```
 1: // $Id: mkstampfiles.c,v 1.3 2012-11-14 21:11:26-08 - - $
 2:
 3: //
 4: // Open several files and write into them.  Then use utime(2)
 5: // to force a time stamp onto them.
 6:
 7: #include <errno.h>
 8: #include <libgen.h>
 9: #include <stdio.h>
10: #include <stdlib.h>
11: #include <string.h>
12: #include <sys/types.h>
13: #include <time.h>
14: #include <utime.h>
15:
16: char *execname = NULL;
17: int exit_status = EXIT_SUCCESS;
18:
19: void maketimefile (time_t when) {
20:    char filename[64];
21:    sprintf (filename, "timestamp.%016lX", when);
22:    FILE *file = fopen (filename, "w");
23:    if (file == NULL) {
24:       fprintf (stderr, "%s: %s: %s\n",
25:                execname, filename, strerror (errno));
26:       exit_status = EXIT_FAILURE;
27:       return;
28:    }
29:    printf ("fopen (%s): OK\n", filename);
30:    char buffer[64];
31:    strftime (buffer, sizeof buffer, "%c %Z", localtime (&when));
32:    fprintf (file, "%s\n", buffer);
33:    strftime (buffer, sizeof buffer, "%c %Z", gmtime (&when));
34:    fprintf (file, "%s\n", buffer);
35:    fclose (file);
36:    struct utimbuf utimbuf;
37:    utimbuf.actime = when;
38:    utimbuf.modtime = when;
39:    utime (filename, &utimbuf);
40: }
41:
42: int main (int argc, char **argv) {
43:    execname = basename (argv[0]);
44:    time_t now = time (NULL);
45:    const time_t DAYS = 24 * 60 * 60;
46:    maketimefile (-0x80000000L);
47:    maketimefile (0);
48:    maketimefile (now - 200 * DAYS);
49:    maketimefile (now);
50:    maketimefile (now + 200 * DAYS);
51:    maketimefile (0x7FFFFFFF);
52:    return exit_status;
53: }
54:
```

```
 1: // $Id: showlink.c,v 1.1 2012-02-28 17:27:55-08 - - $
 2:
 3: //
 4: // Example of how to display a symbolic link.  This program is *NOT*
 5: // called readlink, because one such already exists on Linux.
 6: // Print a symlink or an error message.  PATH_MAX is the max length
 7: // of a pathname.
 8: //
 9: // From man -s 2 readlink:
10: //
11: //    DESCRIPTION
12: //        readlink() places the contents of the symbolic link  path
13: //        in  the  buffer  buf,  which has size bufsiz.  readlink()
14: //        does not append a null byte to buf.  It will truncate the
15: //        contents  (to a length of bufsiz characters), in case the
16: //        buffer is too small to hold all of the contents.
17: //
18: // This looks like a bizarre design bug.
19: //
20:
21: #include <errno.h>
22: #include <libgen.h>
23: #include <limits.h>
24: #include <stdio.h>
25: #include <stdlib.h>
26: #include <string.h>
27: #include <unistd.h>
28:
29: int main (int argc, char **argv) {
30:    int exit_status = EXIT_SUCCESS;
31:    for (int argi = 1; argi < argc; ++argi) {
32:       char *pathname = argv[argi];
33:       char linkname[PATH_MAX + 1];
34:       ssize_t retval = readlink (pathname, linkname, sizeof linkname);
35:       if (retval >= 0) {
36:          linkname[retval < PATH_MAX + 1 ? retval : PATH_MAX] = '\0';
37:          printf ("%s -> \"%s\"\n", pathname, linkname);
38:       }else {
39:          exit_status = EXIT_FAILURE;
40:          fflush (NULL);
41:          fprintf (stderr, "%s: %s: %s\n",
42:                   basename (argv[0]), pathname, strerror (errno));
43:          fflush (NULL);
44:       }
45:    }
