

#|

Copyright (C) 1994 by Yuan Yu. All Rights Reserved.

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

NO WARRANTY

Yuan Yu 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 Yuan Yu 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 "mc20-2" using the compiled version.

```
; some general theorems. They are used to establish properties of string  
; functions.
```

THEOREM: illessp-lessp
 $((x \in \mathbb{N}) \wedge (y \in \mathbb{N})) \rightarrow (\text{illessp}(x, y) = (x < y))$

THEOREM: idifference-numberp
 $(\text{idifference}(x, y) \in \mathbb{N}) = (\neg \text{illessp}(x, y))$

THEOREM: idifference-equal-0
 $(\text{idifference}(x, y) = 0) = (\text{fix-int}(x) = \text{fix-int}(y))$

EVENT: Enable get-nth-0.

EVENT: Enable put-nth-0.

```

; the upper bound of strlen.

THEOREM: strlen-ub
stringp(i, n, lst) → (strlen(i, n, lst) < n)

; the lower bound of strlen.

THEOREM: strlen-lb
strlen(i, n, lst) ⪻ i

THEOREM: strlen-01
(strlen(i, 0, lst) = i)
∧ (strlen(i, 1, lst)
= if (i < 1) ∧ (get-nth(0, lst) ≠ 0) then 1
else i endif)
∧ (strlen(1, n, lst) ≠ 0)

THEOREM: strlen-ii
strlen(i, i, lst) = i

THEOREM: strlen-ii1
strlen(i, 1 + i, lst)
= if get-nth(i, lst) = 0 then i
else 1 + i endif

; some properties of strlen.
; lst[j] =\= 0, if i ≤ j < strlen(lst).

THEOREM: strlen-non0p
((i ≤ j) ∧ (j < strlen(i, n, lst))) → (get-nth(j, lst) ≠ 0)

; lst[strlen] == 0!

THEOREM: strlen-0p
stringp(i, n, lst) → (get-nth(strlen(i, n, lst), lst) = 0)

; some properties of strcpy.

THEOREM: strcpy-get-1
(j < i) → (get-nth(j, strcpy(i, lst1, n2, lst2)) = get-nth(j, lst1))

; strlen(strcpy(lst1, lst2)) == strlen(lst2).

THEOREM: strcpy-strlen
strlen(i, n2, strcpy(i, lst1, n2, lst2)) = strlen(i, n2, lst2)

; if i ≤ j ≤ strlen(lst2), lst1[j] == lst2[j].

```

```

THEOREM: strcpy-cpy
(stringp (i, n2, lst2) ∧ (i ≤ j) ∧ (j ≤ strlen (i, n2, lst2)))
→ (get-nth (j, strcpy (i, lst1, n2, lst2)) = get-nth (j, lst2))

; if lst2 is a string, then lst1' is also a string.

THEOREM: strcpy-stringp
stringp (i, n2, lst2) → stringp (i, n2, strcpy (i, lst1, n2, lst2))

; some properties of strcmp.
; (strcmp lst1 lst) == 0.

THEOREM: strcmp-id
strcmp (i, n, lst, lst) = 0

; (strcmp lst1 lst2) < 0 ==> (strcmp lst2 lst1) >= 0.

THEOREM: strcmp-antisym
negativep (strcmp (i, n, lst1, lst2)) = (0 < strcmp (i, n, lst2, lst1))

; the transitivity of strcmp.

THEOREM: strcmp-trans-1
(lst-of-chrp (lst1)
 ∧ lst-of-chrp (lst2)
 ∧ lst-of-chrp (lst3)
 ∧ (strcmp (i, n, lst1, lst2) ∈ N)
 ∧ (strcmp (i, n, lst2, lst3) ∈ N))
→ (strcmp (i, n, lst1, lst3) ∈ N)

; (strcpy lst1 lst2) == lst2.

THEOREM: strcmp-strcpy
strcmp (i, n, lst2, strcpy (i, lst1, n, lst2)) = 0

; if (strcmp lst1 lst2) == 0, all i <= j < (strlen lst1) lst1[j] == lst2[j]. 

THEOREM: strcmp-thm1
(lst-of-chrp (lst1)
 ∧ lst-of-chrp (lst2)
 ∧ (strcmp (i, n, lst1, lst2) = 0)
 ∧ (i ≤ j)
 ∧ (j < strlen (i, n, lst1)))
→ (get-nth (j, lst1) = get-nth (j, lst2))

```

DEFINITION:

$$\begin{aligned} \text{strcmp-sk}(n1, lst1, lst2) \\ \leftrightarrow \exists j (\forall i ((i < j) \rightarrow (\text{get-nth}(i, lst1) = \text{get-nth}(i, lst2))) \\ \wedge (\text{strcmp}(0, n1, lst1, lst2) \\ = \text{idifference}(\text{get-nth}(j, lst1), \text{get-nth}(j, lst2)))) \end{aligned}$$

DEFINITION:

$$\begin{aligned} \text{strcmp-j}(i, n1, lst1, lst2) \\ = \text{if } i < n1 \\ \quad \text{then if } \text{get-nth}(i, lst1) = \text{get-nth}(i, lst2) \\ \quad \quad \text{then if } \text{get-nth}(i, lst1) = \text{NULL} \text{ then fix}(i) \\ \quad \quad \quad \text{else strcmp-j}(1 + i, n1, lst1, lst2) \text{ endif} \\ \quad \quad \text{else fix}(i) \text{ endif} \\ \text{else fix}(i) \text{ endif} \end{aligned}$$

THEOREM: strcmp-j-1

$$\begin{aligned} ((i < \text{strcmp-j}(i1, n1, lst1, lst2)) \wedge (i1 \leq i)) \\ \rightarrow (\text{get-nth}(i, lst1) = \text{get-nth}(i, lst2)) \end{aligned}$$

THEOREM: strcmp-j-2

$$\begin{aligned} (\text{strcmp}(i1, n1, lst1, lst2) \neq 0) \\ \rightarrow (\text{strcmp}(i1, n1, lst1, lst2) \\ = \text{idifference}(\text{get-nth}(\text{strcmp-j}(i1, n1, lst1, lst2), lst1), \\ \text{get-nth}(\text{strcmp-j}(i1, n1, lst1, lst2), lst2)))) \end{aligned}$$

THEOREM: strcmp-thm2

$$(\text{strcmp}(0, n1, lst1, lst2) \neq 0) \rightarrow \text{strcmp-sk}(n1, lst1, lst2)$$

EVENT: Disable strcmp-j-1.

