```
#|

 Copyright (C) 1994 by Robert S. Boyer and J Strother Moore.  All Rights
 Reserved.

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

 NO WARRANTY

 Robert S. Boyer and J Strother Moore PROVIDE 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 Robert S. Boyer or J Strother Moore 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.

|#
```

EVENT: Start with the library "proveall" using the compiled version.

EVENT: For efficiency, compile those definitions not yet compiled.

EVENT: Disable euclid.

EVENT: Disable quotient-divides.

EVENT: Disable if-times-then-divides.

EVENT: Disable times.

DEFINITION:

crypt $(m,\ e,\ n)$
$=$ **if** $e \simeq 0$ **then** 1
elseif even (e) **then** square (crypt $(m,\ e \div 2,\ n)$) **mod** n
else $(m * (\text{square}(\text{crypt}(m,\ e \div 2,\ n))$ **mod** $n))$ **mod** n **endif**

THEOREM: times-mod-1
$((x * (y \textbf{ mod } n)) \textbf{ mod } n) = ((x * y) \textbf{ mod } n)$

THEOREM: times-mod-2
$((a * (b * (y \textbf{ mod } n))) \textbf{ mod } n) = ((a * b * y) \textbf{ mod } n)$

THEOREM: crypt-correct
$(n \neq 1) \rightarrow (\text{crypt}(m,\ e,\ n) = (\exp(m,\ e) \textbf{ mod } n))$

THEOREM: times-mod-3
$(((a \textbf{ mod } n) * b) \textbf{ mod } n) = ((a * b) \textbf{ mod } n)$

THEOREM: remainder-exp-lemma
$((y \textbf{ mod } a) = (z \textbf{ mod } a))$
$\rightarrow \quad ((((x * y) \textbf{ mod } a) = ((x * z) \textbf{ mod } a)) = \textbf{t})$

THEOREM: remainder-exp
$(\exp(a \textbf{ mod } n,\ i) \textbf{ mod } n) = (\exp(a,\ i) \textbf{ mod } n)$

THEOREM: exp-mod-is-1
$((\exp(m,\ j) \textbf{ mod } p) = 1) \rightarrow ((\exp(m,\ i * j) \textbf{ mod } p) = 1)$

DEFINITION:
pdifference $(a,\ b)$
$=$ **if** $a < b$ **then** $b - a$
else $a - b$ **endif**

THEOREM: times-distributes-over-pdifference
$(m * \text{pdifference}(a,\ b)) = \text{pdifference}(m * a,\ m * b)$

THEOREM: equal-mods-trick-1
$((\text{pdifference}(a,\ b) \textbf{ mod } p) = 0) \rightarrow (((a \textbf{ mod } p) = (b \textbf{ mod } p)) = \textbf{t})$

THEOREM: equal-mods-trick-2
$((a \textbf{ mod } p) = (b \textbf{ mod } p)) \rightarrow ((\text{pdifference}(a,\ b) \textbf{ mod } p) = 0)$

EVENT: Disable pdifference.

THEOREM: prime-key-trick
$((((m * a) \textbf{ mod } p) = ((m * b) \textbf{ mod } p))$
$\land \quad ((m \textbf{ mod } p) \neq 0)$
$\land \quad \text{prime}(p))$
$\rightarrow \quad (((a \textbf{ mod } p) = (b \textbf{ mod } p)) = \textbf{t})$

THEOREM: product-divides-lemma
$((x \textbf{ mod } z) = 0) \rightarrow (((y * x) \textbf{ mod } (y * z)) = 0)$

THEOREM: product-divides
$$
\begin{array}{ll}
(((x \textbf{ mod } p) = 0) & \\
\land & ((x \textbf{ mod } q) = 0) \\
\land & \text{prime}\,(p) \\
\land & \text{prime}\,(q) \\
\land & (p \neq q)) \\
\rightarrow & ((x \textbf{ mod } (p * q)) = 0)
\end{array}
$$

THEOREM: thm-53-specialized-to-primes
$$
\begin{array}{ll}
(\text{prime}\,(p) & \\
\land & \text{prime}\,(q) \\
\land & (p \neq q) \\
\land & ((a \textbf{ mod } p) = (b \textbf{ mod } p)) \\
\land & ((a \textbf{ mod } q) = (b \textbf{ mod } q))) \\
\rightarrow & ((a \textbf{ mod } (p * q)) = (b \textbf{ mod } (p * q)))
\end{array}
$$

THEOREM: corollary-53
$$
\begin{array}{ll}
(\text{prime}\,(p) & \\
\land & \text{prime}\,(q) \\
\land & (p \neq q) \\
\land & ((a \textbf{ mod } p) = (b \textbf{ mod } p)) \\
\land & ((a \textbf{ mod } q) = (b \textbf{ mod } q)) \\
\land & (b \in \mathbf{N}) \\
\land & (b < (p * q))) \\
\rightarrow & (((a \textbf{ mod } (p * q)) = b) = \mathbf{t})
\end{array}
$$

