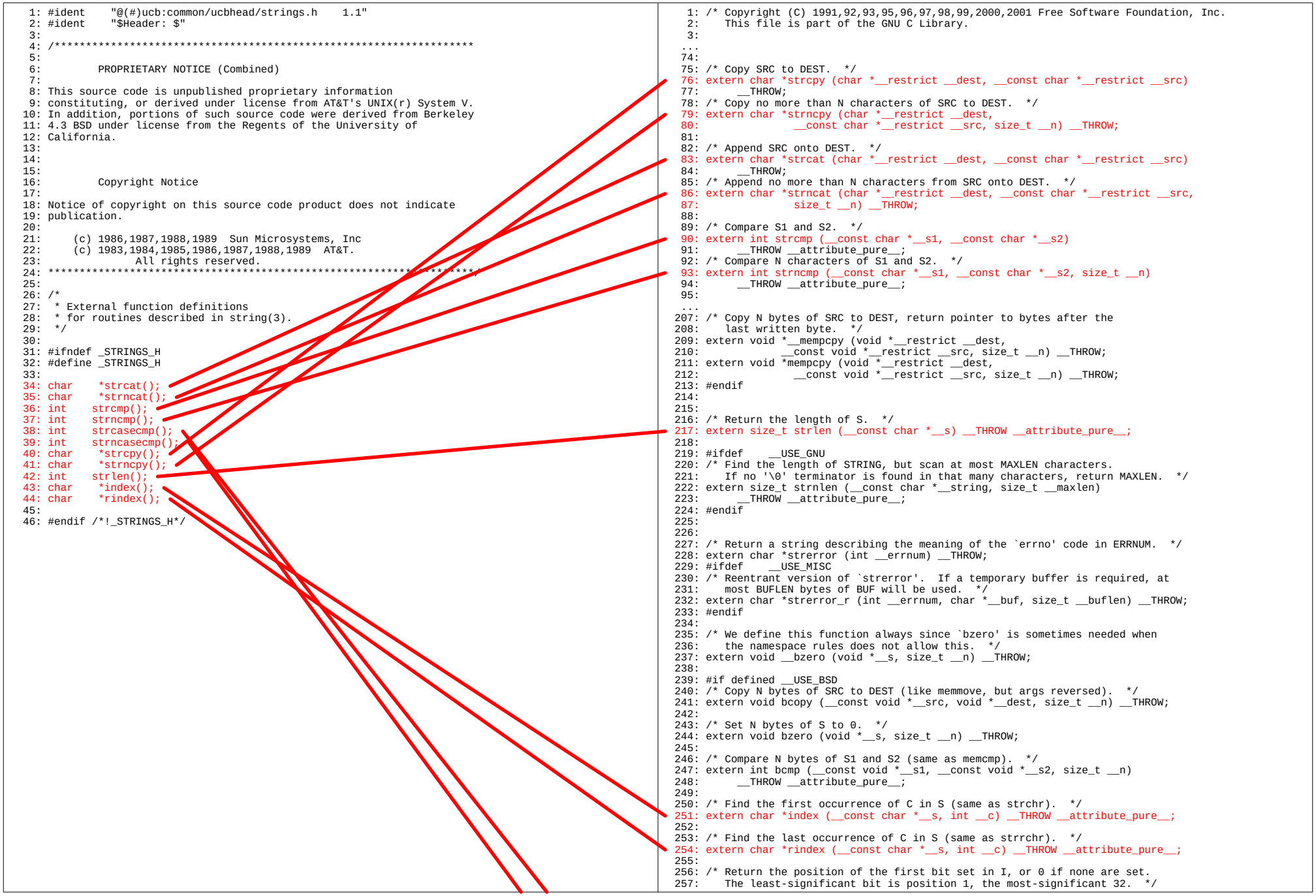
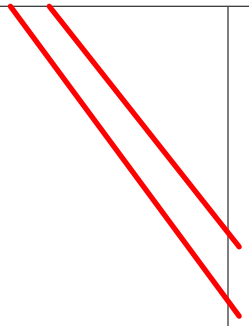


```
 1: #ident    "@(#)ucb:common/ucbhead/strings.h    1.1"
 2: #ident    "$Header: $"
 3:
 4: /*****************************************************************
 5:
 6:         PROPRIETARY NOTICE (Combined)
 7:
 8: This source code is unpublished proprietary information
 9: constituting, or derived under license from AT&T's UNIX(r) System V.
10: In addition, portions of such source code were derived from Berkeley
11: 4.3 BSD under license from the Regents of the University of
12: California.
13:
14:
15:
16:         Copyright Notice
17:
18: Notice of copyright on this source code product does not indicate
19: publication.
20:
21:     (c) 1986,1987,1988,1989  Sun Microsystems, Inc
22:     (c) 1983,1984,1985,1986,1987,1988,1989  AT&T.
23:               All rights reserved.
24: *****************************************************************/
25:
26: /*
27:  * External function definitions
28:  * for routines described in string(3).
