```
 1: // $Id: genericlib.h,v 1.2 2014-05-15 20:35:11-07 - - $
 2:
 3: #ifndef __GENERICLIB_H__
 4: #define __GENERICLIB_H__
 5:
 6: #include <stdlib.h>
 7: #include <stdio.h>
 8:
 9: //
10: // Swap two chunks of storage using malloc and free for the
11: // temporary structure.
12: //
13: void swapm (void* this, void* that, size_t size);
14:
15: //
16: // Swap two chunks of storage using alloca, which is on
17: // the stack and does not have to be freed.
18: //
19: void swapa (void* this, void* that, size_t size);
20:
21: //
22: // Process an array by applying a function to each element,
23: // in place.  The elements may be modified.
24: //
25: void process (void* base,                // base address of the array
26:               size_t nelem,              // number of elements in it
27:               size_t size,               // sizeof one element
28:               void (*function) (void*)); // the processing function
29: //
30: // TRACE macro for start of functions.
31: //
32:
33: #define TRACE(FMT,...) printf ("%s:%d: %s (" FMT ")\n", \
34:                                __FILE__, __LINE__, __func__, \
35:                                __VA_ARGS__);
36:
37: #endif
38:
```

```
 1: // $Id: genericlib.c,v 1.3 2014-05-15 20:57:59-07 - - $
 2:
 3: #include <stdlib.h>
 4: #include <string.h>
 5:
 6: #include "genericlib.h"
 7:
 8: void swapm (void* this, void* that, size_t size) {
 9:    TRACE ("%p, %p, %zd", this, that, size);
10:    void *temp = malloc (size);
11:    printf ("%s: temp=%p\n", __func__, temp);
12:    memcpy (temp, this, size);
13:    memcpy (this, that, size);
14:    memcpy (that, temp, size);
15:    free (temp);
16: }
17:
18: void swapa (void* this, void* that, size_t size) {
19:    TRACE ("%p, %p, %zd", this, that, size);
20:    void *temp = alloca (size);
21:    printf ("%s: temp=%p\n", __func__, temp);
22:    memcpy (temp, this, size);
23:    memcpy (this, that, size);
24:    memcpy (that, temp, size);
25: }
26:
27: void process (void* base, size_t nelem, size_t size,
28:               void (*function) (void*)) {
29:    TRACE ("%p, %zd, %zd, %p", base, nelem, size, function);
30:    for (size_t index = 0; index < nelem; ++index) {
31:       void *element = (char*) base + index * size;
32:       function (element);
33:    }
34: }
35:
```

```
 1: // $Id: numberproc.c,v 1.1 2014-05-15 20:57:59-07 - - $
 2:
 3: //
 4: // Example of processing an array of numbers.
 5: //
 6:
 7: #include <ctype.h>
 8: #include <math.h>
 9: #include <stdio.h>
10: #include <stdlib.h>
11: #include <string.h>
12: #include <values.h>
13:
14: #include "genericlib.h"
15:
16:
17: double numbers[] = {6.02e23, 287, -472, 0, 6e-22, MAXDOUBLE};
18:
19: void log10ify (void *number) {
20:    TRACE ("%.15g", *(double*)number);
21:    double *value = (double*) number;
22:    *value = log10 (*value);
23: }
24:
25: void printnum (void *number) {
26:    TRACE ("%.15g", *(double*)number);
27: }
28:
29: int main (void) {
30:
31:    size_t numberdim = sizeof numbers / sizeof *numbers;
32:    process (numbers, numberdim, sizeof *numbers, printnum);
33:    (void) printf ("\n");
34:
35:    process (numbers, numberdim, sizeof *numbers, log10ify);
36:    (void) printf ("\n");
37:
38:    process (numbers, numberdim, sizeof *numbers, printnum);
39:    (void) printf ("\n");
40:
41:    return 0;
42: }
```

```
 1: // $Id: stringproc.c,v 1.1 2014-05-15 20:57:59-07 - - $
 2:
 3: //
 4: // Example of using genericlib to process strings.
 5: // Array of strings with two processing functions.
