```
 1: // $Id: voidstar.c,v 1.33 2012-11-16 14:59:17-08 - - $
 2:
 3: //
 4: // Simple example of void* processing in C.
 5: // The function process takes an array and a function and
 6: // processes the array according to the rules given by a
 7: // function.
 8: //
 9:
10: #include <ctype.h>
11: #include <math.h>
12: #include <stdio.h>
13: #include <stdlib.h>
14: #include <string.h>
15: #include <values.h>
16:
17: //
18: // Process an array by applying a function to each element.
19: //
20: void process (void *base,   // of the array
21:               size_t nelem, // number of elements
22:               size_t size,  // size of one element
23:               void (*function) (void *)) {
24:    for (size_t index = 0; index < nelem; ++index) {
25:       void *element = (char *) base + index * size;
26:       function (element);
27:    }
28: }
29:
30: //
31: // Array of strings with two processing functions.
32: //
33: char *strings[] = {
34:    "hello", "world", "foo", "bar", "baz", "qux",
35:    "this", "is", "a", "test",
36: };
37:
38: void strdupthem (void *string) {
39:    char **chars = (char **) string;
40:    *chars = strdup (*chars);
41: }
42:
43: void capitalize (void *string) {
44:    for (char *chars = * (char **) string; *chars != '\0'; ++chars) {
45:       *chars = toupper (*chars);
46:    }
47: }
48:
49: void printstr (void *string) {
50:    (void) printf (" %s", * (char **) string);
51: }
52:
53: void freestr (void *string) {
54:    char *str = * (char **) string;
55:    free (str);
56:    str = NULL;
57: }
58:
```

```
 59:
 60: //
 61: // Array of doubles with two processing functions.
 62: //
 63:
 64: double numbers[] = {6.02e23, 287, -472, 0, 6e-22, MAXDOUBLE};
 65:
 66: void exponent (void *number) {
 67:    double *value = (double *) number;
 68:    *value = log10 (*value);
 69: }
 70:
 71: void printnum (void *number) {
 72:    (void) printf (" %10.3e", * (double *) number);
 73: }
 74:
 75: //
 76: // Main function to exercise them.
 77: //
 78:
 79: int main (void) {
 80:
 81:    size_t stringdim = sizeof strings / sizeof *strings;
 82:    process (strings, stringdim, sizeof *strings, printstr);
 83:    (void) printf ("\n");
 84:    process (strings, stringdim, sizeof *strings, strdupthem);
 85:    process (strings, stringdim, sizeof *strings, capitalize);
 86:    process (strings, stringdim, sizeof *strings, printstr);
 87:    process (strings, stringdim, sizeof *strings, freestr);
 88:    (void) printf ("\n");
 89:
 90:    size_t numberdim = sizeof numbers / sizeof *numbers;
 91:    process (numbers, numberdim, sizeof *numbers, printnum);
 92:    (void) printf ("\n");
 93:    process (numbers, numberdim, sizeof *numbers, exponent);
 94:    process (numbers, numberdim, sizeof *numbers, printnum);
 95:    (void) printf ("\n");
 96:
 97:    return EXIT_SUCCESS;
 98: }
 99:
100: //TEST// runprog -x voidstar.lis valgrind --leak-check=full voidstar
101: //TEST// mkpspdf Listing.ps voidstar.c voidstar.c.log voidstar.lis
102:
```

```
1: @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@ mkc: starting voidstar.c
2: voidstar.c: $Id: voidstar.c,v 1.33 2012-11-16 14:59:17-08 - - $
3: gcc -g -O0 -Wall -Wextra -std=gnu99 voidstar.c -o voidstar -lm
4: rm -f voidstar.o
5: @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@ mkc: finished voidstar.c
```

```
    1:
    2: ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
    3: log: voidstar.log
    4: ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
    5:
    6:      1  Script  : /afs/cats.ucsc.edu/courses/cmps012b-wm/bin/runprog
    7:      2  limit c :    0 max core file size (KB)
    8:      3  limit f : 4194303 max output file size (KB)
    9:      4  limit t : 4294967295 max CPU time (sec)
   10:      5  stdin   : /dev/null
   11:      6  stdout  : voidstar.out
   12:      7  stderr  : voidstar.err
   13:      8  log     : voidstar.log
   14:      9  listing : voidstar.lis
   15:     10  Command : valgrind --leak-check=full voidstar
   16:     11  starting: pid 26912: 14:59:17.00
   17:     12  finished: pid 26912: 14:59:18.00, real 1.00, user 0.30, sys 0.04
   18:     13  pstatus: 0x0000 EXIT STATUS = 0
   19:
   20: ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
   21: stdin: /dev/null
   22: ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
   23:
   24:
   25: ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
   26: stdout: voidstar.out
   27: ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
   28:
   29:      1   hello world foo bar baz qux this is a test
   30:      2   HELLO WORLD FOO BAR BAZ QUX THIS IS A TEST
   31:      3    6.020e+23  2.870e+02 -4.720e+02  0.000e+00  6.000e-22 1.798e+308
   32:      4    2.378e+01  2.458e+00        nan       -inf -2.122e+01  3.083e+02
   33:
   34: ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
   35: stderr: voidstar.err
   36: ::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
   37:
   38:      1  ==26912== Memcheck, a memory error detector
   39:      2  ==26912== Copyright (C) 2002-2010, and GNU GPL'd, by Julian Seward et al
.
   40:      3  ==26912== Using Valgrind-3.6.0 and LibVEX; rerun with -h for copyright i
nfo
   41:      4  ==26912== Command: voidstar
   42:      5  ==26912==
   43:      6  ==26912==
   44:      7  ==26912== HEAP SUMMARY:
   45:      8  ==26912==     in use at exit: 0 bytes in 0 blocks
   46:      9  ==26912==   total heap usage: 10 allocs, 10 frees, 43 bytes allocated
   47:     10  ==26912==
   48:     11  ==26912== All heap blocks were freed -- no leaks are possible
   49:     12  ==26912==
   50:     13  ==26912== For counts of detected and suppressed errors, rerun with: -v
   51:     14  ==26912== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 6 from 6)
```