```
#|

 Copyright (C) 1994 by Computational Logic, Inc.  All Rights Reserved.

 This script is hereby placed in the public domain, and therefore unlimited
 editing and redistribution is permitted.

 NO WARRANTY

 Computational Logic, Inc. PROVIDES ABSOLUTELY NO WARRANTY.  THE EVENT SCRIPT
 IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED,
 INCLUDING, BUT NOT LIMITED TO, ANY IMPLIED WARRANTIES OF MERCHANTABILITY AND
 FITNESS FOR A PARTICULAR PURPOSE.  THE ENTIRE RISK AS TO THE QUALITY AND
 PERFORMANCE OF THE SCRIPT IS WITH YOU.  SHOULD THE SCRIPT PROVE DEFECTIVE, YOU
 ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

 IN NO EVENT WILL Computational Logic, Inc. BE LIABLE TO YOU FOR ANY DAMAGES,
 ANY LOST PROFITS, LOST MONIES, OR OTHER SPECIAL, INCIDENTAL OR CONSEQUENTIAL
 DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THIS SCRIPT (INCLUDING BUT
 NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES
 SUSTAINED BY THIRD PARTIES), EVEN IF YOU HAVE ADVISED US OF THE POSSIBILITY OF
 SUCH DAMAGES, OR FOR ANY CLAIM BY ANY OTHER PARTY.
|#

; Matt Wilding
```

EVENT: Start with the library `"extras"` using the compiled version.

```
;; Proof of Matijasevich's lemma constructed by MW week of 5-10-91.  Inspired by
;; proof in Concrete Mathematics by Graham, Knuth, and Patashnik and notes from
;; Dijkstra's Capita Selecta Spring '90 course.

;;;; Some additional facts needed in the proof that appear useful
;;;; enough to add to libraries (and a few facts to prove them)
```

THEOREM: irem-igcd-arg1
$(\mathrm{irem}\,(\mathrm{igcd}\,(a,\,b),\,c) = 0) = ((\mathrm{irem}\,(a,\,c) = 0) \wedge (\mathrm{irem}\,(b,\,c) = 0))$

THEOREM: irem-x-x
$\mathrm{irem}\,(x,\,x) = 0$

```
;; The nonsensical induction hint is a silly trick to get the prover
;; to throw an equality away after using it
```

THEOREM: gcd-times-easy-proof-helper-helper
$((a \in \mathbf{N}) \wedge (b \in \mathbf{N}) \wedge (c \in \mathbf{N}))$
$\rightarrow \quad (((c * \gcd(a,\ b))\ \mathbf{mod}\ \gcd(a,\ b * c)) = 0)$

THEOREM: gcd-times-easy-proof-helper
$((c * \gcd(a,\ b))\ \mathbf{mod}\ \gcd(a,\ b * c)) = 0$

THEOREM: gcd-times-easy-proof
$(\gcd(a,\ b) = 1) \rightarrow (\gcd(a,\ b * c) = \gcd(a,\ c))$

```
;; Would the more general case of rewriting gcd-times regardless
;; of the value of gcd with one of the times arguments be superior?
```

THEOREM: gcd-times-easy
$(\gcd(a,\ b) = 1)$
$\rightarrow \quad ((\gcd(a,\ b * c) = \gcd(a,\ c)) \wedge (\gcd(a,\ c * b) = \gcd(a,\ c)))$

THEOREM: lessp-gcd3
$((\gcd(a,\ b) < a) = (((b\ \mathbf{mod}\ a) \neq 0) \wedge (a \not\simeq 0) \wedge (b \not\simeq 0)))$
$\wedge \quad ((\gcd(b,\ a) < a) = (((b\ \mathbf{mod}\ a) \neq 0) \wedge (a \not\simeq 0) \wedge (b \not\simeq 0)))$

THEOREM: equal-remainder-gcd-0-proof
$((\gcd(a,\ b)\ \mathbf{mod}\ a) = 0) = ((b\ \mathbf{mod}\ a) = 0)$

```
;; not used in proof (though equal-remainder-gcd-0-proof is)
```