EVENT: Disable strcmp-j-2.

; some properties of strcat.

THEOREM: strcpy1-get-1

$$(j < i1) \rightarrow (\text{get-nth}(j, \text{strcpy1}(i1, lst1, i2, n2, lst2)) = \text{get-nth}(j, lst1))$$

DEFINITION:

$$\begin{aligned} \text{strlen1}(i, n, lst) \\ = \text{if } i < n \\ \quad \text{then if } \text{get-nth}(i, lst) = \text{NULL} \text{ then 0} \\ \quad \quad \text{else } 1 + \text{strlen1}(1 + i, n, lst) \text{ endif} \\ \text{else 0 endif} \end{aligned}$$

THEOREM: strlen1-strlen
 $\text{strlen1}(i, n, lst) = (\text{strlen}(i, n, lst) - i)$
 EVENT: Disable strlen1-strlen.

 THEOREM: strcpy1-get-2
 $((i1 + \text{strlen1}(i2, n2, lst2)) < j)$
 $\rightarrow (\text{get-nth}(j, \text{strcpy1}(i1, lst1, i2, n2, lst2)) = \text{get-nth}(j, lst1))$

 THEOREM: get-nth-plus-0
 $(j \notin \mathbf{N}) \rightarrow (\text{get-nth}(i + j, lst) = \text{get-nth}(i, lst))$

 THEOREM: plus-sub1-add1
 $((x + (1 + y)) - 1) = (x + y)$

 THEOREM: strcpy1-get-3
 $((i1 \leq j) \wedge (j < (i1 + \text{strlen1}(i2, n2, lst2))))$
 $\rightarrow (\text{get-nth}(j, \text{strcpy1}(i1, lst1, i2, n2, lst2)))$
 $= \text{get-nth}(i2 + (j - i1), lst2))$

 $; \text{all } 0 \leq j < (\text{strlen lst1}) \text{ lst1}'[j] == \text{lst1}[j].$

 THEOREM: strcat-get-1
 $(j < \text{strlen}(0, n1, lst1))$
 $\rightarrow (\text{get-nth}(j, \text{strcat}(n1, lst1, n2, lst2)) = \text{get-nth}(j, lst1))$

 $; \text{all } (\text{strlen lst1}) \leq j < (\text{strlen lst1} + \text{strlen lst2}) \text{ lst1}'[j] == \text{lst2}[j].$

 THEOREM: strcat-get-2
 $(\text{stringp}(0, n1, lst1))$
 $\wedge (\text{strlen}(0, n1, lst1) \leq j)$
 $\wedge (j < (\text{strlen}(0, n1, lst1) + \text{strlen}(0, n2, lst2))))$
 $\rightarrow (\text{get-nth}(j, \text{strcat}(n1, lst1, n2, lst2)))$
 $= \text{get-nth}(j - \text{strlen}(0, n1, lst1), lst2))$

 $; \text{some properties about strchr.}$
 $; \text{lst[strchr]} == ch, \text{ if strchr returns non-f.}$

 THEOREM: strchr-thm1
 $\text{strchr}(i, n, lst, ch) \rightarrow (\text{get-nth}(\text{strchr}(i, n, lst, ch), lst) = ch)$

 $; \text{all } i \leq j < \text{strchr}, \text{ lst}[j] \neq ch. \text{ i.e. strchr is the first one.}$

 THEOREM: strchr-thm2
 $((i \leq j) \wedge (j < \text{strchr}(i, n, lst, ch))) \rightarrow (\text{get-nth}(j, lst) \neq ch)$

```

; all i <= j < strlen, lst[j] =\= ch, if strchr returns f.

THEOREM: strchr-thm3
((\neg strchr(i, n, lst, ch)) \wedge (i \leq j) \wedge (j < \text{strlen}(i, n, lst)))
\rightarrow (\text{get-nth}(j, lst) \neq ch)

; ch is not equal to the null character, if strchr returns f.

THEOREM: strchr-thm4
(stringp(i, n, lst) \wedge (\neg strchr(i, n, lst, ch))) \rightarrow (ch \neq 0)

; strcpy and strchr.

THEOREM: strcpy-strchr
strchr(i, n, strcpy(i, lst1, n, lst2), ch) = strchr(i, n, lst2, ch)

; some properties about memset.
; lemmas about memset1.

THEOREM: memset1-get-1
(j < i) \rightarrow (\text{get-nth}(j, \text{memset1}(i, n, lst, ch)) = \text{get-nth}(j, lst))

; all i <= j < i+n, lst'[j] == ch.

THEOREM: memset1-thm1
((i \leq j) \wedge (j < (i + n)))
\rightarrow (\text{get-nth}(j, \text{memset1}(i, n, lst, ch)) = ch)

THEOREM: memset1-thm2
(((i + n) \leq j) \wedge (n \neq 0))
\rightarrow (\text{get-nth}(j, \text{memset1}(i, n, lst, ch)) = \text{get-nth}(j, lst))

; all i <= j < n, lst'[j] == ch.

THEOREM: memset-thm1
((i \leq j) \wedge (j < n)) \rightarrow (\text{get-nth}(j, \text{memset}(n, lst, ch)) = ch)

; all j >= n, lst'[j] == lst[j].

THEOREM: memset-thm2
((n \leq j) \wedge (n \in \mathbb{N}))
\rightarrow (\text{get-nth}(j, \text{memset}(n, lst, ch)) = \text{get-nth}(j, lst))

; some properties about memchr.

THEOREM: memchr1-thm1
memchr1(i, n, lst, ch) \rightarrow (\text{get-nth}(\text{memchr1}(i, n, lst, ch), lst) = ch)

