

Wieferich–Kempner Theorem

Jamie Chen

March 17, 2025

Abstract

This document presents a formalised proof of the Wieferich–Kempner Theorem, stating that all nonnegative integers can be expressed as the sum of nine nonnegative cubes. The source of the proof is the book “Additive Number Theory: The Classical Bases” by Melvyn B. Nathanson [2].

Contents

| | | |
|----------|----------------------------------|-----------|
| 1 | Technical Lemmas | 3 |
| 1.1 | Lemma 2.1 in [2] | 3 |
| 1.2 | Lemma 2.2 in [2] | 5 |
| 1.3 | Lemma 2.3 in [2] | 7 |
| 1.4 | Lemma 2.4 in [2] | 11 |
| 2 | Wieferich–Kempner Theorem | 13 |

```

theory Wieferich-Kempner
  imports
    HOL-Number-Theory.Cong
    HOL.Real
    HOL.NthRoot
    HOL.Transcendental
    HOL-Library.Code-Target-Nat
    Three-Squares.Three-Squares
    Main
begin

fun sumpow :: nat ⇒ nat list ⇒ nat where [code-computation-unfold, coercion-enabled]:
  sumpow p l = fold (+) (map (λx. x^p) l) 0
declare sumpow.simps[code]

definition is-sumpow :: nat ⇒ nat ⇒ nat ⇒ bool
  where is-sumpow p n m ≡ ∃ l. length l = n ∧ m = sumpow p l

```

1 Technical Lemmas

We show four lemmas used in the main theorem.

1.1 Lemma 2.1 in [2]

```

lemma sum-of-6-cubes:
  fixes A m :: nat
  assumes mLessASq: m ≤ A2
  assumes mIsSum3Sq: is-sumpow 2 3 m
  shows is-sumpow 3 6 (6 * A * (A2 + m))
proof –
  from mIsSum3Sq obtain l where length l = 3 and m = sumpow 2 l
  using is-sumpow-def by blast
  then obtain m1 m2 m3 where l = [m1, m2, m3]
  by (smt (verit, best) Suc-length-conv length-0-conv numeral-3-eq-3)

  obtain il where il = map (int) l by auto
  obtain iA where iA = (int A) by auto
  obtain im where im = (int m) by auto

  have fold (+) (map power2 l) 0 = m12 + m22 + m32
  using ⟨l = [m1, m2, m3]⟩ by simp
  hence m = m12 + m22 + m32
  using ⟨length l = 3⟩ ⟨m = sumpow 2 l⟩ sumpow.simps by presburger
  obtain im1 im2 im3 where il = [im1, im2, im3]
  by (simp add: ⟨il = map int l⟩ ⟨l = [m1, m2, m3]⟩)
  hence im1 = int m1 and im2 = int m2 and im3 = int m3

```

using $\langle il = \text{map int } l \rangle \langle l = [m_1, m_2, m_3] \rangle$ **by** *auto*

obtain $x_1 x_2 x_3$ **where** $[x_1, x_2, x_3] = \text{map } (\lambda x. A+x) l$
by (*simp add: $\langle l = [m_1, m_2, m_3] \rangle$*)

obtain $x_4 x_5 x_6$ **where** $[x_4, x_5, x_6] = \text{map } (\lambda x. A-x) l$
by (*simp add: $\langle l = [m_1, m_2, m_3] \rangle$*)

obtain ns **where** $ns = [x_1, x_2, x_3, x_4, x_5, x_6]$ **by** *auto*

have $\forall x \in \text{set } l. x \geq 0$
by *auto*

hence $\forall x \in \text{set } l. x^2 \leq m$
using $\langle l = [m_1, m_2, m_3] \rangle \langle m = m_1^2 + m_2^2 + m_3^2 \rangle$ *power2-nat-le-imp-le* **by** *auto*

have $\forall x \in \text{set } il. x \geq 0$
by (*simp add: $\langle il = \text{map int } l \rangle$*)

hence $\forall x \in \text{set } il. x^2 \leq im$
using $\langle im = \text{int } m \rangle \langle il = \text{map int } l \rangle \langle l = [m_1, m_2, m_3] \rangle \langle m = m_1^2 + m_2^2 + m_3^2 \rangle$ *power2-nat-le-imp-le*
by *auto*

have $\forall x \in \text{set } il. iA + x \geq 0$
using $\langle iA = \text{int } A \rangle \langle il = \text{map int } l \rangle$ **by** *auto*

have $\text{int } (x_1) = iA + im_1$
using $\langle [x_1, x_2, x_3] = \text{map } ((+) A) l \rangle \langle iA = \text{int } A \rangle \langle il = [im_1, im_2, im_3] \rangle \langle il = \text{map int } l \rangle$ **by** *force*

have $\text{int } (x_2) = iA + im_2$
using $\langle [x_1, x_2, x_3] = \text{map } ((+) A) l \rangle \langle iA = \text{int } A \rangle \langle il = [im_1, im_2, im_3] \rangle \langle il = \text{map int } l \rangle$ **by** *force*

have $\text{int } (x_3) = iA + im_3$
using $\langle [x_1, x_2, x_3] = \text{map } ((+) A) l \rangle \langle iA = \text{int } A \rangle \langle il = [im_1, im_2, im_3] \rangle \langle il = \text{map int } l \rangle$ **by** *force*

have $A \geq m_1$
by (*metis $\langle m = m_1^2 + m_2^2 + m_3^2 \rangle$ add-leE mLessASq power2-nat-le-eq-le*)

hence $iA - im_1 \geq 0$
using $\langle iA = \text{int } A \rangle \langle il = [im_1, im_2, im_3] \rangle \langle il = \text{map int } l \rangle \langle l = [m_1, m_2, m_3] \rangle$
by *auto*

hence $\text{int } x_4 = iA - im_1$
using $\langle [x_4, x_5, x_6] = \text{map } ((-) A) l \rangle \langle iA = \text{int } A \rangle \langle im_1 = \text{int } m_1 \rangle \langle l = [m_1, m_2, m_3] \rangle$ **by** *auto*

have $A \geq m_2$
by (*metis $\langle m = m_1^2 + m_2^2 + m_3^2 \rangle$ add-leE mLessASq power2-nat-le-eq-le*)

hence $iA - im_2 \geq 0$
using $\langle iA = \text{int } A \rangle \langle il = [im_1, im_2, im_3] \rangle \langle il = \text{map int } l \rangle \langle l = [m_1, m_2, m_3] \rangle$
by *auto*

hence $\text{int } x_5 = iA - im_2$
using $\langle [x_4, x_5, x_6] = \text{map } ((-) A) l \rangle \langle iA = \text{int } A \rangle \langle im_2 = \text{int } m_2 \rangle \langle l = [m_1, m_2, m_3] \rangle$ **by** *auto*

have $A \geq m_3$
by (*metis $\langle m = m_1^2 + m_2^2 + m_3^2 \rangle$ add-leE mLessASq power2-nat-le-eq-le*)

hence $iA - im_3 \geq 0$
using $\langle iA = \text{int } A \rangle \langle il = [im_1, im_2, im_3] \rangle \langle il = \text{map int } l \rangle \langle l = [m_1, m_2, m_3] \rangle$
by auto
hence $\text{int } x_6 = iA - im_3$
using $\langle [x_4, x_5, x_6] = \text{map } ((-) A) l \rangle \langle iA = \text{int } A \rangle \langle im_3 = \text{int } m_3 \rangle \langle l = [m_1, m_2, m_3] \rangle$ **by auto**

have $6 * A * (A^2 + m) = 6 * A * (A^2 + \text{sumpow } 2 l)$
by $(\text{simp add: } \langle m = \text{sumpow } 2 l \rangle)$
have $\text{distr: } \dots = 6 * A \wedge^3 + 6 * A * (\text{sumpow } 2 l)$
by $(\text{simp add: distrib-left power2-eq-square power3-eq-cube})$
have $\dots = 6 * A * (A^2 + m_1^2 + m_2^2 + m_3^2)$
using $\langle m = m_1^2 + m_2^2 + m_3^2 \rangle \langle m = \text{sumpow } 2 l \rangle$ **by** $(\text{metis distr group-cancel.add1})$
hence $\text{expanded: int } \dots = 6 * iA * (iA^2 + im_1^2 + im_2^2 + im_3^2)$
using $\langle iA = \text{int } A \rangle \langle im_1 = \text{int } m_1 \rangle \langle im_2 = \text{int } m_2 \rangle \langle im_3 = \text{int } m_3 \rangle$ **by simp**
have $\text{sixcubes: } \dots = (iA + im_1) \wedge^3 + (iA - im_1) \wedge^3 +$
 $(iA + im_2) \wedge^3 + (iA - im_2) \wedge^3 +$
 $(iA + im_3) \wedge^3 + (iA - im_3) \wedge^3$
by Groebner-Basis.algebra

have $\dots = \text{int } x_1 \wedge^3 + \text{int } x_2 \wedge^3 + \text{int } x_3 \wedge^3 + \text{int } x_4 \wedge^3 + \text{int } x_5 \wedge^3 + \text{int } x_6 \wedge^3$
using $\langle \text{int } x_1 = iA + im_1 \rangle \langle \text{int } x_2 = iA + im_2 \rangle \langle \text{int } x_3 = iA + im_3 \rangle$
 $\langle \text{int } x_4 = iA - im_1 \rangle \langle \text{int } x_5 = iA - im_2 \rangle \langle \text{int } x_6 = iA - im_3 \rangle$ **by simp**
hence $6 * A * (A^2 + m) = x_1 \wedge^3 + x_2 \wedge^3 + x_3 \wedge^3 + x_4 \wedge^3 + x_5 \wedge^3 + x_6 \wedge^3$
using $\text{distr } \langle m = \text{sumpow } 2 l \rangle \langle m = m_1^2 + m_2^2 + m_3^2 \rangle$ $\text{expanded of-nat-eq-iff}$
sixcubes