THEOREM: equal-remainder-gcd-0
$(((\gcd(a,\ b)\ \mathbf{mod}\ a) = 0) = ((b\ \mathbf{mod}\ a) = 0))$
$\wedge \quad (((\gcd(b,\ a)\ \mathbf{mod}\ a) = 0) = ((b\ \mathbf{mod}\ a) = 0))$

THEOREM: equal-remainder-x-y-x
$((x\ \mathbf{mod}\ y) = x)$
$= \quad (((x \in \mathbf{N}) \wedge (x < y))$
$\quad \vee \quad (x = 0)$
$\quad \vee \quad ((\mathrm{fix}(y) = 0) \wedge (x \in \mathbf{N})))$

THEOREM: gcd-a-gcd-exp-a
$\gcd(a, \exp(a,\ b))$
$= \quad \mathbf{if}\ b \simeq 0\ \mathbf{then}\ 1$
$\quad \mathbf{else}\ \mathrm{fix}(a)\ \mathbf{endif}$

THEOREM: equal-gcd-times-1-help1
$((\gcd(a,\ b) = 1) \wedge (\gcd(a,\ c) = 1)) \rightarrow (\gcd(a,\ b * c) = 1)$

THEOREM: remainder-gcd-gcd-0-hack
$(\gcd(a,\ b * c)\ \mathbf{mod}\ \gcd(a,\ c)) = 0$

THEOREM: lessp-gcd-times
$((a \not\simeq 0) \vee ((b * c) \not\simeq 0)) \rightarrow (\gcd(a, b * c) \not< \gcd(a, c))$

THEOREM: lessp-1-hack
$(1 < x) = ((x \not\simeq 0) \wedge (x \neq 1))$

THEOREM: equal-gcd-times-1-help2
$(\gcd(a, b * c) = 1) \rightarrow (\gcd(a, c) = 1)$

THEOREM: equal-gcd-times-1
$(\gcd(a, b * c) = 1) = ((\gcd(a, b) = 1) \wedge (\gcd(a, c) = 1))$

THEOREM: gcd-exp-1
$((\gcd(a, \exp(b, c)) = 1) = ((\gcd(a, b) = 1) \vee (c \simeq 0)))$
$\wedge \quad ((\gcd(\exp(b, c), a) = 1) = ((\gcd(a, b) = 1) \vee (c \simeq 0)))$

```
;; mistake in other version of this lemma limits applicability and
;; makes this version needed
```

THEOREM: gcd-remainder-times-fact1-proof2
$(\gcd(a, b) = 1) \rightarrow ((((c * b) \textbf{ mod } a) = 0) = ((c \textbf{ mod } a) = 0))$

THEOREM: times-quotient-better
$((x \textbf{ mod } y) = 0)$
$\rightarrow \quad ((((x \div y) * y)$
$\quad = \quad \textbf{if } y \simeq 0 \textbf{ then } 0$
$\quad\quad \textbf{else } \mathrm{fix}(x) \textbf{ endif})$
$\quad \wedge \quad ((y * (x \div y))$
$\quad\quad = \quad \textbf{if } y \simeq 0 \textbf{ then } 0$
$\quad\quad\quad \textbf{else } \mathrm{fix}(x) \textbf{ endif}))$

THEOREM: equal-remainder-exp-0
$(((a \textbf{ mod } c) = 0) \wedge (b \not\simeq 0)) \rightarrow ((\exp(a, b) \textbf{ mod } c) = 0)$

THEOREM: remainder-2-hack
$(2 \textbf{ mod } x)$
$= \quad \textbf{if } (x = 1) \vee (x = 2) \textbf{ then } 0$
$\quad \textbf{else } 2 \textbf{ endif}$

THEOREM: remainder-times-hack2
$((a \textbf{ mod } b) \neq 0)$
$\rightarrow \quad (((a \textbf{ mod } (b * c)) \neq 0) \wedge ((a \textbf{ mod } (c * b)) \neq 0))$

THEOREM: gcd-a-add1-a
$\gcd(a, 1 + a) = 1$

```
;; These facts should perhaps be generalized into a meta lemma that
;; factors first
```