```

```

; lst[memchr] == ch, if memchr returns non-f.

THEOREM: memchr-thm1
memchr(n, lst, ch) → (get-nth(memchr(n, lst, ch), lst) = ch)

THEOREM: memchr1-thm2
((i ≤ j) ∧ (j < memchr1(i, n, lst, ch))) → (get-nth(j, lst) ≠ ch)

; all j < memchr, lst[j] =\= ch. i.e. memchr is the first one.

THEOREM: memchr-thm2
(j < memchr(n, lst, ch)) → (get-nth(j, lst) ≠ ch)

THEOREM: memchr1-thm3
((¬ memchr1(i, n, lst, ch)) ∧ (i ≤ j) ∧ (j < (i + n)))
→ (get-nth(j, lst) ≠ ch)

; all j < n, lst[j] =\= ch, if memchr returns f.

THEOREM: memchr-thm3
((¬ memchr(n, lst, ch)) ∧ (j < n)) → (get-nth(j, lst) ≠ ch)

; some properties about strrchr.

THEOREM: strrchr-la1
(j ∈ N) → strrchr(i, n, lst, ch, j)

THEOREM: strrchr-strrchr-la
(¬ strrchr(i, n, lst, ch)) → (strrchr(i, n, lst, ch, j) = j)

THEOREM: strrchr-la2
strrchr(i, n, lst, ch) → (get-nth(strrchr(i, n, lst, ch, j), lst) = ch)

; strrchr finds ch in the string.

THEOREM: strrchr-thm1
strrchr(i, n, lst, ch, f) → (get-nth(strrchr(i, n, lst, ch, f), lst) = ch)

THEOREM: strchr-get
((i ≤ j) ∧ (j < strlen(i, n, lst))) → strchr(i, n, lst, get-nth(j, lst))

THEOREM: strrchr-la3
(strrchr(i, n, lst, ch)
 ∧ (strrchr(i, n, lst, ch, k) < j)
 ∧ (j < strlen(i, n, lst)))
→ (get-nth(j, lst) ≠ ch)

```

; strrchr returns the last one.

THEOREM: strrchr-thm2

$$\begin{aligned} & (\text{strrchr}(i, n, \text{lst}, \text{ch}, \mathbf{f}) \\ & \wedge (\text{strrchr}(i, n, \text{lst}, \text{ch}, \mathbf{f}) < j) \\ & \wedge (j < \text{strlen}(i, n, \text{lst}))) \\ \rightarrow & (\text{get-nth}(j, \text{lst}) \neq \text{ch}) \end{aligned}$$

; no means none.

THEOREM: strrchr-thm3

$$\begin{aligned} & ((\neg \text{strrchr}(i, n, \text{lst}, \text{ch}, k)) \wedge (i \leq j) \wedge (j < \text{strlen}(i, n, \text{lst}))) \\ \rightarrow & (\text{get-nth}(j, \text{lst}) \neq \text{ch}) \end{aligned}$$

; ch is not equal to the null character, if strrchr returns f.

THEOREM: strrchr-thm4

$$(\text{stringp}(i, n, \text{lst}) \wedge (\neg \text{strrchr}(i, n, \text{lst}, \text{ch}, k))) \rightarrow (\text{ch} \neq 0)$$

; some properties about memcmp.

THEOREM: memcmp1-id

$$\text{memcmp1}(i, n, \text{lst}, \text{lst}) = 0$$

THEOREM: memcmp-id

$$\text{memcmp}(n, \text{lst}, \text{lst}) = 0$$

THEOREM: memcmp1-thm1

$$\begin{aligned} & (\text{lst-of-chrp}(\text{lst1}) \\ & \wedge \text{lst-of-chrp}(\text{lst2})) \\ & \wedge (\text{memcmp1}(i, n, \text{lst1}, \text{lst2}) = 0) \\ & \wedge (i \leq j) \\ & \wedge (j < (i + n))) \\ \rightarrow & (\text{get-nth}(j, \text{lst1}) = \text{get-nth}(j, \text{lst2})) \end{aligned}$$

THEOREM: memcmp-thm1

$$\begin{aligned} & (\text{lst-of-chrp}(\text{lst1}) \\ & \wedge \text{lst-of-chrp}(\text{lst2})) \\ & \wedge (\text{memcmp}(n, \text{lst1}, \text{lst2}) = 0) \\ & \wedge (j < n)) \\ \rightarrow & (\text{get-nth}(j, \text{lst1}) = \text{get-nth}(j, \text{lst2})) \end{aligned}$$

DEFINITION:

$\text{memcmp-sk}(n, \text{lst1}, \text{lst2})$

$$\begin{aligned} \leftrightarrow & \exists j (\forall i ((i < j) \rightarrow (\text{get-nth}(i, \text{lst1}) = \text{get-nth}(i, \text{lst2})))) \\ & \wedge (\text{memcmp}(n, \text{lst1}, \text{lst2}) \\ & = \text{idifference}(\text{get-nth}(j, \text{lst1}), \text{get-nth}(j, \text{lst2})))) \end{aligned}$$

DEFINITION:

```
memcmp-j(i, n, lst1, lst2)
=  if get-nth(i, lst1) = get-nth(i, lst2)
  then if (n - 1) = 0 then fix(i)
        else memcmp-j(1 + i, n - 1, lst1, lst2) endif
  else fix(i) endif
```

THEOREM: memcmp-j-1

$$((i < \text{memcmp-j}(i1, n1, lst1, lst2)) \wedge (i1 \leq i)) \\ \rightarrow (\text{get-nth}(i, lst1) = \text{get-nth}(i, lst2))$$

THEOREM: memcmp-j-2

$$(\text{memcmp1}(i1, n1, lst1, lst2) \neq 0) \\ \rightarrow (\text{memcmp1}(i1, n1, lst1, lst2) \\ = \text{idifference}(\text{get-nth}(\text{memcmp-j}(i1, n1, lst1, lst2), lst1), \\ \text{get-nth}(\text{memcmp-j}(i1, n1, lst1, lst2), lst2)))$$

THEOREM: memcmp-thm2

$$(\text{memcmp}(n, lst1, lst2) \neq 0) \rightarrow \text{memcmp-sk}(n, lst1, lst2)$$

EVENT: Disable memcmp-j-1.

EVENT: Disable memcmp-j-2.

; some properties about strncat.

THEOREM: strcpy2-get-1

$$(j < i1) \rightarrow (\text{get-nth}(j, \text{strcpy2}(i1, lst1, i2, n2, lst2)) = \text{get-nth}(j, lst1))$$

THEOREM: strlen1-ii

$$\text{strlen1}(i, i, lst) = 0$$

THEOREM: strcpy2-get-2

$$((i1 \leq j) \wedge (j < (i1 + \text{strlen1}(i2, i2 + n2, lst2)))) \\ \rightarrow (\text{get-nth}(j, \text{strcpy2}(i1, lst1, i2, n2, lst2)) \\ = \text{get-nth}((i2 + j) - i1, lst2))$$

; all j < (strlen lst1), lst'[j] == lst1[j].