by $(\text{smt (verit) of-nat-add of-nat-power})$
have $\text{map } (\lambda x. x \wedge^3) ns = [x_1 \wedge^3, x_2 \wedge^3, x_3 \wedge^3, x_4 \wedge^3, x_5 \wedge^3, x_6 \wedge^3]$
by $(\text{simp add: } \langle ns = [x_1, x_2, x_3, x_4, x_5, x_6] \rangle)$
hence $\text{sumpow } 3 ns = x_1 \wedge^3 + x_2 \wedge^3 + x_3 \wedge^3 + x_4 \wedge^3 + x_5 \wedge^3 + x_6 \wedge^3$
by $(\text{simp add: } \langle ns = [x_1, x_2, x_3, x_4, x_5, x_6] \rangle)$
hence $6 * A * (A^2 + m) = \text{sumpow } 3 ns$
using $\langle 6 * A * (A^2 + m) = x_1 \wedge^3 + x_2 \wedge^3 + x_3 \wedge^3 + x_4 \wedge^3 + x_5 \wedge^3 +$
 $x_6 \wedge^3 \rangle$ **by presburger**
hence $\text{length } ns = 6 \wedge 6 * A * (A^2 + m) = \text{sumpow } 3 ns$
by $(\text{simp add: } \langle ns = [x_1, x_2, x_3, x_4, x_5, x_6] \rangle)$
then show ?thesis
using is-sumpow-def by blast

qed

1.2 Lemma 2.2 in [2]

lemma *if-cube-cong-then-cong*:

fixes $t :: \text{nat}$

fixes $b_1 b_2 :: \text{int}$

assumes $\text{odd: odd } b_1 \wedge \text{odd } b_2$

assumes $b_1 > 0 \wedge b_2 > 0$

assumes $t \text{Geq1: } t \geq 1$

assumes $[b_1 \wedge^3 = b_2 \wedge^3] \pmod{2 \wedge^t}$

shows $[b_1 \neq b_2] \pmod{2 \wedge^t} \implies [b_1 \wedge^3 \neq b_2 \wedge^3] \pmod{2 \wedge^t}$

proof –
have $[b_2^3 - b_1^3 = 0] \pmod{2^t}$
using $\langle [b_1^3 = b_2^3] \pmod{2^t} \rangle$ *cong-diff-iff-cong-0 cong-sym-eq* **by** *blast*
have $(b_2 - b_1) * (b_2^2 + b_2 * b_1 + b_1^2) = b_2^3 - b_1^3$
by *Groebner-Basis.algebra*
hence $[(b_2 - b_1) * (b_2^2 + b_2 * b_1 + b_1^2) = 0] \pmod{2^t}$
using $\langle [b_2^3 - b_1^3 = 0] \pmod{2^t} \rangle$ **by** *auto*
have *coprime* $(b_2^2 + b_2 * b_1 + b_1^2) (2^t)$
by *(simp add: odd)*
hence $[b_2 - b_1 = 0] \pmod{2^t}$
by *(metis $\langle [(b_2 - b_1) * (b_2^2 + b_2 * b_1 + b_1^2) = 0] \pmod{2^t} \rangle$ cong-mult-rcancel mult-zero-left)*
hence $[b_1 = b_2] \pmod{2^t}$
using *cong-diff-iff-cong-0 cong-sym-eq* **by** *blast*
assume $[b_1 \neq b_2] \pmod{2^t}$
then show *?thesis*
using $\langle [b_1 = b_2] \pmod{2^t} \rangle$ **by** *auto*
qed

lemma *every-odd-nat-cong-cube*:

fixes $t w :: nat$
assumes *tPositive*: $t \geq 1$
assumes *wOdd*: *odd* w
shows $\exists b. \text{odd } b \wedge [w = b^3] \pmod{2^t}$
proof *(rule ccontr)*
assume $\neg ?thesis$
hence $\forall b. \text{odd } b \longrightarrow [w \neq b^3] \pmod{2^t}$
by *blast*
obtain $b :: nat$ **where** *odd* b
using *odd-numeral* **by** *blast*
hence $[w \neq b^3] \pmod{2^t}$
by *(simp add: $\langle \forall b. \text{odd } b \longrightarrow [w \neq b^3] \pmod{2^t} \rangle$)*
obtain $bSet :: nat \text{ set}$ **where** *bSet-def*: $bSet = \{x. \text{odd } x \wedge x < 2^t\}$
by *auto*
obtain w' **where** *w'-def*: $w' = w \text{ mod } 2^t$
by *auto*
hence *odd* w'
by *(metis dvd-mod-imp-dvd dvd-power order-less-le-trans tPositive wOdd zero-less-one)*
have $w' < 2^t$
by *(simp add: $\langle w' = w \text{ mod } 2^t \rangle$)*
hence $w' \in bSet$
by *(simp add: $\langle bSet = \{x. \text{odd } x \wedge x < 2^t\} \rangle$ $\langle \text{odd } w' \rangle$)*
define $bSetMinusW' :: nat \text{ set}$ **where** *bSetMinusW'* = $\{x. x \in bSet \wedge x \neq w'\}$

have $\forall x \in bSet. [w \neq x^3] \pmod{2^t}$
using $\langle \forall b. \text{odd } b \longrightarrow [w \neq b^3] \pmod{2^t} \rangle$ $\langle bSet = \{x. \text{odd } x \wedge x < 2^t\} \rangle$ **by** *blast*
have $\forall x \in bSet. \text{odd } (x^3)$
using *bSet-def* **by** *simp*

have $even (2^{\wedge}t)$
using $tPositive$ **by** $fastforce$
hence $\forall x \in bSet. odd (x^{\wedge}3 \bmod 2^{\wedge}t)$
using $\langle \forall x \in bSet. odd (x^{\wedge}3) \rangle dvd-mod-iff$ **by** $blast$
hence $\forall x \in bSet. (x^{\wedge}3 \bmod 2^{\wedge}t) \in bSet$
by $(simp\ add: \langle bSet = \{x. odd\ x \wedge x < 2^{\wedge}t\} \rangle)$
hence $\forall x \in bSet. (x^{\wedge}3 \bmod 2^{\wedge}t) \in bSetMinusW'$
using $\langle \forall x \in bSet. [w \neq x^{\wedge}3] \pmod{2^{\wedge}t} \rangle bSetMinusW'-def\ w'-def\ cong-def$
by $(metis\ (mono-tags,\ lifting)\ mem-Collect-eq\ unique-euclidean-semiring-class.cong-def)$
hence $\forall x \in bSet. \exists y \in bSetMinusW'. [y = x^{\wedge}3] \pmod{2^{\wedge}t}$
using $cong-mod-right\ cong-refl$ **by** $blast$
hence $\forall x \in bSet. \exists! y \in bSetMinusW'. [y = x^{\wedge}3] \pmod{2^{\wedge}t}$
by $(metis\ (no-types,\ lifting)\ bSet-def\ bSetMinusW'-def\ unique-euclidean-semiring-class.cong-def$

 $mem-Collect-eq\ mod-less)$
have $\forall x_1 \in bSet. \forall x_2 \in bSet. odd\ x_1 \wedge odd\ x_2$
using $bSet-def$ **by** $auto$

hence $\forall x_1 \in bSet. \forall x_2 \in bSet. [x_1 \neq x_2] \pmod{2^{\wedge}t} \longrightarrow [x_1^{\wedge}3 \neq x_2^{\wedge}3] \pmod{2^{\wedge}t}$
by $(metis\ (mono-tags)\ cong-int-iff\ even-of-nat\ if-cube-cong-then-cong\ odd-pos\ of-nat-0-less-iff$
 $of-nat-numeral\ of-nat-power\ tPositive)$
hence $\forall x_1 \in bSet. \forall x_2 \in bSet. x_1 \neq x_2 \longrightarrow [x_1^{\wedge}3 \neq x_2^{\wedge}3] \pmod{2^{\wedge}t}$
using $bSet-def\ cong-less-modulus-unique-nat$ **by** $blast$
hence $inj-on\ (\lambda x. x^{\wedge}3 \bmod 2^{\wedge}t)\ bSet$
by $(meson\ inj-onI\ unique-euclidean-semiring-class.cong-def)$

have $bSetMinusW' \subset bSet$
using $bSetMinusW'-def\ \langle w' \in bSet \rangle$ **by** $blast$
moreover **hence** $card\ bSetMinusW' < card\ bSet$
by $(simp\ add: bSet-def\ psubset-card-mono)$
moreover **have** $(\lambda x. x^{\wedge}3 \bmod 2^{\wedge}t) ' bSet \subseteq bSetMinusW'$
using $\langle \forall x \in bSet. x^{\wedge}3 \bmod 2^{\wedge}t \in bSetMinusW' \rangle$ **by** $blast$
ultimately **have** $card\ ((\lambda x. x^{\wedge}3 \bmod 2^{\wedge}t) ' bSet) < card\ bSet$
by $(metis\ card.infinite\ less-zeroE\ psubset-card-mono\ subset-psubset-trans)$
hence $\neg inj-on\ (\lambda x. x^{\wedge}3 \bmod 2^{\wedge}t)\ bSet$
using $pigeonhole$ **by** $auto$
thus $False$
using $\langle inj-on\ (\lambda x. x^{\wedge}3 \bmod 2^{\wedge}t)\ bSet \rangle$ **by** $blast$
qed

1.3 Lemma 2.3 in [2]

It is this section in which we use the Three Squares Theorem AFP Entry [1].

lemma $sum-of-3-squares-exceptions$:

