring-scheme*, ('a, 'r, 'm1) *Module-scheme*,
('b, 'r, 'm1) *Module-scheme*, (('a * 'b), 'r, 'm1) *Module-scheme*] ⇒
(('a * 'b) *set*, 'r) *Module* **where**
tensor-product R M N MN = *MN* /$_m$ (TR_R *M N MN*)

abbreviation
TENSORPROD (‹(*4*-/ -⨂-/ -)› [*92*,*92*,*92*,*93*]*92*) **where**
M $_P\bigotimes_R$ *N* == *tensor-product R M N P*

lemma (**in** *Ring*) *mem-cartesian*:⟦*R module M*; *R module N*; *m* ∈ *carrier M*;
n ∈ *carrier N* ⟧ ⟹ (*m*, *n*) ∈ *M* ×$_c$ *N*
⟨*proof*⟩

lemma (**in** *Ring*) *cartesianTr*:⟦*R module M*; *R module N*; *x* ∈ *M* ×$_c$ *N*⟧ ⟹
∃ *m n*. *m*∈*carrier M* ∧ *n* ∈ *carrier N* ∧ *x* = (*m*, *n*)
⟨*proof*⟩

lemma (**in** *Ring*) *free-module-mem*:⟦*R module M*; *R module N*; *m* ∈ *carrier M*;
n ∈ *carrier N*; FM_R *P M N*⟧ ⟹ (*m*, *n*) ∈ *carrier P*
⟨*proof*⟩

lemma (**in** *Ring*) *FM-P-module*:⟦*R module M*; *R module N*; FM_R *P M N*⟧
⟹ *R module P*
⟨*proof*⟩

lemma (**in** *Ring*) *TR1-sub-carr*:⟦*R module M*; *R module N*; FM_R *P M N*⟧ ⟹
(*TR1 R M N P*) ⊆ *carrier P*
⟨*proof*⟩

lemma (**in** *Ring*) *TR2-sub-carr*:⟦*R module M*; *R module N*; FM_R *P M N*⟧ ⟹

$(TR2\ R\ M\ N\ P) \subseteq carrier\ P$
⟨*proof*⟩

lemma (**in** *Ring*) *TR3-sub-carr*:⟦*R module M*; *R module N*; $FM_R\ P\ M\ N$⟧ ⟹
$(TR3\ R\ M\ N\ P) \subseteq carrier\ P$
⟨*proof*⟩

lemma (**in** *Ring*) *TR4-sub-carr*:⟦*R module M*; *R module N*; $FM_R\ P\ M\ N$⟧ ⟹
$(TR4\ R\ M\ N\ P) \subseteq carrier\ P$
⟨*proof*⟩

lemma (**in** *Ring*) *TR-sub-carr*:⟦*R module M*; *R module N*; $FM_R\ P\ M\ N$⟧ ⟹
$(TR1\ R\ M\ N\ P) \cup (TR2\ R\ M\ N\ P) \cup (TR3\ R\ M\ N\ P) \cup (TR4\ R\ M\ N\ P) \subseteq$ *carrier P*
⟨*proof*⟩

lemma (**in** *Ring*) *TR-submodule*:⟦*R module M*; *R module N*; $FM_R\ P\ M\ N$⟧ ⟹
$submodule\ R\ P\ (TR_R\ M\ N\ P)$
⟨*proof*⟩

lemma (**in** *Ring*) *TR-cont-TR1234*:⟦*R module M*; *R module N*; $FM_R\ P\ M\ N$⟧
⟹
$TR1\ R\ M\ N\ P \cup TR2\ R\ M\ N\ P \cup TR3\ R\ M\ N\ P \cup TR4\ R\ M\ N\ P \subseteq TR_R\ M\ N\ P$
⟨*proof*⟩

lemma (**in** *Ring*) *TR1-mem*:⟦*R module M*; *R module N*; $FM_R\ P\ M\ N$; *m1* ∈ *carrier M*;
m2 ∈ *carrier M*; *n* ∈ *carrier N*⟧ ⟹ $(m1 \pm_M m2, n) \pm_P -_{aP} ((m1, n) \pm_P (m2, n))$
$\in TR_R\ M\ N\ P$
⟨*proof*⟩

lemma (**in** *Ring*) *TR2-mem*:⟦*R module M*; *R module N*; $FM_R\ P\ M\ N$; *m* ∈ *carrier M*;
n1 ∈ *carrier N*; *n2* ∈ *carrier N* ⟧ ⟹
$(m, n1 \pm_N n2) \pm_P -_{aP} ((m, n1) \pm_P (m, n2)) \in TR_R\ M\ N\ P$
⟨*proof*⟩