THEOREM: strncat-get-1

$$(j < \text{strlen}(0, n1, lst1)) \\ \rightarrow (\text{get-nth}(j, \text{strncat}(n1, lst1, n2, lst2)) = \text{get-nth}(j, lst1))$$

; all (strlen lst1) <= j < (strlen lst1)+(strlen lst2),
; lst'[j] == lst2[j-(strlen lst1)].

```

THEOREM: strncat-get-2
(stringp (0, n1, lst1)
         ∧ (strlen (0, n1, lst1) ≤ j)
         ∧ (j < (strlen (0, n1, lst1) + strlen (0, n2, lst2))))
→ (get-nth (j, strncat (n1, lst1, n2, lst2))
            = get-nth (j - strlen (0, n1, lst1), lst2))

; some properties about strncpy.

THEOREM: zero-list1-la
(j < i) → (get-nth (j, zero-list1 (i, n, lst)) = get-nth (j, lst))

THEOREM: zero-list1-get
((i ≤ j) ∧ (j < ((i + n) - 1)))
→ (get-nth (j, zero-list1 (i, n, lst)) = 0)

THEOREM: zero-list-get
((i ≤ j) ∧ (j < (i + n)))
→ (get-nth (j, zero-list (i, n, lst)) = 0)

THEOREM: strncpy1-get
(j < i) → (get-nth (j, strncpy1 (i, n, lst1, lst2)) = get-nth (j, lst1))

EVENT: Disable zero-list.

THEOREM: strncpy1-strlen
strlen (i, i + n, strncpy1 (i, n, lst1, lst2)) = strlen (i, i + n, lst2)

THEOREM: strcpy-0
strncpy (0, lst1, lst2) = lst1

; the length of (strncpy lst1 lst2) equals the length of lst2.

THEOREM: strncpy-strlen
strlen (0, n, strncpy (n, lst1, lst2))
= if n ≈ 0 then strlen (0, n, lst1)
  else strlen (0, n, lst2) endif

THEOREM: strncpy1-cpy
((i ≤ j) ∧ (j < strlen (i, i + n, lst2)))
→ (get-nth (j, strncpy1 (i, n, lst1, lst2)) = get-nth (j, lst2))

; all j < (strlen lst2), lst'[j] == lst2[j]. 

THEOREM: strncpy-cpy
(j < strlen (0, n, lst2))
→ (get-nth (j, strncpy (n, lst1, lst2)) = get-nth (j, lst2))

```

THEOREM: strncpy1-0s
 $((\text{strlen}(i, i + n, \text{lst2}) \leq j) \wedge (j < (i + n)))$
 $\rightarrow (\text{get-nth}(j, \text{strncpy1}(i, n, \text{lst1}, \text{lst2})) = 0)$
 $; \text{all } (\text{strlen } \text{lst2}) \leq j < n, \text{ lst}'[j] == 0.$

THEOREM: strncpy-0s
 $((\text{strlen}(0, n, \text{lst2}) \leq j) \wedge (j < n))$
 $\rightarrow (\text{get-nth}(j, \text{strncpy}(n, \text{lst1}, \text{lst2})) = 0)$
 $; \text{some properties about strncmp}.$

THEOREM: strncmp1-id
 $\text{strncmp1}(i, n, \text{lst}, \text{lst}) = 0$

THEOREM: strncmp-id
 $\text{strncmp}(n, \text{lst}, \text{lst}) = 0$

THEOREM: strncmp1-thm1
 $(\text{lst-of-chrp}(\text{lst1})$
 $\wedge \text{lst-of-chrp}(\text{lst2})$
 $\wedge (\text{strncmp1}(i, n, \text{lst1}, \text{lst2}) = 0)$
 $\wedge (i \leq j)$
 $\wedge (j < \text{strlen}(i, i + n, \text{lst1})))$
 $\rightarrow (\text{get-nth}(j, \text{lst1}) = \text{get-nth}(j, \text{lst2}))$

THEOREM: strncmp-thm1
 $(\text{lst-of-chrp}(\text{lst1})$
 $\wedge \text{lst-of-chrp}(\text{lst2})$
 $\wedge (\text{strncmp}(n, \text{lst1}, \text{lst2}) = 0)$
 $\wedge (j < \text{strlen}(0, n, \text{lst1})))$
 $\rightarrow (\text{get-nth}(j, \text{lst1}) = \text{get-nth}(j, \text{lst2}))$

DEFINITION:
 $\text{strncmp-j}(i, n, \text{lst1}, \text{lst2})$
 $= \text{if } \text{get-nth}(i, \text{lst1}) = \text{get-nth}(i, \text{lst2})$
 $\quad \text{then if } \text{get-nth}(i, \text{lst1}) = 0 \text{ then fix}(i)$
 $\quad \quad \text{elseif } (n - 1) = 0 \text{ then fix}(i)$
 $\quad \quad \text{else } \text{strncmp-j}(1 + i, n - 1, \text{lst1}, \text{lst2}) \text{ endif}$
 $\quad \text{else fix}(i) \text{ endif}$

DEFINITION:
 $\text{strncmp-sk}(n, \text{lst1}, \text{lst2})$
 $\leftrightarrow \exists j (\forall i ((i < j) \rightarrow (\text{get-nth}(i, \text{lst1}) = \text{get-nth}(i, \text{lst2}))))$
 $\wedge (\text{strncmp}(n, \text{lst1}, \text{lst2})$
 $\quad = \text{idifference}(\text{get-nth}(j, \text{lst1}), \text{get-nth}(j, \text{lst2}))))$

THEOREM: strncmp-j-1
 $((i < \text{strcmp-j}(i1, n1, lst1, lst2)) \wedge (i1 \leq i))$
 $\rightarrow (\text{get-nth}(i, lst1) = \text{get-nth}(i, lst2))$

THEOREM: strncmp-j-2
 $(\text{strcmp1}(i1, n1, lst1, lst2) \neq 0)$
 $\rightarrow (\text{strcmp1}(i1, n1, lst1, lst2))$
 $= \text{idifference}(\text{get-nth}(\text{strcmp-j}(i1, n1, lst1, lst2), lst1),$
 $\text{get-nth}(\text{strcmp-j}(i1, n1, lst1, lst2), lst2)))$

THEOREM: strcmp-thm2
 $(\text{strcmp}(n, lst1, lst2) \neq 0) \rightarrow \text{strcmp-sk}(n, lst1, lst2)$

EVENT: Disable strcmp-j-1.

EVENT: Disable strcmp-j-2.

; some properties about strpbrk.

