```
 1: // $Id: voidstar.c,v 1.42 2013-10-18 12:06:30-07 - - $
 2:
 3: //
 4: // Simple example of void* processing in C.
 5: // The function process takes an array and a function and
 6: // applies the function to each element of the array.
 7: //
 8:
 9: #include <ctype.h>
10: #include <math.h>
11: #include <stdio.h>
12: #include <stdlib.h>
13: #include <string.h>
14: #include <values.h>
15:
16: //
17: // Process an array by applying a function to each element.
18: //
19: void process (void *base,   // of the array
20:               size_t nelem, // number of elements
21:               size_t size,  // size of one element
22:               void (*function) (void*)) {
23:    for (size_t index = 0; index < nelem; ++index) {
24:       void *element = (char*) base + index * size;
25:       function (element);
26:    }
27: }
28:
29: //
30: // Array of strings with two processing functions.
31: //
32: char *strings[] = {
33:    "hello", "world", "foo", "bar", "baz", "qux",
34:    "this", "is", "a", "test",
35: };
36:
37: void strdupthem (void *string) {
38:    char **chars = (char**) string;
39:    *chars = strdup (*chars);
40: }
41:
42: void capitalize (void *string) {
43:    for (char *chars = *(char**) string; *chars != '\0'; ++chars) {
44:       *chars = toupper (*chars);
45:    }
46: }
47:
48: void printstr (void *string) {
49:    (void) printf (" %s", *(char**) string);
50: }
51:
52: void freestr (void *string) {
53:    char *str = *(char**) string;
54:    free (str);
55:    str = NULL;
56: }
57:
```

```
 58:
 59: //
 60: // Array of doubles with two processing functions.
 61: //
 62:
 63: double numbers[] = {6.02e23, 287, -472, 0, 6e-22, MAXDOUBLE};
 64:
 65: void exponent (void *number) {
 66:    double *value = (double*) number;
 67:    *value = log10 (*value);
 68: }
 69:
 70: void printnum (void *number) {
 71:    (void) printf (" %10.3g", *(double*) number);
 72: }
 73:
 74: //
 75: // Main function to exercise them.
 76: //
 77:
 78: int main (void) {
 79:
 80:    size_t stringdim = sizeof strings / sizeof *strings;
 81:    process (strings, stringdim, sizeof *strings, printstr);
 82:    (void) printf ("\n");
 83:    process (strings, stringdim, sizeof *strings, strdupthem);
 84:    process (strings, stringdim, sizeof *strings, capitalize);
 85:    process (strings, stringdim, sizeof *strings, printstr);
 86:    process (strings, stringdim, sizeof *strings, freestr);
 87:    (void) printf ("\n");
 88:
 89:    size_t numberdim = sizeof numbers / sizeof *numbers;
 90:    process (numbers, numberdim, sizeof *numbers, printnum);
 91:    (void) printf ("\n");
 92:    process (numbers, numberdim, sizeof *numbers, exponent);
 93:    process (numbers, numberdim, sizeof *numbers, printnum);
 94:    (void) printf ("\n");
 95:
 96:    return EXIT_SUCCESS;
 97: }
 98:
 99: /*
100: //TEST// valgrind --leak-check=full ./voidstar >voidstar.lis 2>&1
101: //TEST// mkpspdf voidstar.ps voidstar.c* voidstar.lis
102: */
103:
```

```
1: @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@ mkc: starting voidstar.c
2: voidstar.c:
3:      $Id: voidstar.c,v 1.42 2013-10-18 12:06:30-07 - - $
4: gcc -g -O0 -Wall -Wextra -rdynamic -std=gnu11 voidstar.c
5:          -o voidstar -lglut -lGLU -lGL -lX11 -lrt -lm
6: rm -f voidstar.o
7: @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@ mkc: finished voidstar.c
```

```
    1: ==18746== Memcheck, a memory error detector
    2: ==18746== Copyright (C) 2002-2013, and GNU GPL'd, by Julian Seward et al
.
    3: ==18746== Using Valgrind-3.10.1 and LibVEX; rerun with -h for copyright
info
    4: ==18746== Command: ./voidstar
    5: ==18746==
    6:  hello world foo bar baz qux this is a test
    7:  HELLO WORLD FOO BAR BAZ QUX THIS IS A TEST
    8:    6.02e+23        287       -472          0      6e-22   1.8e+308
    9:        23.8       2.46        nan       -inf      -21.2        308
   10: ==18746==
   11: ==18746== HEAP SUMMARY:
   12: ==18746==     in use at exit: 0 bytes in 0 blocks
   13: ==18746==   total heap usage: 11 allocs, 11 frees, 59 bytes allocated
   14: ==18746==
   15: ==18746== All heap blocks were freed -- no leaks are possible
   16: ==18746==
   17: ==18746== For counts of detected and suppressed errors, rerun with: -v
   18: ==18746== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 1 from 1)
```