fixes $m::nat$

assumes $notSum3Sq: \neg is-sumpow\ 2\ 3\ m$

shows $6 * m \bmod 96 \in \{0, 72, 42, 90\}$
proof –
have $\neg(\exists l. \text{length } l = 3 \wedge m = \text{sumpow } 2 \ l)$
using *is-sumpow-def notSum3Sq* **by** *blast*
hence $\neg(\exists a \ b \ c. m = \text{sumpow } 2 \ [a, b, c])$
by (*metis length-Cons list.size(3) numeral-3-eq-3*)
hence $\neg(\exists a \ b \ c. m = \text{fold } (+) \ (\text{map } \text{power2 } [a, b, c]) \ 0)$
by *auto*
hence $\neg(\exists a \ b \ c. m = \text{fold } (+) \ [a^2, b^2, c^2] \ 0)$
by *auto*
hence $\neg(\exists a \ b \ c. m = a^2 + b^2 + c^2)$
by (*simp add: add.commute*)
then obtain $s \ t$ **where** $(m = 4^{\wedge} s * (8 * t + 7))$
using *three-squares-iff* **by** *auto*
obtain $h' :: \text{nat set}$ **where** $h' = \{0, 72, 42, 90\}$
by *auto*
consider $(s0) \ s = 0 \mid (s1) \ s = 1 \mid (s2) \ s \geq 2$
by *arith*
hence $6 * 4^{\wedge} s * (8 * t + 7) \bmod 96 \in h'$
proof *cases*
case $s0$
consider $(\text{even}) \ \text{even } t \mid (\text{odd}) \ \text{odd } t$
by *auto*
thus $6 * 4^{\wedge} s * (8 * t + 7) \bmod 96 \in h'$
proof *cases*
case *even*
obtain t' **where** $t' = t \ \text{div } 2$
by *simp*
hence $6 * 4^{\wedge} s * (8 * t + 7) = 96 * t' + 42$
using $s0$ *even* **by** *fastforce*
hence $6 * 4^{\wedge} s * (8 * t + 7) \bmod 96 = 42$
by (*simp add: cong-0-iff cong-add-rcancel-0-nat*)
have $42 \in h'$
by (*simp add: ⟨h' = {0, 72, 42, 90}⟩*)
thus $6 * 4^{\wedge} s * (8 * t + 7) \bmod 96 \in h'$
by (*simp add: ⟨6 * 4^{\wedge} s * (8 * t + 7) \bmod 96 = 42⟩*)
next
case *odd*
obtain t' **where** $t' = (t - 1) \ \text{div } 2$
by *simp*
hence $6 * 4^{\wedge} s * (8 * t + 7) = 6 * (8 * (2 * t' + 1) + 7)$
by (*simp add: s0 odd*)
hence $6 * 4^{\wedge} s * (8 * t + 7) \bmod 96 = 90$
using *cong-le-nat* **by** *auto*
have $90 \in h'$
by (*simp add: ⟨h' = {0, 72, 42, 90}⟩*)
thus $6 * 4^{\wedge} s * (8 * t + 7) \bmod 96 \in h'$
by (*simp add: ⟨6 * 4^{\wedge} s * (8 * t + 7) \bmod 96 = 90⟩*)
qed


```

next
  case s1
  have  $6 * 4^s * (8 * t + 7) = 96 * (2 * t + 1) + 72$ 
  using s1 by auto
  hence  $[6 * 4^s * (8 * t + 7) = 72] \pmod{96}$ 
  by (metis cong-add-rcancel-0-nat cong-mult-self-left)
  hence  $6 * 4^s * (8 * t + 7) \pmod{96} = 72 \pmod{96}$ 
  using unique-euclidean-semiring-class.cong-def by blast
  hence  $6 * 4^s * (8 * t + 7) \pmod{96} = 72$ 
  by auto
  have  $72 \in h'$ 
  by (simp add:  $\langle h' = \{0, 72, 42, 90\} \rangle$ )
  thus  $6 * 4^s * (8 * t + 7) \pmod{96} \in h'$ 
  by (simp add:  $\langle 6 * 4^s * (8 * t + 7) \pmod{96} = 72 \rangle$ )
next
  case s2
  obtain s' where  $s' = s - 2$ 
  by simp
  have  $6 * 4^s * (8 * t + 7) = 6 * 4^2 * 4^{s'} * (8 * t + 7)$ 
  by (metis  $\langle s' = s - 2 \rangle$  le-add-diff-inverse mult.assoc power-add s2)
  have  $\dots = 96 * 4^{s'} * (8 * t + 7)$ 
  by auto
  hence  $[6 * 4^s * (8 * t + 7) = 0] \pmod{96}$ 
  by (metis  $\langle 6 * 4^s * (8 * t + 7) = 6 * 4^2 * 4^{s'} * (8 * t + 7) \rangle$  cong-modulus-mult-nat
  cong-mult-self-left)
  hence  $6 * 4^s * (8 * t + 7) \pmod{96} = 0 \pmod{96}$ 
  using unique-euclidean-semiring-class.cong-def by blast
  hence  $6 * 4^s * (8 * t + 7) \pmod{96} = 0$ 
  by auto
  have  $0 \in h'$ 
  by (simp add:  $\langle h' = \{0, 72, 42, 90\} \rangle$ )
  thus  $6 * 4^s * (8 * t + 7) \pmod{96} \in h'$ 
  by (simp add:  $\langle 6 * 4^s * (8 * t + 7) \pmod{96} = 0 \rangle$ )
qed
thus ?thesis
  by (metis  $\langle h' = \{0, 72, 42, 90\} \rangle$   $\langle m = 4^s * (8 * t + 7) \rangle$  mult.assoc)
qed

```

lemma *values-geq-22-cubed-can-be-normalised:*

```

  fixes r :: nat
  assumes rLarge:  $r \geq 10648$ 
  obtains d m where  $d \geq 0$  and  $d \leq 22$  and  $r = d^3 + 6 * m$  and is-sumpow 2
  3 m

```

proof –

define *MultiplesOf6LessThan96::nat set* **where**

```

  MultiplesOf6LessThan96 =  $\{0, 6, 12, 18, 24, 30, 36, 42, 48, 54, 60, 66, 72, 78, 84, 90\}$ 

```

have $\forall m' \in \text{MultiplesOf6LessThan96}. m' < 96$

unfolding *MultiplesOf6LessThan96-def* **by** auto

have $\forall m' \in \text{MultiplesOf6LessThan96}. m' \pmod{6} = 0$

```

  unfolding MultiplesOf6LessThan96-def by auto
  hence  $\forall m \in \text{MultiplesOf6LessThan96}. \exists n. m = 6 * n$ 
    by fastforce
  have  $\forall m' \in \text{MultiplesOf6LessThan96}. m' \bmod 96 = m'$ 
    by (simp add:  $\langle \forall m' \in \text{MultiplesOf6LessThan96}. m' < 96 \rangle$ )

  define H'::nat set where H' = {0,42,72,90}

  have  $\forall m'. 6 * m' \bmod 96 \notin H' \longrightarrow \text{is-sumpow } 2 \ 3 \ m'$ 
    unfolding H'-def using sum-of-3-squares-exceptions by blast
  hence  $\forall m'. (6 * m' \bmod 96 \in \text{MultiplesOf6LessThan96} - H') \longrightarrow \text{is-sumpow } 2$ 
 $3 \ m'$ 
    by fastforce

  define H::nat set where H = {6, 12, 18, 24, 30, 36, 48, 54, 60, 66, 78, 84}
  have H = MultiplesOf6LessThan96 - H'
    unfolding H-def MultiplesOf6LessThan96-def H'-def by auto
  hence  $\forall m'. 6 * m' \bmod 96 \in H \longrightarrow \text{is-sumpow } 2 \ 3 \ m'$ 
    using  $\langle \forall m'. 6 * m' \bmod 96 \in \text{MultiplesOf6LessThan96} - H' \longrightarrow \text{is-sumpow } 2$ 
 $3 \ m' \rangle$  by blast

  define D::nat set where D = {0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22}
  have  $\forall d \in D. d \geq 0 \wedge d \leq 22$ 
    using D-def by auto
  define residue::nat set where residue = {x.  $\exists h \in H. \exists d \in D. x = (d \wedge 3 + h)$ 
  mod 96}

  have  $\forall x < 10. x \in \text{residue}$ 
    unfolding residue-def H-def D-def by auto
  moreover have  $\forall x \geq 10. x < 19 \longrightarrow x \in \text{residue}$ 
    unfolding residue-def H-def D-def by auto
  moreover have  $\forall x \geq 19. x < 28 \longrightarrow x \in \text{residue}$ 
    unfolding residue-def H-def D-def by auto
  moreover have  $\forall x \geq 28. x < 37 \longrightarrow x \in \text{residue}$ 
    unfolding residue-def H-def D-def by auto
  moreover have  $\forall x \geq 37. x < 46 \longrightarrow x \in \text{residue}$ 
    unfolding residue-def H-def D-def by auto
  moreover have  $\forall x \geq 46. x < 55 \longrightarrow x \in \text{residue}$ 
    unfolding residue-def H-def D-def by auto
  moreover have  $\forall x \geq 55. x < 64 \longrightarrow x \in \text{residue}$ 
    unfolding residue-def H-def D-def by auto
  moreover have  $\forall x \geq 64. x < 73 \longrightarrow x \in \text{residue}$ 
    unfolding residue-def H-def D-def by auto
  moreover have  $\forall x \geq 73. x < 82 \longrightarrow x \in \text{residue}$ 
    unfolding residue-def H-def D-def by auto
  moreover have  $\forall x \geq 82. x < 91 \longrightarrow x \in \text{residue}$ 
    unfolding residue-def H-def D-def by auto
  moreover have  $\forall x \geq 91. x < 96 \longrightarrow x \in \text{residue}$ 
    unfolding residue-def H-def D-def by auto

```

```

ultimately have  $\forall x < 96. x \in \text{residue}$ 
  by (metis leI)

have  $\forall r \in \text{residue}. \exists h \in H. \exists d \in D. r = (d^3 + h) \bmod 96$ 
  unfolding residue-def by auto
have  $\forall h \in H. \forall d \in D. (d^3 + h) \bmod 96 \in \text{residue}$ 
  by (simp add:  $\langle \forall x < 96. x \in \text{residue} \rangle$ )
have  $r \bmod 96 \in \text{residue}$ 
  by (simp add:  $\langle \forall x < 96. x \in \text{residue} \rangle$ )
have  $\forall d \in D. r \geq d^3$ 
  unfolding D-def using rLarge by auto
hence  $\exists h \in H. \exists d \in D. [d^3 + h = r] \pmod{96}$ 
  using  $\langle r \bmod 96 \in \text{residue} \rangle$ 
  by (metis  $\langle \forall r \in \text{residue}. \exists h \in H. \exists d \in D. r = (d^3 + h) \bmod 96 \rangle$  unique-euclidean-semiring-class.cong-def)
hence  $\exists h \in H. \exists d \in D. [r - d^3 = h] \pmod{96}$ 
  by (metis  $\langle \forall d \in D. d^3 \leq r \rangle$  cong-add-lcancel-nat cong-sym ordered-cancel-comm-monoid-diff-class.add-diff)
then obtain h d where  $h \in H$  and  $d \in D$  and  $[r - d^3 = h] \pmod{96}$ 
  by auto
then obtain h' where  $h' \bmod 96 = h$  and  $r - d^3 = h'$ 
  using  $\langle H = \text{MultiplesOf6LessThan96} - H' \rangle \langle \forall m' \in \text{MultiplesOf6LessThan96}. m' \bmod 96 = m' \rangle$ 
  by (simp add: unique-euclidean-semiring-class.cong-def)
have  $[h = 0] \pmod{6}$ 
  using  $\langle H = \text{MultiplesOf6LessThan96} - H' \rangle \langle \forall m \in \text{MultiplesOf6LessThan96}. m = 6 * n \rangle \langle h \in H \rangle$ 
  cong-mult-self-left by fastforce
hence  $[h' = 0] \pmod{6}$ 
  using  $\langle [r - d^3 = h] \pmod{96} \rangle \langle r - d^3 = h' \rangle$  unique-euclidean-semiring-class.cong-def
  by (metis cong-dvd-modulus-nat dvd-triv-right num-double numeral-mult)
then obtain m where  $6 * m = h'$ 
  by (metis cong-0-iff dvd-def)
moreover hence is-sumpow 2 3 m
  using  $\langle \forall m'. 6 * m' \bmod 96 \in H \longrightarrow \text{is-sumpow 2 3 } m' \rangle$  by (simp add:  $\langle h \in H \rangle \langle h' \bmod 96 = h \rangle$ )
moreover have  $d \geq 0$ 
  by auto
moreover have  $d \leq 22$ 
  by (simp add:  $\langle \forall d \in D. 0 \leq d \wedge d \leq 22 \rangle \langle d \in D \rangle$ )
ultimately show ?thesis
  by (metis  $\langle \forall d \in D. d^3 \leq r \rangle \langle d \in D \rangle \langle r - d^3 = h' \rangle$  le-add-diff-inverse that)
qed