THEOREM: strpbrk-thm1
let j **be** strpbrk($i1, n1, lst1, n2, lst2$)
in
 $j \rightarrow \text{strchr1}(0, n2, lst2, \text{get-nth}(j, lst1))$ **endlet**

THEOREM: strpbrk-thm2
 $((i1 \leq j) \wedge (j < \text{strpbrk}(i1, n1, lst1, n2, lst2)))$
 $\rightarrow (\neg \text{strchr1}(0, n2, lst2, \text{get-nth}(j, lst1)))$

THEOREM: strpbrk-thm3
 $((\neg \text{strpbrk}(i1, n1, lst1, n2, lst2))$
 $\wedge (i1 \leq j))$
 $\wedge (j < \text{strlen}(i1, n1, lst1)))$
 $\rightarrow (\neg \text{strchr1}(0, n2, lst2, \text{get-nth}(j, lst1)))$

; strchr1, which is used to specify several string functions, is a variant
; of strchr. strchr1 does not have the ‘thm4’ of strchr.

THEOREM: strchr1-thm1
 $\text{strchr1}(i, n, lst, ch) \rightarrow (\text{get-nth}(\text{strchr1}(i, n, lst, ch), lst) = ch)$

THEOREM: strchr1-thm2
 $((i \leq j) \wedge (j < \text{strchr1}(i, n, lst, ch))) \rightarrow (\text{get-nth}(j, lst) \neq ch)$

THEOREM: strchr1-thm3
 $((\neg \text{strchr1}(i, n, lst, ch)) \wedge (i \leq j) \wedge (j < \text{strlen}(i, n, lst)))$
 $\rightarrow (\text{get-nth}(j, lst) \neq ch)$

; some properties of strcspn.

THEOREM: strcspn-thm1
let j be strcspn($i1, n1, lst1, n2, lst2$)
in
 $j \rightarrow \text{strchr}(0, n2, lst2, \text{get-nth}(j, lst1))$ endlet

THEOREM: strcspn-thm2
 $((i1 \leq j) \wedge (j < \text{strcspn}(i1, n1, lst1, n2, lst2)))$
 $\rightarrow (\neg \text{strchr}(0, n2, lst2, \text{get-nth}(j, lst1)))$

THEOREM: strcspn-thm3
 $((\neg \text{strcspn}(i1, n1, lst1, n2, lst2))$
 $\wedge (i1 \leq j)$
 $\wedge (j < \text{strlen}(i1, n1, lst1)))$
 $\rightarrow (\neg \text{strchr}(0, n2, lst2, \text{get-nth}(j, lst1)))$

; some properties of strspn.

THEOREM: strspn-thm1
let j be strspn($i1, n1, lst1, n2, lst2$)
in
 $j \rightarrow (\neg \text{strchr1}(0, n2, lst2, \text{get-nth}(j, lst1)))$ endlet

THEOREM: strspn-thm2
 $((i1 \leq j) \wedge (j < \text{strspn}(i1, n1, lst1, n2, lst2)))$
 $\rightarrow \text{strchr1}(0, n2, lst2, \text{get-nth}(j, lst1))$

THEOREM: strspn-thm3
 $((\neg \text{strspn}(i1, n1, lst1, n2, lst2))$
 $\wedge (i1 \leq j)$
 $\wedge (j < \text{strlen}(i1, n1, lst1)))$
 $\rightarrow \text{strchr1}(0, n2, lst2, \text{get-nth}(j, lst1))$

; a generalized defn for strncmp.

DEFINITION:

strncmp2($i, n, lst1, j, lst2$)
= if $n \simeq 0$ then 0
elseif $\text{get-nth}(i, lst1) = \text{get-nth}(j, lst2)$
then if $\text{get-nth}(i, lst1) = 0$ then 0
else strncmp2($1 + i, n - 1, lst1, 1 + j, lst2$) endif
else idifference($\text{get-nth}(i, lst1), \text{get-nth}(j, lst2)$) endif

THEOREM: strncmp2-non-numberp
 $(i \notin \mathbb{N}) \rightarrow (\text{strncmp2}(i, n, lst1, j, lst2) = \text{strncmp2}(0, n, lst1, j, lst2))$

```

THEOREM: strncmp2-0
strncmp2(i, 0, lst1, j, lst2) = 0

THEOREM: strncmp1-strncmp2
(n ≠ 0) → (strcmp1(i, n, lst1, lst2) = strcmp2(i, n, lst1, i, lst2))

THEOREM: strncmp-strncmp2
strcmp(n, lst1, lst2) = strcmp2(0, n, lst1, 0, lst2)

THEOREM: strcmp2-mcdr-1
strcmp2(i1, n, mcdr(j, lst1), i2, lst2) = strcmp2(i1 + j, n, lst1, i2, lst2)

THEOREM: strcmp2-mcdr-2
strcmp2(i1, n, lst1, i2, mcdr(j, lst2)) = strcmp2(i1, n, lst1, i2 + j, lst2)

; a lemma of strlen and mcdr.

THEOREM: strlen-cdr
(i ∈ N) → (strlen(i, n - 1, mcdr(1, lst)) = (strlen(1 + i, n, lst) - 1))

THEOREM: strlen-mcdr
(i ∈ N)
→ (strlen(i, n, mcdr(k, lst)) = (strlen(i + k, k + n, lst) - k))

; some properties of strstr.
; a lemma for strstr-thm1.

THEOREM: strstr1-thm1
let j be strstr1(i, n1, lst1, n2, lst2, len)
in
((j ∈ N) ∧ (n = (1 + len)))
→ (strcmp2(j, n, lst1, 0, lst2) = 0) endlet

; all 0 <= i < (strlen lst2), lst1[j+i] == lst2[i], if strstr == j =\= 0.

THEOREM: strstr-thm1
let j be strstr(n1, lst1, n2, lst2)
in
(j ∈ N) → (strcmp(strlen(0, n2, lst2), mcdr(j, lst1), lst2) = 0) endlet

; a few lemmas about strchr1.

THEOREM: strchr1-nth
((i ≤ j) ∧ (j < strlen(i, n, lst)))
→ (strchr1(i, n, lst, get-nth(j, lst)) ∈ N)

```

THEOREM: strlen-strchr1-nth
 $((i \leq j) \wedge (j < \text{strlen}(i, n, lst)))$
 $\rightarrow (\text{strlen}(1 + \text{strchr1}(i, n, lst, \text{get-nth}(j, lst)), n, lst)$
 $= \text{strlen}(i, n, lst))$

THEOREM: strlen-strchr1
 $(\text{strchr1}(i, n, lst, 0) \in \mathbb{N}) \rightarrow (\text{strchr1}(i, n, lst, 0) \not< \text{strlen}(i, n, lst))$

