

```
1: // $Id: oclib.oh,v 1.33 2016-11-30 14:45:59-08 - - $
2:
3: #ifndef __OCLIB_OH__
4: #define __OCLIB_OH__
5:
6: #ifdef __OCLIB_C__
7: void* xcalloc (int nelem, int size);
8: void __putb (char __b);
9: void __putc (char __c);
10: void __puti (int __i);
11: void __puts (char* __s);
12: void __endl (void);
13: int __getc (void);
14: char* __getw (void);
15: char* __getln (void);
16: char** __getargv (void);
17: void __exit (int status);
18: void ____assert_fail (char* expr, char* file, int line);
19:
20: #else
21: #define EOF (-1)
22: #define char int
23: #define bool int
24: #define true 1
25: #define false 0
26: #define assert(expr) \
27:     {if (!(expr)) ____assert_fail (#expr, __FILE__, __LINE__);}
28: void ____assert_fail (string expr, string file, int line);
29: void putb (int b);
30: void putc (char c);
31: void puti (int i);
32: void puts (string s);
33: void endl ();
34: int getc ();
35: string getw ();
36: string getln ();
37: string[] getargv ();
38: void exit (int status);
39:
40: #endif
41: #endif
42:
```

```
1: # 1 "oclib.oh"
2: # 1 "<built-in>"
3: # 1 "<command-line>"
4: # 1 "/usr/include/stdc-predef.h" 1 3 4
5: # 1 "<command-line>" 2
6: # 1 "oclib.oh"
7:
8:
9:
10:
11:
12:
13: void* xalloc (int nelem, int size);
14: void __putb (char __b);
15: void __putc (char __c);
16: void __puti (int __i);
17: void __puts (char* __s);
18: void __endl (void);
19: int __getc (void);
20: char* __getw (void);
21: char* __getln (void);
22: char** __getargv (void);
23: void __exit (int status);
24: void ____assert_fail (char* expr, char* file, int line);
```

```
1: # 1 "oclib.oh"
2: # 1 "<built-in>"
3: # 1 "<command-line>"
4: # 1 "/usr/include/stdc-predef.h" 1 3 4
5: # 1 "<command-line>" 2
6: # 1 "oclib.oh"
7: # 28 "oclib.oh"
8: void __assert_fail (string expr, string file, int line);
9: void putb (int b);
10: void putc (int c);
11: void puti (int i);
12: void puts (string s);
13: void endl ();
14: int getc ();
15: string getw ();
16: string getln ();
17: string[] getargv ();
18: void exit (int status);
```

```
1: // $Id: oclib.c,v 1.50 2015-05-20 19:57:55-07 - - $
2:
3: #include <assert.h>
4: #include <ctype.h>
5: #include <libgen.h>
6: #include <stdio.h>
7: #include <stdlib.h>
8: #include <string.h>
9:
10: #define __OCLIB_C__
11: #include "oclib.oh"
12:
13: char** oc_argv;
14:
15: void ____assert_fail (char* expr, char* file, int line) {
16:     fflush (NULL);
17:     fprintf (stderr, "%s: %s:%d: assert (%s) failed.\n",
18:             basename ((char*) oc_argv[0]), file, line, expr);
19:     fflush (NULL);
20:     abort();
21: }
22:
23: void* xcalloc (int nelem, int size) {
24:     void* result = calloc (nelem, size);
25:     assert (result != NULL);
26:     return result;
27: }
28:
29: void __ocmain (void);
30: int main (int argc, char** argv) {
31:     (void) argc; // warning: unused parameter 'argc'
32:     oc_argv = argv;
33:     __ocmain();
34:     return EXIT_SUCCESS;
35: }
36:
```

```
37:
38: char* scan (int (*skipover) (int), int (*stopat) (int)) {
39:     int byte;
40:     do {
41:         byte = getchar();
42:         if (byte == EOF) return NULL;
43:     } while (skipover (byte));
44:     char buffer[0x1000];
45:     char* end = buffer;
46:     do {
47:         *end++ = byte;
48:         assert (end < &buffer[sizeof buffer]);
49:         *end = '\0';
50:         byte = getchar();
51:     }while (byte != EOF && ! stopat (byte));
52:     char* result = (char*) strdup ((char*) buffer);
53:     assert (result != NULL);
54:     return result;
55: }
56:
57: int isfalse (int byte)    { return 0 & byte; }
58: int isnl (int byte)      { return byte == '\n'; }
59: void __putb (char byte)  { printf ("%s", byte ? "true" : "false"); }
60: void __putc (char byte)  { printf ("%c", byte); }
61: void __puti (int val)     { printf ("%d", val); }
62: void __puts (char* str)  { printf ("%s", str); }
63: void __endl (void)       { printf ("\n"); fflush (NULL); }
64: int __getc (void)        { return getchar(); }
65: char* __getw (void)       { return scan (isspace, isspace); }
66: char* __getln (void)     { return scan (isfalse, isnl); }
67: char** __getargv (void)  { return oc_argv; }
68: void __exit (int status) { exit (status); }
69:
```