THEOREM: remainder-times-times-hack
$((b * (a * c)) \textbf{ mod } (a * x)) = (a * ((b * c) \textbf{ mod } x))$

THEOREM: remainder-plus-times-hack
$(((a * x) + (b * (c * (a * y)))) \textbf{ mod } (a * z))$
$= \quad (a * ((x + (b * (c * y))) \textbf{ mod } z))$

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;; Now, on to the proof
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
```

DEFINITION:
$\text{fib}\,(x)$
$= \quad \textbf{if } x \simeq 0 \textbf{ then } 0$
$\quad \textbf{elseif } x = 1 \textbf{ then } 1$
$\quad \textbf{else } \text{fib}\,((x - 1) - 1) + \text{fib}\,(x - 1) \textbf{ endif}$

THEOREM: fib-plus
$\text{fib}\,(j + k)$
$= \quad \textbf{if } j \simeq 0 \textbf{ then } \text{fib}\,(k)$
$\quad \textbf{else } (\text{fib}\,(1 + k) * \text{fib}\,(j)) + (\text{fib}\,(j - 1) * \text{fib}\,(k)) \textbf{ endif}$

THEOREM: gcd-fib-next-fib
$\text{gcd}\,(\text{fib}\,(n), \text{fib}\,(1 + n)) = 1$

THEOREM: gcd-fib
$\text{gcd}\,(\text{fib}\,(a), \text{fib}\,(b)) = \text{fib}\,(\text{gcd}\,(a, b))$

THEOREM: equal-fib-constant
$((\text{fib}\,(a) = 0) = (a \simeq 0))$
$\wedge \quad ((\text{fib}\,(a) = 1) = ((a = 1) \vee (a = 2)))$

THEOREM: fib-small
$(a < 3)$
$\rightarrow \quad (\text{fib}\,(a)$
$\quad = \quad \textbf{if } a \simeq 0 \textbf{ then } 0$
$\quad \quad \textbf{else } 1 \textbf{ endif})$

DEFINITION:
$\text{double-fib-induction}\,(a, b)$
$= \quad \textbf{if } (a < 3) \vee (b < 3) \textbf{ then t}$
$\quad \textbf{else } \text{double-fib-induction}\,(a - 1, b - 1)$
$\quad \quad \wedge \quad \text{double-fib-induction}\,((a - 1) - 1, (b - 1) - 1) \textbf{ endif}$

THEOREM: lessp-fib-fib
$(\text{fib}\,(a) < \text{fib}\,(b)) = ((a < b) \wedge (\neg\,((a = 1) \wedge (b = 2))))$

THEOREM: remainder-fib-fib-0
$((\text{fib}\,(a)\ \textbf{mod}\ \text{fib}\,(b)) = 0) = (((a\ \textbf{mod}\ b) = 0) \vee (b = 2))$

```
;; fib-times-open and fib-add1-times-open apply fib-plus to
;; particular terms in a way that will be useful in the proof
;; of fib-times-special
```

THEOREM: fib-times-open
$(0 < k)$
$\rightarrow\quad (\text{fib}\,(k * n)$
$\quad = \quad ((\text{fib}\,(n) * \text{fib}\,(1 + ((k - 1) * n)))$
$\quad\quad + \quad (\text{fib}\,(n - 1) * \text{fib}\,((k - 1) * n))))$

EVENT: Disable fib-times-open.

THEOREM: fib-add1-times-open
$(0 < k)$
$\rightarrow\quad (\text{fib}\,(1 + (k * n))$
$\quad = \quad ((\text{fib}\,(1 + ((k - 1) * n)) * \text{fib}\,(1 + n))$
$\quad\quad + \quad (\text{fib}\,((k - 1) * n) * \text{fib}\,(n))))$

EVENT: Disable fib-add1-times-open.

THEOREM: fib-times-special-step
$(1 < k)$
$\rightarrow\quad (((k * (\text{fib}\,(n) * \exp(\text{fib}\,(1 + n),\ k - 1)))\ \textbf{mod}\ (\text{fib}\,(n) * \text{fib}\,(n)))$
$\quad = \quad (((\text{fib}\,(n) * \exp(\text{fib}\,(1 + n),\ k - 1))$
$\quad\quad + \quad (\text{fib}\,(n - 1)$
$\quad\quad\quad * \quad ((k - 1)$
$\quad\quad\quad\quad * \quad (\text{fib}\,(n)$
$\quad\quad\quad\quad\quad * \quad \exp(\text{fib}\,(1 + n),\ (k - 1) - 1)))))$
$\quad\quad \textbf{mod} \quad (\text{fib}\,(n) * \text{fib}\,(n))))$

THEOREM: fib-remainder-hack
$(\text{fib}\,(x * y)\ \textbf{mod}\ \text{fib}\,(x)) = 0$