THEOREM: strchr1-ch0
 $\neg \text{strchr1}(i, n, lst, 0)$

THEOREM: strchr1-nth-first
 $((\text{strchr1}(i, n, lst, \text{get-nth}(j, lst)) \in \mathbb{N}) \wedge (i \leq j))$
 $\rightarrow (j \not< \text{strchr1}(i, n, lst, \text{get-nth}(j, lst)))$

`; a lemma for strstr-thm2.`

THEOREM: strstr1-thm2
 $(\text{lst-of-chrp}(lst1)$
 $\wedge \text{lst-of-chrp}(lst2)$
 $\wedge (i \leq j)$
 $\wedge (j < \text{strstr1}(i, n1, lst1, n2, lst2, len))$
 $\wedge (n = (1 + len)))$
 $\rightarrow (\text{strncmp2}(j, n, lst1, 0, lst2) \neq 0)$

`; all j < (strstr lst1 lst2), (strncmp (mcdr j lst1) lst2) =\= 0.`

THEOREM: strstr-thm2
 $(\text{lst-of-chrp}(lst1)$
 $\wedge \text{lst-of-chrp}(lst2)$
 $\wedge (j < \text{strstr}(n1, lst1, n2, lst2))$
 $\wedge (n2 \neq 0))$
 $\rightarrow (\text{strncmp}(\text{strlen}(0, n2, lst2), \text{mcdr}(j, lst1), lst2) \neq 0)$

`; a lemma for strstr-thm3.`

THEOREM: strstr1-thm3
 $(\text{lst-of-chrp}(lst1)$
 $\wedge \text{lst-of-chrp}(lst2)$
 $\wedge (\neg \text{strstr1}(i, n1, lst1, n2, lst2, len))$
 $\wedge (i \leq j)$
 $\wedge (j < \text{strlen}(i, n1, lst1))$
 $\wedge (n = (1 + len)))$
 $\rightarrow (\text{strncmp2}(j, n, lst1, 0, lst2) \neq 0)$

`; all j < (strlen lst1), (strncmp (mcdr j lst1) lst2) =\= 0.`

THEOREM: strstr-thm3

$$\begin{aligned}
 & (\text{lst-of-chrp}(lst1) \\
 & \wedge \text{lst-of-chrp}(lst2)) \\
 & \wedge (\neg \text{strstr}(n1, lst1, n2, lst2)) \\
 & \wedge (j < \text{strlen}(0, n1, lst1)) \\
 & \wedge (n2 \neq 0)) \\
 \rightarrow & (\text{strncmp}(\text{strlen}(0, n2, lst2), \text{mcdr}(j, lst1), lst2) \neq 0)
 \end{aligned}$$

; some properties of strtok.

THEOREM: strtok-thm1

$$\begin{aligned}
 & \text{let } i \text{ be strspn}(0, n1, lst1, n2, lst2) \\
 & \text{in} \\
 & ((\text{nat-to-uint}(s1) \neq 0) \wedge (\text{get-nth}(i, lst1) = 0)) \\
 \rightarrow & ((\text{strtok-tok}(s1, last, n1, lst1, n2, lst2) = 0) \\
 & \wedge (\text{strtok-last}(s1, last, n1, lst1, n2, lst2) = 0) \\
 & \wedge (\text{strtok-lst}(n1, lst1, n2, lst2) = lst1)) \text{ endlet}
 \end{aligned}$$

THEOREM: strtok-thm2

$$\begin{aligned}
 & \text{let } i \text{ be strspn}(0, n1, lst1, n2, lst2) \\
 & \text{in} \\
 & ((\text{nat-to-uint}(s1) \neq 0) \\
 & \wedge (\text{get-nth}(i, lst1) \neq 0) \\
 & \wedge (\text{get-nth}(\text{strcspn}(i, n1, lst1, n2, lst2), lst1) = 0) \\
 & \wedge \text{stringp}(0, n1, lst1) \\
 & \wedge (n1 \in \mathbb{N})) \\
 \rightarrow & ((\text{strtok-tok}(s1, last, n1, lst1, n2, lst2) \\
 & = \text{add}(32, s1, \text{strspn}^*(0, 0, n1, lst1, n2, lst2))) \\
 & \wedge (\text{strtok-last}(s1, last, n1, lst1, n2, lst2) = 0) \\
 & \wedge (\text{strtok-lst}(n1, lst1, n2, lst2) = lst1)) \text{ endlet}
 \end{aligned}$$

THEOREM: strspn-strchr1

$$\begin{aligned}
 & \text{strspn}(i1, n1, lst1, n2, lst2) \\
 \rightarrow & (\neg \text{strchr1}(0, n2, lst2, \text{get-nth}(\text{strspn}(i1, n1, lst1, n2, lst2), lst1)))
 \end{aligned}$$

THEOREM: strchr-strchr1

$$(ch \neq 0) \rightarrow (\text{strchr}(i, n, lst, ch) = \text{strchr1}(i, n, lst, ch))$$

THEOREM: strspn-true

$$(\text{stringp}(i, n1, lst1) \wedge \text{stringp}(0, n2, lst2)) \rightarrow \text{strspn}(i, n1, lst1, n2, lst2)$$

THEOREM: strtok-thm3

$$\begin{aligned}
 & \text{let } i^* \text{ be strspn}^*(0, 0, n1, lst1, n2, lst2), \\
 & \quad i \text{ be strspn}(0, n1, lst1, n2, lst2) \\
 & \text{in}
 \end{aligned}$$

```

((nat-to-uint (s1) ≠ 0)
 ∧ (get-nth (i, lst1) ≠ 0)
 ∧ (get-nth (strcspn (i, n1, lst1, n2, lst2), lst1) ≠ 0)
 ∧ stringp (0, n1, lst1)
 ∧ stringp (0, n2, lst2)
 ∧ (n1 ∈ ℒ))
→ ((strtok-tok (s1, last, n1, lst1, n2, lst2)
 = add (32, s1, strspn* (0, 0, n1, lst1, n2, lst2)))
 ∧ (strtok-last (s1, last, n1, lst1, n2, lst2)
 = add (32,
       s1,
       add (32, strcspn* (i*, i, n1, lst1, n2, lst2), 1)))
 ∧ (get-nth (strcspn (i, n1, lst1, n2, lst2),
               strtok-lst (n1, lst1, n2, lst2))
 = 0)) endlet

```

THEOREM: strtok-thm4

```

(strtok-tok (0, 0, n1, lst1, n2, lst2) = 0)
∧ (strtok-last (0, 0, n1, lst1, n2, lst2) = 0)