lemma (**in** *Ring*) *TR3-mem*:⟦*R module M*; *R module N*; $FM_R\ P\ M\ N$; *m* ∈ *carrier M*;
n ∈ *carrier N*; *a* ∈ *carrier R*⟧ ⟹
$(a \cdot_{sM} m, n) \pm_P -_{aP} (a \cdot_{sP} (m, n)) \in TR_R\ M\ N\ P$
⟨*proof*⟩

lemma (**in** *Ring*) *TR4-mem*:⟦*R module M*; *R module N*; $FM_R\ P\ M\ N$; *m* ∈ *carrier M*;
n ∈ *carrier N*; *a* ∈ *carrier R*⟧ ⟹
$(m, a \cdot_{sN} n) \pm_P -_{aP} (a \cdot_{sP} (m, n)) \in TR_R\ M\ N\ P$

⟨*proof*⟩

lemma (**in** *Ring*) *tensor-product-module*:⟦*R module M*; *R module N*; FM_R *P M N*⟧ ⟹
R module (*tensor-product R M N P*)
⟨*proof*⟩

lemma (**in** *Ring*) *tau-mpj-bilin1*:⟦ *R module M*; *R module N*; FM_R *P M N*;
x1 ∈ *carrier M*; *x2* ∈ *carrier M*; *y* ∈ *carrier N*⟧ ⟹
(*mpj P* (TR_R *M N P*)) (*x1* $\pm_M$ *x2*, *y*) =
(*mpj P* (TR_R *M N P*)) (*x1*, *y*) $\pm_{(M\ _P\bigotimes_R\ N)}$ (*mpj P* (TR_R *M N P*) (*x2*, *y*))
⟨*proof*⟩

lemma (**in** *Ring*) *tau-mpj-bilin2*:⟦*R module M*; *R module N*; FM_R *P M N*;
m ∈ *carrier M*; *n1* ∈ *carrier N*; *n2* ∈ *carrier N*⟧ ⟹
(*mpj P* (TR_R *M N P*)) (*m*, *n1* $\pm_N$ *n2*) =
(*mpj P* (TR_R *M N P*)) (*m*, *n1*) $\pm_{(M\ _P\bigotimes_R\ N)}$ (*mpj P* (TR_R *M N P*) (*m*, *n2*))
⟨*proof*⟩

lemma (**in** *Ring*) *tau-mpj-bilin3*:⟦*R module M*; *R module N*; FM_R *P M N*;
m ∈ *carrier M*; *n* ∈ *carrier N*; *a* ∈ *carrier R*⟧ ⟹
(*mpj P* (TR_R *M N P*)) (*a* $\cdot_{sM}$ *m*, *n*) = *a* $\cdot_{s(M\ _P\bigotimes_R\ N)}$
(*mpj P* (TR_R *M N P*) (*m*, *n*))
⟨*proof*⟩

lemma (**in** *Ring*) *tau-mpj-bilin4*:⟦*R module M*; *R module N*; FM_R *P M N*;
m ∈ *carrier M*; *n* ∈ *carrier N*; *a* ∈ *carrier R*⟧ ⟹
(*mpj P* (TR_R *M N P*)) (*m*, *a* $\cdot_{sN}$ *n*) = *a* $\cdot_{s(M\ _P\bigotimes_R\ N)}$
(*mpj P* (TR_R *M N P*) (*m*, *n*))
⟨*proof*⟩

definition
tau :: [('*r*, '*m*) *Ring-scheme*, ('*a*, '*r*, '*m1*) *Module-scheme*,
('*b*, '*r*, '*m1*) *Module-scheme*, (('*a* ∗ '*b*), '*r*, '*m1*) *Module-scheme*] ⇒
('*a* ∗ '*b*) ⇒ ('*a* ∗ '*b*) **where**
tau R M N P = (λ*x*∈(*M* $\times_c$ *N*). *x*)

lemma (**in** *Ring*) *tau-func*:⟦*R module M*; *R module N*; FM_R *P M N*⟧ ⟹
tau R M N P ∈ *M* $\times_c$ *N* → *carrier P*
⟨*proof*⟩

lemma (**in** *Ring*) *tau-mem*:⟦*R module M*; *R module N*; *m* ∈ *carrier M*;
n ∈ *carrier N*; FM_R *P M N*⟧ ⟹ *tau R M N P* (*m*, *n*) ∈ *carrier P*
⟨*proof*⟩

lemma (**in** *Ring*) *tau-inj0*:⟦¬ *zeroring R*; *R module M*; *R module N*; FM_R *P M N*⟧