THEOREM: thm-55-specialized-to-primes
$$
\begin{array}{ll}
(\text{prime}\,(p) \land ((m \textbf{ mod } p) \neq 0)) & \\
\rightarrow & ((((m * x) \textbf{ mod } p) = ((m * y) \textbf{ mod } p)) \\
 & \quad = \quad ((x \textbf{ mod } p) = (y \textbf{ mod } p)))
\end{array}
$$

THEOREM: corollary-55
$$
\begin{array}{ll}
\text{prime}\,(p) & \\
\rightarrow & ((((m * x) \textbf{ mod } p) = (m \textbf{ mod } p)) \\
 & \quad = \quad (((m \textbf{ mod } p) = 0) \lor ((x \textbf{ mod } p) = 1)))
\end{array}
$$

DEFINITION:
all-distinct (l)
$$
\begin{array}{ll}
= & \textbf{if } l \simeq \textbf{nil} \textbf{ then } \mathbf{t} \\
 & \textbf{else } (\text{car}\,(l) \notin \text{cdr}\,(l)) \land \text{all-distinct}\,(\text{cdr}\,(l)) \textbf{ endif}
\end{array}
$$

DEFINITION:

all-lesseqp $(l,\ n)$
$=$ **if** $l \simeq$ **nil then t**
else $(n \not< \text{car}(l)) \land \text{all-lesseqp}(\text{cdr}(l),\ n)$ **endif**

DEFINITION:
all-non-zerop (l)
$=$ **if** $l \simeq$ **nil then t**
else $(\text{car}(l) \not\simeq 0) \land \text{all-non-zerop}(\text{cdr}(l))$ **endif**

DEFINITION:
positives (n)
$=$ **if** $n \simeq 0$ **then nil**
else $\text{cons}(n,\ \text{positives}(n - 1))$ **endif**

THEOREM: listp-positives
$\text{listp}(\text{positives}(n)) = (n \not\simeq 0)$

THEOREM: car-positives
$\text{car}(\text{positives}(n))$
$=$ **if** $n \simeq 0$ **then** 0
else n **endif**

THEOREM: member-positives
$(x \in \text{positives}(n))$
$=$ **if** $x \simeq 0$ **then f**
else $x < (1 + n)$ **endif**

THEOREM: all-non-zerop-delete
$\text{all-non-zerop}(l) \rightarrow \text{all-non-zerop}(\text{delete}(x,\ l))$

THEOREM: all-distinct-delete
$\text{all-distinct}(l) \rightarrow \text{all-distinct}(\text{delete}(x,\ l))$

THEOREM: pigeon-hole-principle-lemma-1
$(\text{all-distinct}(l) \land \text{all-lesseqp}(l,\ 1 + n))$
$\rightarrow$ $\text{all-lesseqp}(\text{delete}(1 + n,\ l),\ n)$

THEOREM: pigeon-hole-principle-lemma-2
$(((1 + n) \notin x) \land \text{all-lesseqp}(x,\ 1 + n)) \rightarrow \text{all-lesseqp}(x,\ n)$

THEOREM: perm-member
$(\text{perm}(a,\ b) \land (x \in a)) \rightarrow (x \in b)$

DEFINITION:
pigeon-hole-induction (l)
$=$ **if** $\text{listp}(l)$

then if length $(l) \in l$
then pigeon-hole-induction (delete (length (l), l))
else pigeon-hole-induction (cdr (l)) **endif**
else t endif

THEOREM: pigeon-hole-principle
(all-non-zerop $(l) \wedge$ all-distinct $(l) \wedge$ all-lesseqp (l, length (l)))
$\rightarrow$ perm (positives (length (l)), l)

THEOREM: perm-times-list
perm ($l1$, $l2$) $\rightarrow$ (times-list ($l1$) = times-list ($l2$))

THEOREM: times-list-positives
times-list (positives (n)) = fact (n)

THEOREM: times-list-equal-fact
perm (positives (n), l) $\rightarrow$ (times-list (l) = fact (n))

THEOREM: prime-fact
(prime $(p) \wedge (n < p)$) $\rightarrow$ ((fact (n) **mod** p) $\neq 0$)

DEFINITION:
s (n, m, p)
= **if** $n \simeq 0$ **then nil**
else cons (($m * n$) **mod** p, s $(n - 1, m, p)$) **endif**

THEOREM: remainder-times-list-s
(times-list (s (n, m, p)) **mod** p) = ((fact $(n) *$ exp (m, n)) **mod** p)