```

THEOREM: strtok-thm5

```

let i be strspn (0, n1, lst1, n2, lst2)
in
((nat-to-uint (last) ≠ 0)
 ∧ (get-nth (i, lst1) ≠ 0)
 ∧ (get-nth (strcspn (i, n1, lst1, n2, lst2), lst1) = 0)
 ∧ stringp (0, n1, lst1)
 ∧ (n1 ∈ ℒ))
→ ((strtok-tok (0, last, n1, lst1, n2, lst2)
 = add (32, last, strspn* (0, 0, n1, lst1, n2, lst2)))
 ∧ (strtok-last (0, last, n1, lst1, n2, lst2) = 0)
 ∧ (strtok-lst (n1, lst1, n2, lst2) = lst1)) endlet

```

THEOREM: strtok-thm6

```

let i* be strspn* (0, 0, n1, lst1, n2, lst2),
      i be strspn (0, n1, lst1, n2, lst2)
in
((nat-to-uint (last) ≠ 0)
 ∧ (get-nth (i, lst1) ≠ 0)
 ∧ (get-nth (strcspn (i, n1, lst1, n2, lst2), lst1) ≠ 0)
 ∧ stringp (0, n1, lst1)
 ∧ stringp (0, n2, lst2)
 ∧ (n1 ∈ ℒ))
→ ((strtok-tok (0, last, n1, lst1, n2, lst2)
 = add (32, last, strspn* (0, 0, n1, lst1, n2, lst2)))

```

```

 $\wedge \quad (\text{strtok-last} (0, \text{last}, n1, \text{lst1}, n2, \text{lst2})$ 
 $= \quad \text{add} (32,$ 
 $\quad \quad \text{last},$ 
 $\quad \quad \text{add} (32, \text{strcspn}^*(i^*, i, n1, \text{lst1}, n2, \text{lst2}), 1)))$ 
 $\wedge \quad (\text{get-nth} (\text{strcspn} (i, n1, \text{lst1}, n2, \text{lst2}),$ 
 $\quad \quad \text{strtok-lst} (n1, \text{lst1}, n2, \text{lst2}))$ 
 $= \quad 0)) \text{ endlet}$ 

```

EVENT: Disable strchr-strchr1.

```

; some properties of memmove.
; memmove is equivalent to lstmov.

```

THEOREM: mmov1-lst-cpy
 $\text{mmov1-lst} (i, \text{lst1}, \text{lst2}, n) = \text{lstcpy} (i, \text{lst1}, i, \text{lst2}, n)$

THEOREM: mmovn-lst
 $\text{mmovn-lst} (h, \text{lst1}, \text{lst2}, i, n) = \text{mmov1-lst} (i, \text{lst1}, \text{lst2}, h * n)$

THEOREM: lstcpy-nonnumberp
 $(i1 \notin \mathbb{N})$
 $\rightarrow ((\text{lstcpy} (i1, \text{lst1}, i2, \text{lst2}, n) = \text{lstcpy} (0, \text{lst1}, i2, \text{lst2}, n))$
 $\quad \wedge \quad (\text{lstcpy} (i2, \text{lst1}, i1, \text{lst2}, n) = \text{lstcpy} (i2, \text{lst1}, 0, \text{lst2}, n)))$

THEOREM: lstcpy-lstcpy-0
 $\text{lstcpy} (h1, \text{lstcpy} (0, \text{lst1}, 0, \text{lst2}, h1), h1, \text{lst2}, h2)$
 $= \quad \text{lstcpy} (0, \text{lst1}, 0, \text{lst2}, h1 + h2)$

THEOREM: mmov1-lst1-cpy
 $(n \leq i)$
 $\rightarrow (\text{mmov1-lst1} (i, \text{lst1}, \text{lst2}, n) = \text{lstcpy} (i - n, \text{lst1}, i - n, \text{lst2}, n))$

THEOREM: lstcpy-lstcpy-1
 $((j1 = (h1 + i1)) \wedge (j2 = (h1 + i2)))$
 $\rightarrow (\text{lstcpy} (i1, \text{lstcpy} (j1, \text{lst1}, j2, \text{lst2}, h2), i2, \text{lst2}, h1)$
 $\quad \quad = \quad \text{lstcpy} (i1, \text{lst1}, i2, \text{lst2}, h1 + h2))$

THEOREM: times-lessp-dual
 $(x < y) \rightarrow ((x < (y * z)) = (z \neq 0))$

THEOREM: mmovn-lst1-mmov1-lst
 $((h * n) \leq i)$
 $\rightarrow (\text{mmovn-lst1} (h, \text{lst1}, \text{lst2}, i, n)$
 $\quad \quad = \quad \text{mmov1-lst} (i - (h * n), \text{lst1}, \text{lst2}, h * n))$

THEOREM: quotient-shrink-fast-1

$$(x < (y * (x \div y))) = \mathbf{f}$$

THEOREM: memmove-mmov1-lst

$$\text{memmove}(\text{str1}, \text{str2}, n, \text{lst1}, \text{lst2})$$

$$= \begin{aligned} & \text{if } n \simeq 0 \text{ then } \text{lst1} \\ & \text{elseif nat-to-uint}(\text{str1}) = \text{nat-to-uint}(\text{str2}) \text{ then } \text{lst2} \\ & \text{else mmov1-lst}(0, \text{lst1}, \text{lst2}, n) \text{ endif} \end{aligned}$$

; for k < i, lstcpy(lst1, lst2)[k] == lst1[k].

THEOREM: lstcpy-get-1

$$(k < i) \rightarrow (\text{get-nth}(k, \text{lstcpy}(i, \text{lst1}, j, \text{lst2}, n)) = \text{get-nth}(k, \text{lst1}))$$

; for i <= i1 < i+n, lstcpy(lst1, lst2)[i1] == lst2[i1].

THEOREM: lstcpy-get-2

$$((i \leq i1) \wedge (i1 < (i + n)))$$

$$\rightarrow \begin{aligned} & (\text{get-nth}(i1, \text{lstcpy}(i, \text{lst1}, j, \text{lst2}, n)) \\ & = \text{get-nth}((j + i1) - i, \text{lst2})) \end{aligned}$$

; for 0 <= i < n, mmov1-lst(lst1, lst2)[i] == lst2[i].

THEOREM: mmov1-lst-thm1

$$(i1 < n) \rightarrow (\text{get-nth}(i1, \text{mmov1-lst}(0, \text{lst1}, \text{lst2}, n)) = \text{get-nth}(i1, \text{lst2}))$$

; for 0 <= i < n, memmove(lst1, lst2, n)[i] == lst2[i].