29:  */
30:
31: #ifndef _STRINGS_H
32: #define _STRINGS_H
33:
34: char    *strcat();
35: char    *strncat();
36: int    strcmp();
37: int    strncmp();
38: int    strcasecmp();
39: int    strncasecmp();
40: char    *strcpy();
41: char    *strncpy();
42: int    strlen();
43: char    *index();
44: char    *rindex();
45:
46: #endif /*!_STRINGS_H*/
```

```
  1: /* Copyright (C) 1991,92,93,95,96,97,98,99,2000,2001 Free Software Foundation, Inc.
  2:    This file is part of the GNU C Library.
  3:
...
 74:
 75: /* Copy SRC to DEST.  */
 76: extern char *strcpy (char *__restrict __dest, __const char *__restrict __src)
 77:      __THROW;
 78: /* Copy no more than N characters of SRC to DEST.  */
 79: extern char *strncpy (char *__restrict __dest,
 80:               __const char *__restrict __src, size_t __n) __THROW;
 81:
 82: /* Append SRC onto DEST.  */
 83: extern char *strcat (char *__restrict __dest, __const char *__restrict __src)
 84:      __THROW;
 85: /* Append no more than N characters from SRC onto DEST.  */
 86: extern char *strncat (char *__restrict __dest, __const char *__restrict __src,
 87:               size_t __n) __THROW;
 88:
 89: /* Compare S1 and S2.  */
 90: extern int strcmp (__const char *__s1, __const char *__s2)
 91:      __THROW __attribute_pure__;
 92: /* Compare N characters of S1 and S2.  */
 93: extern int strncmp (__const char *__s1, __const char *__s2, size_t __n)
 94:      __THROW __attribute_pure__;
 95:
...
207: /* Copy N bytes of SRC to DEST, return pointer to bytes after the
208:    last written byte.  */
209: extern void *__mempcpy (void *__restrict __dest,
210:             __const void *__restrict __src, size_t __n) __THROW;
211: extern void *mempcpy (void *__restrict __dest,
212:               __const void *__restrict __src, size_t __n) __THROW;
213: #endif
214:
215:
216: /* Return the length of S.  */
217: extern size_t strlen (__const char *__s) __THROW __attribute_pure__;
218:
219: #ifdef    __USE_GNU
220: /* Find the length of STRING, but scan at most MAXLEN characters.
221:    If no '\0' terminator is found in that many characters, return MAXLEN.  */
222: extern size_t strnlen (__const char *__string, size_t __maxlen)
223:      __THROW __attribute_pure__;
224: #endif
225:
226:
227: /* Return a string describing the meaning of the `errno' code in ERRNUM.  */
228: extern char *strerror (int __errnum) __THROW;
229: #ifdef    __USE_MISC
230: /* Reentrant version of `strerror'.  If a temporary buffer is required, at
231:    most BUFLEN bytes of BUF will be used.  */
232: extern char *strerror_r (int __errnum, char *__buf, size_t __buflen) __THROW;
233: #endif
234:
235: /* We define this function always since `bzero' is sometimes needed when
236:    the namespace rules does not allow this.  */
237: extern void __bzero (void *__s, size_t __n) __THROW;
238:
239: #if defined __USE_BSD
240: /* Copy N bytes of SRC to DEST (like memmove, but args reversed).  */
241: extern void bcopy (__const void *__src, void *__dest, size_t __n) __THROW;
242:
243: /* Set N bytes of S to 0.  */
244: extern void bzero (void *__s, size_t __n) __THROW;
245:
246: /* Compare N bytes of S1 and S2 (same as memcmp).  */
247: extern int bcmp (__const void *__s1, __const void *__s2, size_t __n)
248:      __THROW __attribute_pure__;
249:
250: /* Find the first occurrence of C in S (same as strchr).  */
251: extern char *index (__const char *__s, int __c) __THROW __attribute_pure__;
252:
253: /* Find the last occurrence of C in S (same as strrchr).  */
254: extern char *rindex (__const char *__s, int __c) __THROW __attribute_pure__;
255:
256: /* Return the position of the first bit set in I, or 0 if none are set.
257:    The least-significant bit is position 1, the most-significant 32.  */
```


(For readability, not all matches are indicated.)



```
258: extern int ffs (int __i) __THROW __attribute__ ((__const__));
259:
260: /* The following two functions are non-standard but necessary for non-32 bit
261:    platforms.  */
262: # ifdef    __USE_GNU
263: extern int ffsl (long int __l) __THROW __attribute__ ((__const__));
264: #  ifdef __GNUC__
265: __extension__ extern int ffsll (long long int __ll)
266:      __THROW __attribute__ ((__const__));
267: #  endif
268: # endif
269:
270: /* Compare S1 and S2, ignoring case.  */
271: extern int strcasecmp (__const char *__s1, __const char *__s2)
272:      __THROW __attribute_pure__;
273:
274: /* Compare no more than N chars of S1 and S2, ignoring case.  */
275: extern int strncasecmp (__const char *__s1, __const char *__s2, size_t __n)
276:      __THROW __attribute_pure__;
277: #endif /* Use BSD.  */
 ...
364:
365: #endif /* string.h  */
```


(For readability, not all matches are indicated.)