```

1.4 Lemma 2.4 in [2]

```

partial-function (tailrec) list-builder :: nat  $\Rightarrow$  nat  $\Rightarrow$  nat list  $\Rightarrow$  nat list
  where [code-computation-unfold, coercion-enabled]:
    list-builder m n l = (if n = 0 then l else (list-builder m (n-1) (m#l)))
declare list-builder.simps[code]

```

```

partial-function(tailrec) dec-list :: nat ⇒ nat list ⇒ nat list
  where [code-computation-unfold, coercion-enabled]:
    dec-list depth l = (if (tl l) = [] then list-builder (hd l + 1) (depth+1) [] else
      if hd l ≤ hd (tl l) + 1 then dec-list (depth+1) ((hd (tl l) + 1)#(tl (tl l))) else
      list-builder (hd (tl l) + 1) (depth + 2) (tl (tl l)))
declare dec-list.simps[code]

```

```

partial-function(tailrec) sumcubepow-finder :: nat ⇒ nat list ⇒ nat list
  where [code-computation-unfold, coercion-enabled]:
    sumcubepow-finder n l = (if (sumpow 3 l < n) then
      (sumcubepow-finder n ((Suc (hd l))#(tl l)))
      else if (sumpow 3 l) = n then l else sumcubepow-finder n (dec-list 0 l))
declare sumcubepow-finder.simps[code]

```

```

lemma leq-40000-is-sum-of-9-cubes:
  fixes n :: nat
  assumes n ≤ 40000
  shows is-sumpow 3 9 n and n > 8042 → is-sumpow 3 6 n
proof –

```

```

let ?A1 = [8043 ..< 15000]
let ?A2 = [15000 ..< 23000]
let ?A3 = [23000 ..< 29000]
let ?A4 = [29000 ..< 35000]
let ?A5 = [35000 ..< 40001]

```

```

let ?test = λ n. let result = sumcubepow-finder n [0,0,0,0,0] in
  sumpow 3 result = n ∧ length result = 6
have split: ∧ n. n ≤ 40000 ⇒ n > 8042 ⇒
  n ∈ set ?A1 ∨ n ∈ set ?A2 ∨ n ∈ set ?A3 ∨ n ∈ set ?A4 ∨ n ∈ set ?A5
  unfolding set-upt by simp linarith
have ∀ n ∈ set ?A1. ?test n
  by eval
moreover have ∀ n ∈ set ?A2. ?test n
  by eval
moreover have ∀ n ∈ set ?A3. ?test n
  by eval
moreover have ∀ n ∈ set ?A4. ?test n
  by eval
moreover have ∀ n ∈ set ?A5. ?test n
  by eval
ultimately have ∀ n ≤ 40000. n > 8042 → ?test n using split by blast
then have ∀ n ≤ 40000. n > 8042 → is-sumpow 3 6 n
  using is-sumpow-def by metis
then show n > 8042 → is-sumpow 3 6 n
  using ⟨n ≤ 40000⟩ by simp
have ∀ n ≤ 40000. n > 8042 → (∃ l. length l = 6 ∧ n = sumpow 3 l)
  using is-sumpow-def ⟨∀ n ≤ 40000. n > 8042 → is-sumpow 3 6 n⟩ by simp
then have ∀ n ≤ 40000. n > 8042 → (∃ l. length (l@[0,0,0]) = 9 ∧ n = sumpow

```

```

3 (l@[0,0,0])
  by simp
then have  $\forall n \leq 40000. n > 8042 \longrightarrow is\_sumpow\ 3\ 9\ n$ 
  using is-sumpow-def by blast
have  $\forall n \leq 8042. let\ res = sumcubepow\_finder\ n\ [0,0,0,0,0,0,0,0,0]\ in$ 
   $sumpow\ 3\ res = n \wedge length\ res = 9$ 
  by eval
then have  $\forall n \leq 8042. is\_sumpow\ 3\ 9\ n$ 
  using is-sumpow-def by metis
then have  $\forall n \leq 40000. is\_sumpow\ 3\ 9\ n$ 
  using  $\langle \forall n \leq 40000. n > 8042 \longrightarrow is\_sumpow\ 3\ 9\ n \rangle$  by auto
then show is-sumpow 3 9 n
  using  $\langle n \leq 40000 \rangle$  by simp
qed

```

2 Wieferich–Kempner Theorem

Theorem 2.1 in [2]

theorem *Wieferich-Kempner*:

fixes $N :: nat$

shows $is_sumpow\ 3\ 9\ N$

proof *cases*

consider $(ge8pow10)\ N > 8^{10} \mid (leq8pow10)\ N \leq 8^{10}$

by *arith*

thus *?thesis*

proof *cases*

case $ge8pow10$

define $n::int$ **where** $n = \lfloor root\ 3\ N \rfloor$

hence $n \geq \lfloor root\ 3\ (8^{10}) \rfloor$

by *(meson floor-mono ge8pow10 less-imp-le-nat numeral-power-le-of-nat-cancel-iff*

real-root-le-iff zero-less-numeral)

have $(2::nat)^3 = 8$

by *simp*

hence $8^{10} = ((2::nat)^{10})^3$

by *simp*

have $root\ 3\ ((2::nat)^{10})^3 = (2::nat)^{10}$

by *simp*

hence $n \geq \lfloor (2::nat)^{10} \rfloor$

by *(metis $\langle 8^{10} = (2^{10})^3 \rangle \langle \lfloor root\ 3\ (8^{10}) \rfloor \leq n \rangle$ of-nat-numeral of-nat-power*

power3-eq-cube

real-root-mult)

hence $ngeq2pow10: n \geq (2::nat)^{10}$

by *auto*

obtain $k::int$ **where** $k = \lceil (\log\ 8\ (N/8))/3 \rceil - 1$

by *simp*

hence $3*k < (\log\ 8\ (N/8))$

by *linarith*

```

hence  $8 * (8 \text{ powr } (3 * k)) < N$ 
  using ge8pow10 less-log-iff by simp
have  $k + 1 \geq (\log 8 (N / 8)) / 3$ 
  by (simp add:  $\langle k = \lceil \log 8 (\text{real } N / 8) / 3 \rceil - 1 \rangle$ )
hence  $8 \text{ powr } (3 * (k + 1)) \geq 8 \text{ powr } (\log 8 (N / 8))$ 
  by simp
hence  $8 * (8 \text{ powr } (3 * (k + 1))) \geq N$ 
  using ge8pow10 by simp
have  $(N / 8) > 8^9$ 
  using ge8pow10 by simp
hence  $(\log 8 (N / 8)) > 9$ 
  by (metis less-log-of-power numeral-One numeral-less-iff of-nat-numeral semiring-norm(76))
hence  $(\log 8 (N / 8)) / 3 > 3$ 
  by simp
hence  $k \geq 3$ 
  using  $\langle k = \lceil (\log 8 (N / 8)) / 3 \rceil - 1 \rangle$  by simp

define i::int where  $i = \lfloor \text{root } 3 (N - (8 * (8 \text{ powr } (3 * k)))) \rfloor$ 
hence  $i \leq \text{root } 3 (N - (8 * (8 \text{ powr } (3 * k))))$ 
  by fastforce
have  $\text{root } 3 (N - (8 * (8 \text{ powr } (3 * k)))) \geq 0$ 
  using  $\langle 8 * (8 \text{ powr } (3 * k)) < N \rangle$  by force
hence  $i \geq 0$ 
  by (simp add: i-def)
hence  $i^3 \leq (\text{root } 3 (N - (8 * (8 \text{ powr } (3 * k))))^3$ 
  using  $\langle i \leq \text{root } 3 (N - 8 * 8 \text{ powr } (3 * k)) \rangle$  power-mono by fastforce
hence  $i^3 \leq N - (8 * (8 \text{ powr } (3 * k)))$ 
  using  $\langle 8 * 8 \text{ powr } (3 * k) < N \rangle$  by force
hence exp01:  $8 * (8 \text{ powr } (3 * k)) \leq N - i^3$ 
  by simp
have  $i + 1 > \text{root } 3 (N - (8 * (8 \text{ powr } (3 * k))))$ 
  by (simp add: i-def)
hence  $(i + 1)^3 > (\text{root } 3 (N - (8 * (8 \text{ powr } (3 * k))))^3$ 
  by (metis  $\langle 0 \leq \text{root } 3 (N - 8 * 8 \text{ powr } (3 * k)) \rangle$  of-int-power power-strict-mono zero-less-numeral)
hence  $(i + 1)^3 > (N - (8 * (8 \text{ powr } (3 * k))))$ 
  using  $\langle 8 * 8 \text{ powr } (3 * k) < N \rangle$  by force
hence exp02:  $8 * (8 \text{ powr } (3 * k)) > N - (i + 1)^3$ 
  by simp

have  $i \geq 1$ 
proof (rule ccontr)
  assume  $\neg i \geq 1$ 
  hence  $i = 0$ 
    using  $\langle i \geq 0 \rangle$  by simp
  hence  $8 * (8 \text{ powr } (3 * k)) \leq N$ 
    using  $\langle 8 * (8 \text{ powr } (3 * k)) \leq N - i^3 \rangle$  by simp
  moreover have  $8 * (8 \text{ powr } (3 * k)) + 1 > N$ 