$\Longrightarrow$ *inj-on* (*tau R M N P*) (*M* $\times_c$ *N*)
⟨*proof*⟩

lemma (**in** *Ring*) *tau-inj1*:⟦*zeroring R*; *R module M*; *R module N*; FM_R *P M N*⟧
$\Longrightarrow$
inj-on (*tau R M N P*) (*M* $\times_c$ *N*)
⟨*proof*⟩

lemma (**in** *Ring*) *tau-inj*:⟦*R module M*; *R module N*; FM_R *P M N*⟧ $\Longrightarrow$
inj-on (*tau R M N P*) (*M* $\times_c$ *N*)
⟨*proof*⟩

lemma (**in** *Ring*) *tau-mpj-bilinear*:⟦*R module M*; *R module N*; FM_R *P M N*⟧ $\Longrightarrow$

bilinear-map (*compose* (*M* $\times_c$ *N*) (*mpj P* (TR_R *M N P*)) (*tau R M N P*))
R M N (*M* $_P\bigotimes_R$ *N*)
⟨*proof*⟩

definition
tnm :: [('*r*, '*m*) *Ring-scheme*, (('*a* ∗ '*b*), '*r*, '*m1*) *Module-scheme*,
('*a*, '*r*, '*m1*) *Module-scheme*, ('*b*, '*r*, '*m1*) *Module-scheme*] ⇒
('*a* ∗ '*b*) ⇒ ('*a* ∗ '*b*) *set* **where**
tnm R P M N = *compose* (*M* $\times_c$ *N*) (*mpj P* (TR_R *M N P*)) (*tau R M N P*)

lemma (**in** *Ring*) *tnm-bilinear*:⟦*R module M*; *R module N*; FM_R *P M N*⟧ $\Longrightarrow$
bilinear-map (*tnm R P M N*) *R M N* (*M* $_P\bigotimes_R$ *N*)
⟨*proof*⟩

lemma (**in** *Ring*) *tnm-mem*:⟦ *R module M*; *R module N*; FM_R *P M N*; *m* ∈ *carrier M*;
n ∈ *carrier N*⟧ $\Longrightarrow$ *tnm R P M N* (*m*, *n*) ∈ *carrier* (*M* $_P\bigotimes_R$ *N*)
⟨*proof*⟩

definition
tensor-elem :: [('*r*, '*m*) *Ring-scheme*, (('*a* ∗ '*b*), '*r*, '*m1*) *Module-scheme*,
('*a*, '*r*, '*m1*) *Module-scheme*, ('*b*, '*r*, '*m1*) *Module-scheme*] ⇒ '*a* ⇒ '*b*
⇒ ('*a* ∗ '*b*) *set* **where**
tensor-elem R P M N m n = *tnm R P M N* (*m*, *n*)

abbreviation
TNSELEM (‹(6- _,_⊗_,_/ _)› [100,100,100,100,100,101]101) **where**
m $_{R,P\otimes M,N}$ *n* == *tensor-elem R P M N m n*

lemma (**in** *Ring*) *tensor-univ-propTr*:⟦*R module M*; *R module N*; FM_R *P M N*;
R module Z; *bilinear-map f R M N Z*⟧ $\Longrightarrow$
∃ *g*. *g* ∈ *mHom R P Z* ∧ (*compose* (*M* $\times_c$ *N*) *g* (*tau R M N P*)) = *f*
⟨*proof*⟩

lemma (**in** *Ring*) *tensor-univ-propTr1*:⟦*R module M*; *R module N*; FM_R *P M N*;

R module Z; *bilinear-map f R M N Z*⟧ ⟹
∃!*g*. *g*∈(*mHom R* (*M* $_P\bigotimes_R$ *N*) *Z*) ∧ (*compose* (*M* ×$_c$ *N*) *g* (*tnm R P M N*))
= *f*
⟨*proof*⟩

lemma (**in** *Ring*) *tensor-universal-property*:⟦*R module M*; *R module N*; FM_R *P M N* ⟧
⟹ *universal-property R M N* (*M* $_P\bigotimes_R$ *N*) (*tnm R P M N*)
⟨*proof*⟩

end