```
;; We wish the rewriter to use a modulo arithmetic equality to rewrite
;; a term.  Fortunately, we can can construct REMAINDER-FIB-BACKCHAIN
;; to do it the way we wish since the terms are of a specialized form
```

THEOREM: remainder-backchain-proof1
$((a\ \textbf{mod}\ c) = (b\ \textbf{mod}\ c)) \rightarrow (((a * d)\ \textbf{mod}\ c) = ((b * d)\ \textbf{mod}\ c))$

THEOREM: remainder-backchain-proof2
$(((a \textbf{ mod } c) = (b \textbf{ mod } c)) \wedge ((d \textbf{ mod } c) = (e \textbf{ mod } c)))$
$\rightarrow \quad (((a + d) \textbf{ mod } c) = ((b + e) \textbf{ mod } c))$

THEOREM: remainder-fib-backchain
$(((\text{fib}\,(x) \textbf{ mod } c) = (\exp(p,\, q) \textbf{ mod } c))$
$\wedge \quad ((\text{fib}\,(y) \textbf{ mod } c) = ((r * s) \textbf{ mod } c)))$
$\rightarrow \quad ((((a * \text{fib}\,(x)) + (b * \text{fib}\,(y))) \textbf{ mod } c)$
$= \quad (((a * \exp(p,\, q)) + (b * (r * s))) \textbf{ mod } c))$

```
;; This is the guts of the proof
```

THEOREM: fib-times-special
$(1 < k)$
$\rightarrow \quad (((\text{fib}\,(k * n) \textbf{ mod } \exp(\text{fib}\,(n),\, 2))$
$= \quad ((k * (\text{fib}\,(n) * \exp(\text{fib}\,(1 + n),\, k - 1)))$
$\textbf{mod} \quad \exp(\text{fib}\,(n),\, 2)))$
$\wedge \quad ((\text{fib}\,(1 + (k * n)) \textbf{ mod } \exp(\text{fib}\,(n),\, 2))$
$= \quad (\exp(\text{fib}\,(1 + n),\, k) \textbf{ mod } \exp(\text{fib}\,(n),\, 2))))$

THEOREM: fib-remainder-times-special
$(\text{fib}\,(n * k) \textbf{ mod } (\text{fib}\,(n) * \text{fib}\,(n)))$
$= \quad ((k * (\text{fib}\,(n) * \exp(\text{fib}\,(1 + n),\, k - 1))) \textbf{ mod } (\text{fib}\,(n) * \text{fib}\,(n)))$

THEOREM: remainder-times-times-hack2
$((k * n) \textbf{ mod } (n * p)) = (n * (k \textbf{ mod } p))$

THEOREM: matijasevich-lemma-helper
$((2 < n) \wedge ((m \textbf{ mod } n) = 0))$
$\rightarrow \quad (((\text{fib}\,(m) \textbf{ mod } (\text{fib}\,(n) * \text{fib}\,(n))) = 0)$
$= \quad ((m \textbf{ mod } (n * \text{fib}\,(n))) = 0))$

THEOREM: matijasevich-lemma
$(2 < n)$
$\rightarrow \quad (((\text{fib}\,(m) \textbf{ mod } (\text{fib}\,(n) * \text{fib}\,(n))) = 0)$
$= \quad ((m \textbf{ mod } (n * \text{fib}\,(n))) = 0))$

Index