```

using $\langle i = 0 \rangle \langle (N - (i+1))^3 < 8*(8 \text{ powr } (3*k)) \rangle$ *ge8pow10* **by** *auto*
moreover have $\forall x. x \leq N \wedge x + 1 > N \longrightarrow x = N$
by *linarith*
have $8*8^{\wedge}(\text{nat}(3*k)) = 8*(8 \text{ powr } (3*k))$
by (*smt (verit, ccfv-SIG) $\langle 3 \leq k \rangle$ powr-real-of-int*)
ultimately have $8*(8 \text{ powr } (3*k)) = N$
by (*metis nat-le-real-less nle-le of-nat-le-iff of-nat-numeral of-nat-power*
power.simps(2))
thus *False*
using $\langle 8 * 8 \text{ powr } (3 * k) < N \rangle$ **by** *simp*
qed

have $8 \text{ powr } (3*(k+1)) = (8 \text{ powr } (k+1))^3$
by (*metis of-int-mult of-int-numeral powr-ge-zero powr-numeral powr-powr*
powr-powr-swap)
have $\forall (x::\text{int}) \geq 1. x^3 - (x-1)^3 = 3*x^2 - 3*x + 1$
by *Groebner-Basis.algebra*
hence $\forall (x::\text{int}) \geq 1. x^3 - (x-1)^3 < 3*x^2$
by *auto*
have *exp03*: $\forall (x::\text{int}) \geq 1. x \leq n \longrightarrow 3*x^2 \leq 3*(\text{root } 3 \text{ } N)^2$
using *n-def* **by** (*simp add: le-floor-iff*)
have $\text{root } 3 \text{ } N \leq \text{root } 3 \text{ } (8*(8 \text{ powr } (3*(k+1))))$
using $\langle N \leq 8 * 8 \text{ powr } (3 * (k + 1)) \rangle$ **by** *force*
moreover have $\dots = (\text{root } 3 \text{ } 8)*(\text{root } 3 \text{ } ((8 \text{ powr } ((k+1)))^3))$
using $\langle 8 \text{ powr } (3*(k+1)) = (8 \text{ powr } (k+1))^3 \rangle$ *real-root-mult* **by** *simp*
moreover have $\dots = 2*(8 \text{ powr } ((k+1)))$
using *odd-real-root-power-cancel* **by** *simp*
ultimately have *exp04*: $(\text{root } 3 \text{ } N)^2 \leq (2*(8 \text{ powr } (k+1)))^2$
by (*metis of-nat-0-le-iff power-mono real-root-pos-pos-le*)
moreover hence *exp05*: $\dots = (4*(8 \text{ powr } ((k+1)))^2)$
using *four-x-squared* **by** *presburger*
moreover hence *exp06*: $\dots = (4*(8 \text{ powr } (2*(k+1))))$
by (*smt (verit) of-int-add power2-eq-square powr-add*)
moreover hence $\dots = (8*(8 \text{ powr } (2*k + 2)))/2$
by *simp*
moreover hence *exp07*: $\dots = (8 \text{ powr } (2*k + 3))/2$
by (*simp add: add commute powr-mult-base*)
ultimately have *exp08*: $3*(\text{root } 3 \text{ } N)^2 \leq (3*8 \text{ powr } (2*k + 3))/2$
by *linarith*
have *exp09*: $\forall x \geq 1. x \leq n \longrightarrow \text{real-of-int } (x^3 - (x-1)^3) < \text{real-of-int}$
 $(3*x^2)$
using $\langle \forall (x::\text{int}) \geq 1. x^3 - (x-1)^3 < 3*x^2 \rangle$ **by** *presburger*
hence $\forall x \geq 1. x \leq n \longrightarrow x^3 - (x-1)^3 < 3*(\text{root } 3 \text{ } N)^2$
using *exp03 less-le-trans*
by *fastforce*
hence *exp10*: $\forall x \geq 1. x \leq n \longrightarrow x^3 - (x-1)^3 \leq (3*8 \text{ powr } (2*k + 3))/2$
using *exp08* **by** *fastforce*

have $(\text{root } 3 \text{ } N)^3 = N$

```

    by simp
  have root 3 N < n+1
    using n-def by linarith
  hence (root 3 N)3 < (n+1)3
    by (metis of-int-power of-nat-0-le-iff real-root-pos-pos-le zero-less-numeral
power-strict-mono)

```

The following few lines have been slightly modified from the original proof to simplify formalisation. This does not affect the proof in any meaningful way.

```

  hence exp11: N - n3 < (n+1)3 - n3
    using ⟨(root 3 N)3 = N⟩ by linarith
  have exp12: ... = 3*n2 + 3*n + 1
    by Groebner-Basis.algebra
  have exp13: ... ≤ 6*n2 + 1
    using ⟨n ≥ (2::nat)10⟩ power-strict-mono by (simp add: self-le-power)
  have ... ≤ 6*(root 3 N)2 + 1
    using n-def by auto
  have ... ≤ (3*8 powr (2*k + 3)) + 1
    using ⟨3*(root 3 N)2 ≤ (3*8 powr (2*k + 3))/2⟩ by auto
  hence exp14: ... ≤ 4*8 powr (2*k + 3)
    using ⟨3 ≤ k⟩ ge-one-powr-ge-zero by auto
  hence exp15: ... ≤ 8*8 powr (3*k)
    by (smt (verit, best) ⟨3 ≤ k⟩ of-int-le-iff powr-mono)

  have 6*n2 ≤ 3*8 powr (2*k + 3)
    using ⟨6*(root 3 N)2 + 1 ≤ 3*8 powr (2*k + 3) + 1⟩ ⟨(6*n2 + 1) ≤ 6*(root
3 N)2 + 1⟩ by linarith
  hence i ≤ n-1
    using exp12 exp14 exp13 exp01 i-def n-def exp11 exp15
  by (smt (verit, ccfv-SIG) floor-mono of-int-less-iff real-root-le-iff zero-less-numeral)
  hence N - (i+1)3 ≤ 8*8 powr (3*k)
    using ⟨(int N - (i + 1)3) < 8 * 8 powr (3*k)⟩ by linarith
  hence N ≥ (i+1)3
    using exp04 exp06 exp15 exp07 exp01 exp03 exp09 ⟨i ≤ n - 1⟩ ⟨0 ≤ i⟩
  by (smt (verit, ccfv-threshold) divide-cancel-right exp05 of-int-0-le-iff of-int-diff)
  have exp16: ((i+1)3 - i3) ≤ 3*8 powr (2*k + 3)
    using exp10 exp05 exp04 exp06 exp07 exp03 exp09 ⟨i ≤ n-1⟩ ⟨0 ≤ i⟩
    by (smt (verit, ccfv-SIG) divide-cancel-right powr-non-neg)
  have (i3 - (i-1)3) ≤ 3*8 powr (2*k + 3)
    using exp10 exp05 exp04 exp06 exp07 exp03 exp09 ⟨i ≤ n-1⟩ ⟨1 ≤ i⟩
    by (smt (verit, ccfv-SIG) divide-cancel-right powr-non-neg)
  have i3 > (i-1)3
    by (smt (verit, best) power-minus-Bit1 power-mono-iff zero-less-numeral)
  have N - (i-1)3 = ((N - (i-1)3) - (N - i3)) + ((N - i3) - (N - (i+1)3))
+ (N - (i+1)3)
    by simp
  have ... = (i3 - (i-1)3) + ((i+1)3 - i3) + (N - (i+1)3)
    by simp

```



```

have exp17: ... < 3*8 powr (2*k + 3) + 8*8 powr (3*k)
  using <N - (i+1)^3 ≤ 8*8 powr (3*k)> <i ≤ n-1> <1 ≤ i> <0 ≤ i> exp08
exp10 exp03 exp09
  by (smt (verit) field-sum-of-halves of-int-add of-int-diff power-strict-mono
zero-less-numeral)
have exp18: ... ≤ 11*(8 powr (3*k))
  by (simp add: <3 ≤ k>)
have (even (i^3) ∧ odd ((i-1)^3)) ∨ (even ((i-1)^3) ∧ odd (i^3))
  by simp
hence (even (N - i^3) ∧ odd (N - (i-1)^3)) ∨ (even (N - (i-1)^3) ∧ odd
(N - i^3))
  by auto
hence (even (N - (i-1)^3)) → odd (N - i^3)
  by blast
obtain a::nat where a-def: if odd (N - (i-1)^3) then a = i-1 else a = i
  by (metis <0 ≤ i> <1 ≤ i> diff-ge-0-iff-ge nonneg-int-cases)
have a = int a
  by simp
consider (odd) odd (N - (i-1)^3) | (even) even (N - (i-1)^3)
  by blast
hence odd (N - a^3)
proof cases
  case odd
    have int a = i-1
      using a-def odd by simp
    have odd (N - (i-1)^3)
      using odd by simp
    hence odd (N - (int a)^3)
      using <a = i-1> by simp
    have a^3 ≤ N
      using <N ≥ (i+1)^3> <int a = i-1> power-mono <i ≥ 1>
      by (smt (verit, ccfv-SIG) of-nat-le-of-nat-power-cancel-iff)
    hence N - a^3 = N - (int a)^3
      by (simp add: of-nat-diff)
    thus ?thesis
      using <odd (N - (int a)^3)> by presburger
  next
    case even
      have a = i
        using a-def even by simp
      have odd (N - i^3)
        using <(even (N - (i-1)^3)) → odd (N - i^3)> even by simp
      hence odd (N - (int a)^3)
        using <a = i> by simp
      have a^3 ≤ N
        using <N ≥ (i+1)^3> <int a = i> power-mono <i ≥ 1>
        by (smt (verit, ccfv-SIG) of-nat-le-of-nat-power-cancel-iff)
      hence N - a^3 = N - (int a)^3
        by (simp add: of-nat-diff)