46:    return exit_status;
47: }
```

```
 1: // $Id: sometimes.c,v 1.6 2012-11-28 20:40:49-08 - - $
 2:
 3: #include <limits.h>
 4: #include <stdio.h>
 5: #include <stdlib.h>
 6: #include <time.h>
 7:
 8: int main (void) {
 9:    time_t times[] = {INT_MIN, 0, time (NULL), INT_MAX};
10:    char *timeformat = "%a %d %b %Y %H:%M:%S %Z";
11:    printf ("sizeof (time_t) = %ld\n", sizeof (time_t));
12:    for (size_t i = 0; i < sizeof times / sizeof *times; ++i) {
13:       time_t when = times[i];
14:       struct tm *tm = localtime (&when);
15:       char buffer_local[64];
16:       strftime (buffer_local, sizeof buffer_local, timeformat, tm);
17:       char buffer_gmt[64];
18:       tm = gmtime (&when);
19:       strftime (buffer_gmt, sizeof buffer_gmt, timeformat, tm);
20:       printf ("0x%08X = %s = %s\n",
21:               (int) times[i], buffer_local, buffer_gmt);
22:    }
23:    return EXIT_SUCCESS;
24: }
25:
```

```
 1: #!/usr/bin/perl
 2: # $Id: l8stat.perl,v 1.1 2012-02-29 19:04:02-08 - - $
 3: use POSIX qw(strftime);
 4: $0 =~ s|.*/||;
 5: $days180 = 180 * 24 * 3600;
 6: for $file (@ARGV ? @ARGV : ".") {
 7:    ($_, $_, $mode, $_, $_, $_, $_, $size, $_, $mtime, $_, $_, $_)
 8:          = lstat $file;
 9:    print STDERR "$0: $file: $!\n" and next unless defined $size;
10:    $fmt = $mtime < $^T - $days180 || $^T + $days180 < $mtime
11:         ? "%b %e  %Y" : "%b %e %R";
12:    printf "%06o %9d %s %s%s\n",
13:           $mode, $size, (strftime $fmt, localtime $mtime), $file,
14:           defined ($link = readlink $file) ? " -> $link" : "";
15: }
```

```
 1: 100600     32090 Nov 15 11:56 Listing.pdf
 2: 100600     24383 Nov 15 11:56 Listing.ps
 3: 040700      2048 Feb 22 13:35 RCS
 4: 100700       586 Feb 22 13:34 l8stat.perl
 5: 100700       146 Feb 22 13:35 mklis
 6: 100700     12207 Feb 22 13:34 mkstampfiles
 7: 100600      1419 Feb 22 13:34 mkstampfiles.c
 8: 100600         0 Feb 22 13:35 output.lis
 9: 100700     10649 Feb 22 13:34 showlink
10: 100600      1496 Feb 22 13:34 showlink.c
11: 100700      9450 Feb 22 13:34 sometimes
12: 100600       787 Feb 22 13:34 sometimes.c
13: 120755        11 Feb 22 13:34 this-is-a-symlink -> l8stat.perl
14: 100600        58 Dec 31  1969 timestamp.0000000000000000
15: 100600        58 Aug 12  2011 timestamp.000000004E45D2EA
16: 100600        58 Aug 12  2011 timestamp.000000004E45D323
17: 100600        58 Feb 28  2012 timestamp.000000004F4D7EEA
18: 100600        58 Feb 28  2012 timestamp.000000004F4D7F23
19: 100600        58 Apr 28  2012 timestamp.000000004F9CCD6A
20: 100600        58 Apr 28  2012 timestamp.000000004F9CCD82
21: 100600        58 Sep 15 18:27 timestamp.0000000050552AEA
22: 100600        58 Sep 15 18:28 timestamp.0000000050552B23
23: 100600        58 Nov 14 21:11 timestamp.0000000050A4796A
24: 100600        58 Nov 14 21:11 timestamp.0000000050A47982
25: 100600        58 Jun  2 22:11 timestamp.0000000051AC256A
26: 100600        58 Jun  2 22:11 timestamp.0000000051AC2582
27: 100600        58 Jan 18  2038 timestamp.000000007FFFFFFF
28: 100600         0 Nov 14 21:11 timestamp.7FFFFFFFFFFFFFFF
29: 100600        58 Dec 13  1901 timestamp.FFFFFFFF80000000
30: 040700      4096 Feb 22 13:35 .
31: 040700      2048 Feb 22 13:34 ..
32: 040700      2048 Feb 22 13:30 ../..
```