THEOREM: all-distinct-s-lemma
(prime $(p) \wedge$ ((m **mod** p) $\neq 0$) $\wedge$ ($n1 \in \mathbf{N}$) $\wedge$ ($n2 < n1$) $\wedge$ ($n1 < p$))
$\rightarrow$ ((($m * n1$) **mod** p) $\notin$ s ($n2$, m, p))

THEOREM: all-distinct-s
(prime $(p) \wedge$ ((m **mod** p) $\neq 0$) $\wedge (n < p)$) $\rightarrow$ all-distinct (s (n, m, p))

THEOREM: all-non-zerop-s
(prime $(p) \wedge$ ((m **mod** p) $\neq 0$) $\wedge (n < p)$) $\rightarrow$ all-non-zerop (s (n, m, p))

THEOREM: all-lesseqp-s
$(p \not\simeq 0) \rightarrow$ all-lesseqp (s (n, m, p), $p - 1$)

THEOREM: length-s
length (s (n, m, p)) = fix (n)

THEOREM: fermat-thm
(prime $(p) \wedge$ ((m **mod** p) $\neq 0$)) $\rightarrow$ ((exp $(m, p - 1)$ **mod** p) = 1)

THEOREM: crypt-inverts-step-1
$\text{prime}(p) \rightarrow (((m * \exp(m,\ k * (p - 1))) \mathbf{mod}\ p) = (m\ \mathbf{mod}\ p))$

THEOREM: crypt-inverts-step-1a
$\text{prime}(p)$
$\rightarrow \quad (((m * \exp(m,\ k * (p - 1) * (q - 1))) \mathbf{mod}\ p) = (m\ \mathbf{mod}\ p))$

THEOREM: crypt-inverts-step-1b
$\text{prime}(q)$
$\rightarrow \quad (((m * \exp(m,\ k * (p - 1) * (q - 1))) \mathbf{mod}\ q) = (m\ \mathbf{mod}\ q))$

THEOREM: crypt-inverts-step-2
$(\text{prime}(p)$
$\wedge \quad \text{prime}(q)$
$\wedge \quad (p \neq q)$
$\wedge \quad (m \in \mathbf{N})$
$\wedge \quad (m < (p * q))$
$\wedge \quad ((ed\ \mathbf{mod}\ ((p - 1) * (q - 1))) = 1))$
$\rightarrow \quad ((\exp(m,\ ed)\ \mathbf{mod}\ (p * q)) = m)$

THEOREM: crypt-inverts
$(\text{prime}(p)$
$\wedge \quad \text{prime}(q)$
$\wedge \quad (p \neq q)$
$\wedge \quad (n = (p * q))$
$\wedge \quad (m \in \mathbf{N})$
$\wedge \quad (m < n)$
$\wedge \quad (((e * d)\ \mathbf{mod}\ ((p - 1) * (q - 1))) = 1))$
$\rightarrow \quad (\text{crypt}(\text{crypt}(m,\ e,\ n),\ d,\ n) = m)$

EVENT: Make the library `"rsa"` and compile it.

Index

all-distinct, 3–5
all-distinct-delete, 4
all-distinct-s, 5
all-distinct-s-lemma, 5
all-lesseqp, 3–5
all-lesseqp-s, 5
all-non-zerop, 4, 5
all-non-zerop-delete, 4
all-non-zerop-s, 5

car-positives, 4
corollary-53, 3
corollary-55, 3
crypt, 1, 2, 6
crypt-correct, 2
crypt-inverts, 6
crypt-inverts-step-1, 6
crypt-inverts-step-1a, 6
crypt-inverts-step-1b, 6
crypt-inverts-step-2, 6

delete, 4, 5

equal-mods-trick-1, 2
equal-mods-trick-2, 2
even, 2
exp, 2, 5, 6
exp-mod-is-1, 2

fact, 5
fermat-thm, 5

length, 5
length-s, 5
listp-positives, 4

member-positives, 4

pdifference, 2
perm, 4, 5
perm-member, 4
perm-times-list, 5
pigeon-hole-induction, 4, 5
pigeon-hole-principle, 5
pigeon-hole-principle-lemma-1, 4
pigeon-hole-principle-lemma-2, 4
positives, 4, 5
prime, 2, 3, 5, 6
prime-fact, 5
prime-key-trick, 2
product-divides, 3
product-divides-lemma, 3

remainder-exp, 2
remainder-exp-lemma, 2
remainder-times-list-s, 5

s, 5
square, 2

thm-53-specialized-to-primes, 3
thm-55-specialized-to-primes, 3
times-distributes-over-pdifference, 2
times-list, 5
times-list-equal-fact, 5
times-list-positives, 5
times-mod-1, 2
times-mod-2, 2
times-mod-3, 2