```

```

thus ?thesis
  using ⟨odd (N - (int a)^3)⟩ by presburger
qed
hence odd (nat (N - a^3))
  using nat-int by presburger
moreover have 3*(nat k) ≥ 1
  using ⟨3 ≤ k⟩ by auto
ultimately have ∃ b. odd b ∧ [(nat (N - a^3)) = b^3] (mod 2^(3*(nat k)))
  using every-odd-nat-cong-cube by presburger
  then obtain b'::nat where odd b' and [(nat (N - a^3)) = b'^3] (mod
2^(3*(nat k)))
  by auto
define b::nat where b = b' mod (8^(nat k))
have [nat (N - a^3) ≠ 0] (mod 2^(3*(nat k)))
  by (meson ⟨1 ≤ 3 * nat k⟩ ⟨odd (nat (int (N - a^3)))⟩ cong-0-iff even-power

      cong-dvd-modulus-nat dvd-refl order-less-le-trans zero-less-one)
hence b > 0
  by (metis ⟨1 ≤ 3 * nat k⟩ ⟨2^3 = 8⟩ b-def ⟨odd b'⟩ dvd-power gcd-nat.trans

      mod-greater-zero-iff-not-dvd order-less-le-trans power-mult zero-less-one)
hence b^3 > 0
  by simp

have b < 8 powr k
  by (simp add: b-def powr-int)
hence b^3 < (8 powr (k))^3
  using power-strict-mono by fastforce
hence b^3 < 8 powr (3*k)
  by (metis mult.commute of-int-mult of-int-numeral powr-ge-zero powr-numeral
powr-powr)
have N - a^3 ≥ 8*8 powr (3*k)
  using ⟨(i - 1)^3 < i^3⟩ ⟨8 * 8 powr (3*k) ≤ N - i^3⟩
by (smt (verit, best) a-def int-ops(6) of-int-less-iff of-int-of-nat-eq of-nat-power)

have exp19: 7*8 powr (3*k) = 8*8 powr (3*k) - 8 powr (3*k)
  by simp
have exp20: ... < (N - a^3) - b^3
  using ⟨N - a^3 ≥ 8*8 powr (3*k)⟩ ⟨b^3 < 8 powr (3*k)⟩ by linarith
hence ... < N - a^3
  using ⟨0 < b^3⟩ ⟨(b^3) < 8 powr (3 * k)⟩ by linarith
have exp21: ... < 11*8 powr (3*k)
  using a-def ⟨(i-1)^3 < i^3⟩ exp16 exp01 exp17 exp18 exp02
by (smt (verit) of-int-less-iff of-int-of-nat-eq of-nat-diff of-nat-le-of-nat-power-cancel-iff

      of-nat-power)

have [N - a^3 - b^3 = 0] (mod (8^(nat k)))
  by (smt (verit, del-insts) ⟨2^3 = 8⟩ ⟨nat (N - a^3) = b'^3⟩ (mod 2^(3*nat

```

```

k))› b-def
  unique-euclidean-semiring-class.cong-def cong-diff-iff-cong-0-nat diff-is-0-eq'
nat-int
  nle-le power-mod power-mult)
define q::real where q = (N - a^3 - b^3)/(8 powr k)
have (N - a^3 - b^3)/(8 powr k) ≥ 1
by (smt (verit, ccfv-SIG) ⟨1 ≤ 3 * nat k⟩ exp20 le-divide-eq-1-pos less-numeral-extra(1)

  nat-0-less-mult-iff of-int-less-iff order-less-le-trans powr-gt-zero powr-less-cancel-iff

  zero-less-nat-eq)
hence q ≠ 0
  using q-def by auto
moreover have (N - a^3 - b^3) ≠ 0
  using exp20 by auto

moreover have (8 powr k) ≠ 0
  using power-not-zero by auto
ultimately have exp22: q*(8 powr k) = (N - a^3 - b^3)
  using q-def by auto

have exp23: (8::nat)^(nat k) = 8 powr k
  using ⟨1 ≤ 3 * nat k⟩ powr-int by auto
hence q = ⌊q⌋
  using ⟨[N - a^3 - b^3 = 0] (mod (8::nat)^(nat k))⟩ q-def
  by (metis cong-0-iff floor-of-nat of-int-of-nat-eq real-of-nat-div)
have 7*8 powr (3*k) < q*8 powr (k)
  using exp19 exp20 exp22
  by presburger
hence q > 7*8 powr (3*k)/(8 powr k)
  by (simp add: divide-less-eq)
hence ... > 7*(8 powr (3*k - k))
  using powr-diff by (metis of-int-diff times-divide-eq-right)
hence ... > 7 * 8 powr (2*k)
  by simp

have q*8 powr k < 11*8 powr (3*k)
  using exp22 exp21 by linarith
hence q < 11*8 powr (3*k) / 8 powr k
  by (simp add: pos-less-divide-eq)
hence q < 11*8 powr (3*k - k)
  using powr-diff by (metis of-int-diff times-divide-eq-right)
hence q < 11*8 powr (2*k)
  by simp
have exp24:(8::nat)^(2*(nat k)) = 8 powr (2*k)
  using ⟨3 ≤ k⟩ powr-int
by (metis exp23 of-int-mult of-int-numeral of-nat-power power-even-eq powr-ge-zero

  powr-numeral powr-powr powr-powr-swap)

```

```

hence 6*(8::nat)^(2*(nat k)) = 6*8 powr (2*k)
  by simp
have 6*(8 powr (2* k)) < 7*(8 powr (2*k))
  by simp
hence [q] - 6*(8 ^ (2*(nat k))) > 0
  using ⟨7 * 8 powr (2 * k) < q⟩ ⟨q = [q]⟩
proof -
  have 6 * 8 powr (2*k) < 7 * 8 powr (2*k)
    by simp
  have 7 * 8 powr (2 * k) < q
    using ⟨7 * 8 powr (2 * k) < q⟩ by fastforce
  hence 6 * 8 powr (2 * k) < q
    using ⟨6 * 8 powr (2*k) < 7 * 8 powr (2*k)⟩ by linarith
  thus ?thesis
  by (metis (no-types) ⟨q = [q]⟩ exp24 of-int-0-less-iff of-int-diff of-int-eq-numeral-power-cancel-iff
    of-int-mult of-int-numeral real-of-nat-eq-numeral-power-cancel-iff diff-gt-0-iff-gt)
qed
then obtain r::nat where r = [q] - 6*(8 ^ (2*(nat k)))
  by (metis zero-less-imp-eq-int)
hence r = q - 6*(8 powr (2*k))
  by (metis ⟨q = [q]⟩ exp24 of-int-diff of-int-mult of-int-numeral of-int-of-nat-eq

    real-of-nat-eq-numeral-power-cancel-iff)
have (22::nat)^3 < 8^6
  using power-mono by simp
have ... ≤ 8^(nat (k*2))
  by (smt (verit, best) ⟨k ≥ 3⟩ nat-mono nat-numeral numeral-Bit0 numeral-Bit1

    numeral-eq-one-iff numeral-less-iff power-increasing-iff)
have ... = 8 powr (2*k)
  by (smt (verit, best) ⟨3 ≤ k⟩ numeral-Bit0 numeral-One of-nat-1 of-nat-numeral
of-nat-power powr-int)
have ... < r
  using ⟨r = q - 6*(8 powr (2*k))⟩ ⟨q > 7*8 powr (2*k)⟩ by auto
have ... < 5*(8 powr (2*k))
  using ⟨q < 11*8 powr (2*k)⟩ ⟨r = q - 6*(8 powr (2*k))⟩ by auto

have r > 22^3
  using ⟨(22::nat)^3 < 8^6⟩ ⟨8^6 ≤ 8^(nat (k*2))⟩ ⟨real (8^(nat (k*2))) = 8
powr (2*k)⟩ ⟨8 powr (2*k) < r⟩
  by linarith
hence r ≥ 10648
  by simp
then obtain d m where d ≥ 0 and d ≤ 22 and r = d^3 + 6*m and
is-sumpow 2 3 m
  using values-geq-22-cubed-can-be-normalised by auto

define A::nat where A = 8^(nat k)
have m ≤ r/6

```

```

    using ⟨r = d^3 + 6*m⟩ ⟨d ≥ 0⟩ by linarith
    have ... < (5*8^(nat (2*k)))/6
    by (smt (verit, ccfv-SIG) ⟨3 ≤ k⟩ ⟨r < 5*(8 powr (2*k))⟩ divide-strict-right-mono
powr-int)
    have ... < 8^(nat (2*k))
    by simp
    have ... = A^2
    by (metis A-def ⟨real (8^(2*nat k)) = 8 powr (2*k)⟩ ⟨real (8 ^ nat(k*2)) =
8 powr (2*k)⟩
mult.commute power-mult real-of-nat-eq-numeral-power-cancel-iff)

define c::nat where c = d*2^(nat k)
have N = a^3 + b^3 + (8^(nat k))*q
by (smt (verit, del-Insts) ⟨1 ≤ 3 * nat k⟩ ⟨N - a^3 - b^3 ≠ 0⟩ exp22
diff-is-0-eq' mult.commute nat-0-iff nat-0-less-mult-iff nat-int nle-le
of-nat-add
of-nat-diff powr-int zero-less-nat-eq zero-less-one)
moreover have ... = a^3 + b^3 + 8^(nat k)*(6*8 powr (2*k) + r)
using ⟨r = q - 6*(8 powr (2*k))⟩ by simp
moreover have ... = a^3 + b^3 + 8^(nat k)*(6*8^(nat(2*k)) + r)
proof -
have 8 powr (int 2 * k) = (8 ^ nat (k * int 2))
using ⟨real (8 ^ nat (k * 2)) = 8 powr (2 * k)⟩ by auto
thus ?thesis
by simp
qed
moreover have ... = a^3 + b^3 + (8^(nat k)*d^3) + 8^(nat k)*(6*8^(nat(2*k))
+ 6*m)
by (simp add: ⟨r = d^3 + 6*m⟩ add-mult-distrib2)
moreover have ... = a^3 + b^3 + ((2^(nat k))*d)^3 + 8^(nat k)*(6*8^(nat(2*k))
+ 6*m)
by (smt (verit) ⟨2 ^ 3 = 8⟩ mult.commute power-mult power-mult-distrib)
moreover have ... = a^3 + b^3 + c^3 + 6*A*(A^2 + m)
by (smt (verit) A-def c-def exp24 ⟨real (8 ^ nat(k*2)) = 8 powr (2*k)⟩
add-mult-distrib2
mult.assoc mult.commute of-nat-eq-iff power-even-eq)
ultimately have N = a^3 + b^3 + c^3 + 6*A*(A^2 + m)
using of-nat-eq-iff by metis
have is-sumpow 3 6 (6*A*(A^2 + m))
proof -
have m ≤ A^2
using ⟨8 ^ nat (2*k) = real (A^2)⟩ ⟨m ≤ r/6⟩ ⟨real r / 6 < 5*8^nat (2 *
k)/6⟩ by linarith
thus ?thesis
using sum-of-6-cubes ⟨is-sumpow 2 3 m⟩ by simp
qed
then obtain l::nat list where length l = 6 and 6*A*(A^2 + m) = sumpow 3 l
using is-sumpow-def by blast
have a^3 + b^3 + c^3 + sumpow 3 l = sumpow 3 (a#b#c#l)

```

```

    by (simp add: fold-plus-sum-list-rev)
  hence ... = N
    using ⟨6 * A * (A2 + m) = sumpow 3 l⟩ ⟨N = a3 + b3 + c3 + 6*A*(A2
+ m)⟩ by presburger
  have length (a#b#c#l) = 9
    using ⟨length l = 6⟩ by simp
  thus ?thesis
    using ⟨sumpow 3 (a#b#c#l) = N⟩ is-sumpow-def by blast
next
case leq8pow10
consider (leq40000) N ≤ 40000 | (ge40000) N > 40000
  by arith
thus ?thesis
proof (cases)
case ge40000
  define a::int where a = [(N - 10000) powr (1/3)]