 6: //
 7:
 8: #include <ctype.h>
 9: #include <stdio.h>
10: #include <string.h>
11:
12: #include "genericlib.h"
13:
14: static char *strings[] = {"hello", "world", "foo", "bar", "baz", "qux"};
15:
16: void strdupthem (void *string) {
17:    TRACE ("%p->\"%s\"", string, *(char**)string);
18:    char **chars = (char**) string;
19:    *chars = strdup (*chars);
20: }
21:
22: void capitalize (void *string) {
23:    TRACE ("%p->\"%s\"", string, *(char**)string);
24:    for (char *chars = *(char**) string; *chars != '\0'; ++chars) {
25:       *chars = toupper (*chars);
26:    }
27: }
28:
29: void printstr (void *string) {
30:    TRACE ("%p->\"%s\"", string, *(char**)string);
31: }
32:
33: void freestr (void *string) {
34:    TRACE ("%p->\"%s\"", string, *(char**)string);
35:    char *str = *(char**) string;
36:    free (str);
37:    str = NULL;
38: }
39:
40: int main (void) {
41:
42:    size_t stringdim = sizeof strings / sizeof *strings;
43:    process (strings, stringdim, sizeof *strings, printstr);
44:    (void) printf ("\n");
45:    process (strings, stringdim, sizeof *strings, strdupthem);
46:    (void) printf ("\n");
47:    process (strings, stringdim, sizeof *strings, capitalize);
48:    (void) printf ("\n");
49:    process (strings, stringdim, sizeof *strings, printstr);
50:    (void) printf ("\n");
51:    process (strings, stringdim, sizeof *strings, freestr);
52:    (void) printf ("\n");
53:
54:    return 0;
55: }
56:
```

```
 1: // $Id: testswap.c,v 1.2 2014-05-15 20:37:32-07 - - $
 2:
 3: //
 4: // Example program showing testing of genericlib.
 5: //
 6:
 7: #include <stdio.h>
 8: #include <string.h>
 9:
10: #include "genericlib.h"
11:
12: int main (int argc, char** argv) {
13:    (void) argc;
14:    printf ("%s:\n\n", argv[0]);
15:
16:    double d1 = 3;
17:    double d2 = 6;
18:    printf ("d1 = %g, d2 = %g\n", d1, d2);
19:    swapm (&d1, &d2, sizeof (double));
20:    printf ("d1 = %g, d2 = %g\n\n", d1, d2);
21:
22:    char s1[] = "Hello, World.";
23:    char s2[] = "This is a test of swapa.";
24:    printf ("s1 = \"%s\", s2 = \"%s\"\n", s1, s2);
25:    swapa (s1, s2, strlen (s1));
26:    printf ("s1 = \"%s\", s2 = \"%s\"\n\n", s1, s2);
27:
28:    return 0;
29: }
```

```
 1: # $Id: Makefile,v 1.2 2014-05-15 20:57:59-07 - - $
 2:
 3: GCC      = gcc -g -O0 -Wall -Wextra -std=gnu11
 4: MKDEPS   = gcc -MM
 5:
 6: EXECBINS = numberproc stringproc testswap
 7: LIBSRC   = genericlib.h genericlib.c
 8: OBJECTS  = ${EXECBINS:=.o} genericlib.o
 9: SOURCES  = ${LIBSRC} ${EXECBINS:=.c} Makefile
10: DEPFILE  = Makefile.deps
11: LISFILES = ${SOURCES} ${DEPFILE}
12: LISTING  = Listing.ps
13:
14: all: ${EXECBINS}
15:
16: %: %.o genericlib.o
17:         ${GCC} -o $@ $^ -lm
18:
19: %.o: %.c
20:         ${GCC} -c $<
21:
22: ci: ${SOURCES}
23:         checksource ${SOURCES}
24:         cid + ${SOURCES}
25:
26: lis: ${SOURCES} ${DEPFILE} ${EXECBINS:=.out}
27:         mkpspdf ${LISTING} $^
28:
29: clean:
30:         - rm ${OBJECTS} ${EXECBINS:=.out}
31:
32: spotless: clean
33:         - rm ${EXECBINS} ${LISTING} ${LISTING:.ps=.pdf}
34:
35: %.out: %
36:         $< >$@ 2>&1
37:
38: ${DEPFILE}:
39:         ${MKDEPS} ${SOURCES} >${DEPFILE}
40:
41: dep:
42:         - rm ${DEPFILE}
43:         make --no-print-directory ${DEPFILE}
44:
45: include ${DEPFILE}
46:
47: again:
48:         make --no-print-directory spotless dep ${EXECBINS} lis
49:
```

```
1: genericlib.o: genericlib.h
2: genericlib.o: genericlib.c genericlib.h
3: numberproc.o: numberproc.c genericlib.h
4: stringproc.o: stringproc.c genericlib.h
5: testswap.o: testswap.c genericlib.h
```

```
 1: genericlib.c:29: process (0x600fa0, 6, 8, 0x400719)
 2: numberproc.c:26: printnum (6.02e+23)
 3: numberproc.c:26: printnum (287)
 4: numberproc.c:26: printnum (-472)
 5: numberproc.c:26: printnum (0)
 6: numberproc.c:26: printnum (6e-22)
 7: numberproc.c:26: printnum (1.79769313486232e+308)
 8:
 9: genericlib.c:29: process (0x600fa0, 6, 8, 0x4006b0)
10: numberproc.c:20: log10ify (6.02e+23)
11: numberproc.c:20: log10ify (287)
12: numberproc.c:20: log10ify (-472)
13: numberproc.c:20: log10ify (0)
14: numberproc.c:20: log10ify (6e-22)
15: numberproc.c:20: log10ify (1.79769313486232e+308)
16:
17: genericlib.c:29: process (0x600fa0, 6, 8, 0x400719)
18: numberproc.c:26: printnum (23.7795964912578)
19: numberproc.c:26: printnum (2.45788189673399)
20: numberproc.c:26: printnum (nan)
21: numberproc.c:26: printnum (-inf)
22: numberproc.c:26: printnum (-21.2218487496164)
23: numberproc.c:26: printnum (308.254715559917)
24:
```

```
 1: genericlib.c:29: process (0x601280, 6, 8, 0x4007a1)
 2: stringproc.c:30: printstr (0x601280->"hello")
 3: stringproc.c:30: printstr (0x601288->"world")
 4: stringproc.c:30: printstr (0x601290->"foo")
 5: stringproc.c:30: printstr (0x601298->"bar")
 6: stringproc.c:30: printstr (0x6012a0->"baz")
 7: stringproc.c:30: printstr (0x6012a8->"qux")
 8:
 9: genericlib.c:29: process (0x601280, 6, 8, 0x4006d0)
10: stringproc.c:17: strdupthem (0x601280->"hello")
11: stringproc.c:17: strdupthem (0x601288->"world")
12: stringproc.c:17: strdupthem (0x601290->"foo")
13: stringproc.c:17: strdupthem (0x601298->"bar")
14: stringproc.c:17: strdupthem (0x6012a0->"baz")
15: stringproc.c:17: strdupthem (0x6012a8->"qux")
16:
17: genericlib.c:29: process (0x601280, 6, 8, 0x40072e)
18: stringproc.c:23: capitalize (0x601280->"hello")
19: stringproc.c:23: capitalize (0x601288->"world")
20: stringproc.c:23: capitalize (0x601290->"foo")
21: stringproc.c:23: capitalize (0x601298->"bar")
22: stringproc.c:23: capitalize (0x6012a0->"baz")
23: stringproc.c:23: capitalize (0x6012a8->"qux")
24:
25: genericlib.c:29: process (0x601280, 6, 8, 0x4007a1)
26: stringproc.c:30: printstr (0x601280->"HELLO")
27: stringproc.c:30: printstr (0x601288->"WORLD")
28: stringproc.c:30: printstr (0x601290->"FOO")
29: stringproc.c:30: printstr (0x601298->"BAR")
30: stringproc.c:30: printstr (0x6012a0->"BAZ")
31: stringproc.c:30: printstr (0x6012a8->"QUX")
32:
33: genericlib.c:29: process (0x601280, 6, 8, 0x4007de)
34: stringproc.c:34: freestr (0x601280->"HELLO")
35: stringproc.c:34: freestr (0x601288->"WORLD")
36: stringproc.c:34: freestr (0x601290->"FOO")
37: stringproc.c:34: freestr (0x601298->"BAR")
38: stringproc.c:34: freestr (0x6012a0->"BAZ")
39: stringproc.c:34: freestr (0x6012a8->"QUX")
40:
```

```
 1: testswap:
 2:
 3: d1 = 3, d2 = 6
 4: genericlib.c:9: swapm (0x7fffed7ca418, 0x7fffed7ca410, 8)
 5: swapm: temp=0x1b5f010
 6: d1 = 6, d2 = 3
 7:
 8: s1 = "Hello, World.", s2 = "This is a test of swapa."
 9: genericlib.c:19: swapa (0x7fffed7ca400, 0x7fffed7ca3e0, 13)
10: swapa: temp=0x7fffed7ca360
11: s1 = "This is a tes", s2 = "Hello, World.t of swapa."
12:
```