THEOREM: memmove-thm1

$$(i < n)$$

$$\rightarrow (\text{get-nth}(i, \text{memmove}(\text{str1}, \text{str2}, n, \text{lst1}, \text{lst2})) = \text{get-nth}(i, \text{lst2}))$$

Index

- add, 16–18
- exists, 4, 8, 11
- fix-int, 1
- forall, 4, 8, 11
- get-nth, 2–19
- get-nth-plus-0, 5
- idifference, 1, 4, 8, 9, 11–13
- idifference-equal-0, 1
- idifference-numberp, 1
- ilessp, 1
- ilessp-lessp, 1
- lst-of-chrp, 3, 8, 11, 15, 16
- lstcpy, 18, 19
- lstcpy-get-1, 19
- lstcpy-get-2, 19
- lstcpy-lstcpy-0, 18
- lstcpy-lstcpy-1, 18
- lstcpy-nonnumberp, 18
- mcdr, 14–16
- memchr, 7
- memchr-thm1, 7
- memchr-thm2, 7
- memchr-thm3, 7
- memchr1, 6, 7
- memchr1-thm1, 6
- memchr1-thm2, 7
- memchr1-thm3, 7
- memcmp, 8, 9
- memcmp-id, 8
- memcmp-j, 9
- memcmp-j-1, 9
- memcmp-j-2, 9
- memcmp-sk, 8, 9
- memcmp-thm1, 8
- memcmp-thm2, 9
- memcmp1, 8, 9
- memcmp1-id, 8
- memcmp1-thm1, 8
- memmove, 19
- memmove-mmov1-lst, 19
- memmove-thm1, 19
- memset, 6
- memset-thm1, 6
- memset-thm2, 6
- memset1, 6
- memset1-get-1, 6
- memset1-thm1, 6
- memset1-thm2, 6
- mmov1-lst, 18, 19
- mmov1-lst-cpy, 18
- mmov1-lst-thm1, 19
- mmov1-lst1, 18
- mmov1-lst1-cpy, 18
- mmovn-lst, 18
- mmovn-lst1, 18
- mmovn-lst1-mmov1-lst, 18
- mmovn-mmov1-lst, 18
- nat-to-uint, 16, 17, 19
- null, 4
- plus-sub1-add1, 5
- quotient-shrink-fast-1, 19
- strcat, 5
- strcat-get-1, 5
- strcat-get-2, 5
- strchr, 5–7, 13, 16
- strchr-get, 7
- strchr-strchr1, 16
- strchr-thm1, 5
- strchr-thm2, 5
- strchr-thm3, 6
- strchr-thm4, 6
- strchr1, 12–16
- strchr1-ch0, 15
- strchr1-nth, 14

strchr1-nth-first, 15
strchr1-thm1, 12
strchr1-thm2, 12
strchr1-thm3, 12
strcmp, 3, 4
strcmp-antisym, 3
strcmp-id, 3
strcmp-j, 4
strcmp-j-1, 4
strcmp-j-2, 4
strcmp-sk, 4
strcmp-strcpy, 3
strcmp-thm1, 3
strcmp-thm2, 4
strcmp-trans-1, 3
strcpy, 2, 3, 6
strcpy-0, 10
strcpy-cpy, 3
strcpy-get-1, 2
strcpy-strchr, 6
strcpy-stringp, 3
strcpy-strlen, 2
strcpy1, 4, 5
strcpy1-get-1, 4
strcpy1-get-2, 5
strcpy1-get-3, 5
strcpy2, 9
strcpy2-get-1, 9
strcpy2-get-2, 9
strcspn, 13, 16–18
strcspn*, 17, 18
strcspn-thm1, 13
strcspn-thm2, 13
strcspn-thm3, 13
stringp, 2, 3, 5, 6, 8, 10, 16, 17
strlen, 2, 3, 5–16
strlen-01, 2
strlen-0p, 2
strlen-cdr, 14
strlen-ii, 2
strlen-ii1, 2
strlen-lb, 2
strlen-mcdr, 14
strlen-non0p, 2
strlen-strchr1, 15
strlen-strchr1-nth, 15
strlen-ub, 2
strlen1, 4, 5, 9
strlen1-ii, 9
strlen1-strlen, 5
strncat, 9, 10
strncat-get-1, 9
strncat-get-2, 10
strncmp, 11, 12, 14–16
strncmp-id, 11
strncmp-j, 11, 12
strncmp-j-1, 12
strncmp-j-2, 12
strncmp-sk, 11, 12
strncmp-strncmp2, 14
strncmp-thm1, 11
strncmp-thm2, 12
strncmp1, 11, 12, 14
strncmp1-id, 11
strncmp1-strncmp2, 14
strncmp1-thm1, 11
strncmp2, 13–15
strncmp2-0, 14
strncmp2-mcdr-1, 14
strncmp2-mcdr-2, 14
strncmp2-non-numberp, 13
strncpy, 10, 11
strncpy-0s, 11
strncpy-cpy, 10
strncpy-strlen, 10
strncpy1, 10, 11
strncpy1-0s, 11
strncpy1-cpy, 10
strncpy1-get, 10
strncpy1-strlen, 10
strpbrk, 12
strpbrk-thm1, 12
strpbrk-thm2, 12
strpbrk-thm3, 12
strrchr, 7, 8
strrchr-la1, 7
strrchr-la2, 7
strrchr-la3, 7

strrchr-strchr-la, 7
strrchr-thm1, 7
strrchr-thm2, 8
strrchr-thm3, 8
strrchr-thm4, 8
strspn, 13, 16, 17
strspn*, 16, 17
strspn-strchr1, 16
strspn-thm1, 13
strspn-thm2, 13
strspn-thm3, 13
strspn-true, 16
strstr, 14–16
strstr-thm1, 14
strstr-thm2, 15
strstr-thm3, 16
strstr1, 14, 15
strstr1-thm1, 14
strstr1-thm2, 15
strstr1-thm3, 15
strtok-last, 16–18
strtok-lst, 16–18
strtok-thm1, 16
strtok-thm2, 16
strtok-thm3, 16
strtok-thm4, 17
strtok-thm5, 17
strtok-thm6, 17
strtok-tok, 16, 17

times-lessp-dual, 18

zero-list, 10
zero-list-get, 10
zero-list1, 10
zero-list1-get, 10
zero-list1-la, 10