```

The following inequalities differ from those in [2], which erroneously result in the false statement $a > 31$, we have corrected these mistakes. This does not affect the rest of the proof.

```

  have (N - 10000) > 30000
    using ge40000 by linarith
  hence (N - 10000) powr (1/3) ≥ (30000) powr (1/3)
    using powr-mono2 by simp
  hence a ≥ [(30000::int) powr (1/3)]
    using a-def floor-mono by simp
  have (30000::nat) > (31)3
    by simp
  hence (30000::nat) powr (1/3) > (313) powr (1/3)
    by (metis numeral-less-iff numeral-pow of-nat-numeral powr-less-mono2
zero-le-numeral
  zero-less-divide-1-iff zero-less-numeral)
  hence ... > 31
    by auto
  hence [(30000::nat) powr (1/3)] ≥ 31
    by linarith
  hence a ≥ 31
    using ⟨a ≥ [(30000::int) powr (1/3)]⟩
    by (metis nle-le of-int-numeral of-nat-numeral order-trans)
  hence nat a = a
    by simp
  have ∀(x::int) > 4. x*(x - 3) > 1
    by (simp add: less-1-mult)
  hence ∀(x::int) > 4. x2 - 3*x - 1 > 0
    by (simp add: mult.commute power2-eq-square right-diff-distrib')
  define d::int where d = (a+1)3 - a3
  moreover have ... = 3*a2 + 3*a + 1
    by Groebner-Basis.algebra
  moreover hence a2 - 3*a - 1 > 0

```

```

    using ⟨a ≥ 31⟩ ⟨∀ (x::int) > 4. x2 - 3*x - 1 > 0⟩ by simp
  moreover hence 4*a2 > 3*a2 + 3*a + 1
    by simp
  ultimately have 4*a2 > d
    by presburger
  have a < (N powr (1/3))
    by (smt (verit, best) a-def diff-less floor-eq-iff ge40000 of-nat-0-le-iff
of-nat-less-iff
    powr-less-mono2 zero-less-divide-1-iff zero-less-numeral)
  hence a powr 2 < (N powr (1/3)) powr 2
    using ⟨31 ≤ a⟩ order-less-le by fastforce
  hence 4*a2 < 4*(N powr (2/3))
    using a-def powr-powr by fastforce
  have N - (N - 10000) ≤ 10000
    by simp
  hence N - ((N - 10000) powr (1/3)) powr 3 ≤ 10000
    using powr-powr powr-one-gt-zero-iff ge40000 by force
  hence N - ⌊(N-10000) powr (1/3) + 1⌋ powr 3 < 10000
  by (smt (verit, best) powr-ge-zero powr-less-mono2 real-of-int-floor-add-one-gt)
  hence N - (a+1) powr 3 < 10000
    using a-def one-add-floor by simp
  have (a+1) powr 3 = (a+1)3
    using a-def by auto
  hence N - (a+1)3 < 10000
    using ⟨N - (a+1) powr 3 < 10000⟩ by linarith
  have ... ≤ N - (N - 10000)
    using ge40000 by auto
  moreover have ... = N - ((N - 10000) powr (1/3)) powr 3
    using powr-powr by simp
  moreover have ... ≤ N - ⌊((N - 10000) powr (1/3))⌋ powr 3
    by simp
  moreover have ... = N - a3
    using a-def by simp
  ultimately have 10000 ≤ N - a3
    by linarith
  have ... = N - (a+1)3 + d
    using d-def by simp
  moreover have ... < 10000 + 4*(N powr (2/3))
    using ⟨N - (a+1)3 < 10000⟩ ⟨4*a2 > d⟩ ⟨4*a2 < 4*(N powr (2/3))⟩
le-less-trans by linarith
  ultimately have N - a3 < 10000 + 4*(N powr (2/3))
    by presburger
  hence int (N - (nat a)3) < 10000 + 4*(N powr (2/3))
    using ⟨nat a = a⟩
    by (smt (verit, best) ⟨10000 ≤ int N - a3⟩ int-ops(6) of-int-of-nat-eq
of-nat-power)

  consider (N-min-a-cube-leq40000) N-(nat a)3 ≤ 40000 | (N-min-a-cube-ge40000)
N - (nat a)3 > 40000

```

```

    by arith
  thus ?thesis
proof (cases)
  case N-min-a-cube-leq40000
  have  $N - (\text{nat } a)^3 \geq 10000$ 
    using <math>10000 \leq N - a^3</math> int-nat-eq <math>a \geq 31</math>
  by (smt (verit) int-ops(6) numeral-Bit0 numeral-Bit1 numeral-One of-nat-1
of-nat-le-iff
    of-nat-numeral of-nat-power)
  hence is-sumpow 3 6 (N-(nat a)^3)
    using leq-40000-is-sum-of-9-cubes N-min-a-cube-leq40000 by force

  then obtain l::nat list where length l = 6 and  $N - (\text{nat } a)^3 = \text{sumpow } 3 \ l$ 
    3 l
    using is-sumpow-def by blast
  have  $0 + 0 + (\text{nat } a)^3 + \text{sumpow } 3 \ l = \text{sumpow } 3 \ ((\text{nat } a)\#0\#0\#l)$ 
    by (simp add: fold-plus-sum-list-rev)
  hence ... = N
    using <math>N - (\text{nat } a)^3 = \text{sumpow } 3 \ l</math> <math>10000 \leq N - \text{nat } a^3</math> by
presburger
  have length ((nat a)#0#0#l) = 9
    using <math>\text{length } l = 6</math> by simp
  thus ?thesis
    using <math>\text{sumpow } 3 \ ((\text{nat } a)\#0\#0\#l) = N</math> is-sumpow-def by blast
next
  case N-min-a-cube-ge40000
  define N'::nat where  $N' = N - (\text{nat } a)^3$ 
  hence  $N' > 40000$ 
    using N-min-a-cube-ge40000 by simp

  define b::int where  $b = \lfloor (N' - 10000) \text{ powr } (1/3) \rfloor$ 

```

The same mistake as above crops up, and we have corrected it in the same way.

```

  have ...  $\geq 31$ 
  by (smt (verit) <math>31 \leq \lfloor \text{real } 30000 \text{ powr } (1 / 3) \rfloor</math> <math>40000 < N'</math> floor-mono
floor-of-nat
    int-ops(2) int-ops(6) less-imp-of-nat-less numeral-Bit0 numeral-Bit1
numeral-One
    of-nat-numeral powr-mono2 zero-le-divide-1-iff)
  hence  $b \geq 31$ 
    using b-def by simp
  hence nat b = b
    by simp

  define d'::int where  $d' = (b+1)^3 - b^3$ 
  moreover have ...  $= 3*b^2 + 3*b + 1$ 
    by Groebner-Basis.algebra
  moreover have  $b^2 - 3*b - 1 > 0$ 

```


using $\langle b \geq 31 \rangle \langle \forall (x::int) > 4. x^2 - 3*x - 1 > 0 \rangle$ by *simp*
 moreover hence $4*b^2 > 3*b^2 + 3*b + 1$
 by *simp*
 ultimately have $4*b^2 > d'$
 by *presburger*

have $b < (N' \text{ powr } (1/3))$
 using *b-def*
 by (*smt (verit, best) N-min-a-cube-ge40000 N'-def diff-less floor-eq-iff of-nat-0-le-iff of-nat-less-iff powr-less-mono2 zero-less-divide-1-iff zero-less-numeral*)
 hence $b \text{ powr } 2 < (N' \text{ powr } (1/3)) \text{ powr } 2$
 using $\langle 31 \leq b \rangle$ *order-less-le* by *fastforce*
 hence $4*b^2 < 4*(N' \text{ powr } (2/3))$
 using *b-def powr-powr* by *fastforce*
 have $N' - (N' - 10000) \leq 10000$
 by *simp*
 hence $N' - ((N' - 10000) \text{ powr } (1/3)) \text{ powr } 3 \leq 10000$
 using *powr-powr powr-one-gt-zero-iff* $\langle N' > 40000 \rangle$ by *force*
 hence $N' - \lfloor (N' - 10000) \text{ powr } (1/3) + 1 \rfloor \text{ powr } 3 < 10000$
 by (*smt (verit, best) powr-ge-zero powr-less-mono2 real-of-int-floor-add-one-gt*)
 hence $N' - (b+1) \text{ powr } 3 < 10000$
 using *b-def one-add-floor* by *simp*
 have $(b+1) \text{ powr } 3 = (b+1)^3$
 using *b-def* by *auto*
 hence $N' - (b+1)^3 < 10000$
 using $\langle N' - (b+1) \text{ powr } 3 < 10000 \rangle$ by *linarith*
 have $\dots \leq N' - (N' - 10000)$
 using $\langle N' > 40000 \rangle$ by *auto*
 moreover have $\dots = N' - ((N' - 10000) \text{ powr } (1/3)) \text{ powr } 3$
 using *powr-powr* by *simp*
 moreover have $\dots \leq N' - \lfloor ((N' - 10000) \text{ powr } (1/3)) \rfloor \text{ powr } 3$
 by *simp*
 moreover have $\dots = N' - b^3$
 using *b-def* by *simp*
 ultimately have $10000 \leq N' - b^3$
 by *linarith*
 have $\dots = N' - (b+1)^3 + d'$
 using *d'-def* by *simp*
 moreover have $\dots < 10000 + 4*(N' \text{ powr } (2/3))$
 using $\langle N' - (b+1)^3 < 10000 \rangle \langle 4*b^2 > d' \rangle \langle 4*b^2 < 4*(N' \text{ powr } (2/3)) \rangle$ *le-less-trans*
 by *linarith*
 ultimately have $N' - b^3 < 10000 + 4*(N' \text{ powr } (2/3))$
 by *presburger*
 hence $\text{int } (N' - (\text{nat } b)^3) < 10000 + 4*(N' \text{ powr } (2/3))$
 using $\langle \text{nat } b = b \rangle$
 by (*smt (verit, best) 10000 ≤ int N' - b ^ 3 int-ops(6) of-int-of-nat-eq of-nat-power*)

```

consider (N'-min-b-cube-leq40000) N'-(nat b)^3 ≤ 40000 | (N'-min-b-cube-ge40000)
N' - (nat b)^3 > 40000
  by arith
thus ?thesis
proof (cases)
  case N'-min-b-cube-leq40000
  have N' - (nat b)^3 ≥ 10000
    using ⟨10000 ≤ N' - b^3⟩ int-nat-eq ⟨b ≥ 31⟩
    by (smt (verit) int-ops(6) numeral-Bit0 numeral-Bit1 numeral-One
of-nat-1 of-nat-le-iff
of-nat-numeral of-nat-power)
  hence N - (nat a)^3 - (nat b)^3 ≥ 10000
    using N'-def by simp
  hence is-sumpow 3 6 (N - (nat a)^3 - (nat b)^3)
    using leq-40000-is-sum-of-9-cubes N'-min-b-cube-leq40000 N'-def by
force
  then obtain l::nat list where length l = 6 and N - (nat a)^3 - (nat
b)^3 = sumpow 3 l
    using is-sumpow-def by blast
  have 0 + (nat b)^3 + (nat a)^3 + sumpow 3 l = sumpow 3 ((nat a)#(nat
b)#0#l)
    by (simp add: fold-plus-sum-list-rev)
  hence ... = N
    using ⟨N - (nat a)^3 - (nat b)^3 = sumpow 3 l⟩ ⟨10000 ≤ N - nat
a^3 - nat b^3⟩
    by presburger
  have length ((nat a)#(nat b)#0#l) = 9
    using ⟨length l = 6⟩ by simp
  thus ?thesis
    using ⟨sumpow 3 (nat a # nat b # 0 # l) = N⟩ is-sumpow-def by blast
next
  case N'-min-b-cube-ge40000

define N''::nat where N'' = N - (nat a)^3 - (nat b)^3
hence N'' > 40000
  using N'-min-b-cube-ge40000 N'-def by simp

define c::int where c = ⌊(N'' - 10000) powr (1/3)⌋

```

We correct the same mistake as above.

```

have ... ≥ 31
  by (smt (verit) ⟨31 ≤ ⌊real 30000 powr (1 / 3)⌋⟩ ⟨40000 < N''⟩
floor-mono floor-of-nat
int-ops(2) int-ops(6) less-imp-of-nat-less numeral-Bit0 numeral-Bit1
numeral-One
of-nat-numeral powr-mono2 zero-le-divide-1-iff)
hence c ≥ 31
  using c-def by simp

```

hence $c \text{ notneg}$: $\text{nat } c = c$
using int-nat-eq **by** simp

define $d''::\text{int}$ **where** $d'' = (c+1)^3 - c^3$
moreover have $\dots = 3*c^2 + 3*c + 1$
by $\text{Groebner-Basis.algebra}$
moreover have $c^2 - 3*c - 1 > 0$
using $\langle c \geq 31 \rangle \langle \forall (x::\text{int}) > 4. x^2 - 3*x - 1 > 0 \rangle$ **by** simp
moreover hence $4*c^2 > 3*c^2 + 3*c + 1$
by simp
ultimately have $4*c^2 > d''$
by presburger

have $c < (N'' \text{ powr } (1/3))$
using $c\text{-def } N'\text{-min-b-cube-ge40000 } N'\text{-def } N''\text{-def}$
by $(\text{smt } (\text{verit}, \text{cfv-SIG}) \text{ diff-less floor-eq-iff of-nat-0-le-iff of-nat-less-iff } \text{ powr-less-mono2 zero-less-divide-1-iff zero-less-numeral})$
hence $c \text{ powr } 2 < (N'' \text{ powr } (1/3)) \text{ powr } 2$
using $\langle 31 \leq c \rangle \text{ order-less-le}$ **by** fastforce
hence $4*c^2 < 4*(N'' \text{ powr } (2/3))$
using $c\text{-def powr-powr}$ **by** fastforce

have $N'' - (N'' - 10000) \leq 10000$
by simp
hence $N'' - ((N'' - 10000) \text{ powr } (1/3)) \text{ powr } 3 \leq 10000$
using $\text{powr-powr powr-one-gt-zero-iff } \langle N' > 40000 \rangle$ **by** force
hence $N'' - \lfloor (N'' - 10000) \text{ powr } (1/3) + 1 \rfloor \text{ powr } 3 < 10000$
by $(\text{smt } (\text{verit}, \text{best}) \text{ powr-ge-zero powr-less-mono2 real-of-int-floor-add-one-gt})$
hence $N'' - (c+1) \text{ powr } 3 < 10000$
using $c\text{-def one-add-floor}$ **by** simp
have $(c+1) \text{ powr } 3 = (c+1)^3$
using $c\text{-def}$ **by** auto
hence $N'' - (c+1)^3 < 10000$
using $\langle N'' - (c+1) \text{ powr } 3 < 10000 \rangle$ **by** linarith
have $\dots \leq N'' - (N'' - 10000)$
using $\langle N'' > 40000 \rangle$ **by** auto
moreover have $\dots = N'' - ((N'' - 10000) \text{ powr } (1/3)) \text{ powr } 3$
using powr-powr **by** simp
moreover have $\dots \leq N'' - \lfloor ((N'' - 10000) \text{ powr } (1/3)) \rfloor \text{ powr } 3$
by simp
moreover have $\dots = N'' - c^3$
using $c\text{-def}$ **by** simp
ultimately have $10000 \leq N'' - c^3$
by linarith
moreover have $\dots = N'' - (c+1)^3 + d''$
using $d''\text{-def}$ **by** simp
moreover have $\dots < 10000 + 4*(N'' \text{ powr } (2/3))$
using $\langle N'' - (c+1)^3 < 10000 \rangle \langle 4*c^2 > d'' \rangle \langle 4*c^2 < 4*(N'' \text{ powr } (2/3)) \rangle$ **le-less-trans**

by *linarith*
ultimately have $N'' - c^3 < 10000 + 4*(N'' \text{ powr } (2/3))$
by *presburger*
hence $\text{int } (N'' - (\text{nat } c)^3) < 10000 + 4*(N'' \text{ powr } (2/3))$
using *cnotneg*
by (*smt (verit, best) <10000 ≤ int N'' - c ^ 3> int-ops(6) of-int-of-nat-eq of-nat-power*)

have $N - (\text{nat } a)^3 - (\text{nat } b)^3 - (c+1)^3 < 10000$
using $\langle N'' - (c+1)^3 < 10000 \rangle$ *N''-def* **by** *simp*
have $\dots \leq N'' - (\text{nat } c)^3$
by (*smt (verit, del-insts) <10000 ≤ int N'' - c ^ 3> cnotneg int-ops(6) of-nat-power*)

have $\dots < 10000 + 4*((N - (\text{nat } a)^3 - (\text{nat } b)^3) \text{ powr } (2/3))$
using *N''-def* $\langle \text{int } (N'' - (\text{nat } c)^3) < 10000 + 4*(N'' \text{ powr } (2/3)) \rangle$
by *simp*

moreover have $\dots < 10000 + 4*((10000 + 4*((N - (\text{nat } a)^3) \text{ powr } (2/3))) \text{ powr } (2/3))$
using $\langle \text{int } (N' - (\text{nat } b)^3) < 10000 + 4*(N' \text{ powr } (2/3)) \rangle$
powr-less-mono2 N'-def **by** *simp*

moreover have $\dots < 10000 + 4*((10000 + 4*((10000 + 4*(N \text{ powr } (2/3))) \text{ powr } (2/3))) \text{ powr } (2/3))$
using $\langle \text{int } (N - (\text{nat } a)^3) < 10000 + 4*(N \text{ powr } (2/3)) \rangle$ *powr-less-mono2*
by *simp*

moreover have $\dots \leq 10000 + 4*((10000 + 4*((10000 + 4*((\text{int } 8^10) \text{ powr } (2/3))) \text{ powr } (2/3))) \text{ powr } (2/3))$
using *leq8pow10 powr-less-mono2 nle-le*
by (*smt (verit, best) numeral-power-le-of-nat-cancel-iff of-int-of-nat-eq of-nat-0-le-iff of-nat-power powr-ge-zero zero-less-divide-iff*)

moreover have $\dots = 10000 + 4*((10000 + 4*((4204304) \text{ powr } (2/3))) \text{ powr } (2/3))$
using *powr-powr* **by** *simp*

moreover have $\dots \leq 10000 + 4*((10000 + 4*((4251528) \text{ powr } (2/3))) \text{ powr } (2/3))$
by (*smt (verit, best) powr-ge-zero powr-less-mono2 zero-less-divide-iff*)

moreover have $\dots = 10000 + 4*((114976) \text{ powr } (2/3))$
by *auto*

moreover have $\dots \leq 10000 + 4*((117649) \text{ powr } (2/3))$
by (*smt (verit, best) powr-ge-zero powr-less-mono2 zero-less-divide-iff*)

moreover have $\dots \leq 20000$
by *auto*
ultimately have $N - (\text{nat } a)^3 - (\text{nat } b)^3 - (\text{nat } c)^3 \leq 20000$
by (*smt (verit, del-insts) N''-def numeral-Bit0 numeral-Bit1 numerals(1) of-int-of-nat-eq of-nat-1 of-nat-le-iff of-nat-numeral*)

```

hence is-sumpow 3 6 (N - (nat a)^3 - (nat b)^3 - (nat c)^3)
using leq-40000-is-sum-of-9-cubes <10000 ≤ N'' - c^3> N''-def cnotneg
by (smt (verit) <10000 ≤ int (N'' - nat c ^ 3)> int-ops(2) numeral-Bit0
numeral-Bit1
numeral-One of-nat-le-iff of-nat-numeral order-less-le)
then obtain l::nat list where length l = 6 and
N - (nat a)^3 - (nat b)^3 - (nat c)^3 = sumpow 3 l
using is-sumpow-def by blast
moreover have (nat c)^3 + (nat b)^3 + (nat a)^3 + sumpow 3 l =
sumpow 3 ((nat a)#(nat b)#(nat c)#l)
by (simp add: fold-plus-sum-list-rev)
ultimately have ... = N
using <10000 ≤ int N'' - c^3> N''-def cnotneg
by (smt (verit, del-insts) diff-diff-left of-nat-power of-nat-le-iff le-add-diff-inverse

add commute int-ops(6))
moreover have length ((nat a)#(nat b)#(nat c)#l) = 9
using <length l = 6> by simp
ultimately show ?thesis
using is-sumpow-def by blast
qed
qed
next
case leq40000
thus ?thesis
using leq-40000-is-sum-of-9-cubes by blast
qed
qed
thus ?thesis
by simp
qed
end

```

References

- [1] A. Danilkin and L. Chevalier. Three squares theorem. *Archive of Formal Proofs*, May 2023.
https://isa-afp.org/entries/Three_Squares.html, Formal proof development.
- [2] M. B. Nathanson. *Additive Number Theory: The Classical Bases*, volume 164 of *Graduate Texts in Mathematics*. Springer, New York, 1996.