```
// $Id: oclib.oh,v 1.33 2016-11-30 14:45:59-08 - - $

#ifndef __OCLIB_OH__
#define __OCLIB_OH__

#ifdef __OCLIB_C__
void* xcalloc (int nelem, int size);
void __putb (char __b);
void __putc (char __c);
void __puti (int __i);
void __puts (char* __s);
void __endl (void);
int __getc (void);
char* __getw (void);
char* __getln (void);
char** __getargv (void);
void __exit (int status);
void ____assert_fail (char* expr, char* file, int line);

#else
#define EOF (-1)
#define char int
#define bool int
#define true 1
#define false 0
#define assert(expr) \
        {if (!(expr)) __assert_fail (#expr, __FILE__, __LINE__);}
void __assert_fail (string expr, string file, int line);
void putb (int b);
void putc (char c);
void puti (int i);
void puts (string s);
void endl ();
int getc ();
string getw ();
string getln ();
string[] getargv ();
void exit (int status);

#endif
#endif

```

```
# 1 "oclib.oh"
# 1 "<built-in>"
# 1 "<command-line>"
# 1 "/usr/include/stdc-predef.h" 1 3 4
# 1 "<command-line>" 2
# 1 "oclib.oh"






void* xcalloc (int nelem, int size);
void __putb (char __b);
void __putc (char __c);
void __puti (int __i);
void __puts (char* __s);
void __endl (void);
int __getc (void);
char* __getw (void);
char* __getln (void);
char** __getargv (void);
void __exit (int status);
void ____assert_fail (char* expr, char* file, int line);
```

```
# 1 "oclib.oh"
# 1 "<built-in>"
# 1 "<command-line>"
# 1 "/usr/include/stdc-predef.h" 1 3 4
# 1 "<command-line>" 2
# 1 "oclib.oh"
# 28 "oclib.oh"
void __assert_fail (string expr, string file, int line);
void putb (int b);
void putc (int c);
void puti (int i);
void puts (string s);
void endl ();
int getc ();
string getw ();
string getln ();
string[] getargv ();
void exit (int status);
```

```c
// $Id: oclib.c,v 1.50 2015-05-20 19:57:55-07 - - $

#include <assert.h>
#include <ctype.h>
#include <libgen.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define __OCLIB_C__
#include "oclib.oh"

char** oc_argv;

void ____assert_fail (char* expr, char* file, int line) {
   fflush (NULL);
   fprintf (stderr, "%s: %s:%d: assert (%s) failed.\n",
            basename ((char*) oc_argv[0]), file, line, expr);
   fflush (NULL);
   abort();
}

void* xcalloc (int nelem, int size) {
   void* result = calloc (nelem, size);
   assert (result != NULL);
   return result;
}

void __ocmain (void);
int main (int argc, char** argv) {
   (void) argc; // warning: unused parameter 'argc'
   oc_argv = argv;
   __ocmain();
   return EXIT_SUCCESS;
}

```

```

char* scan (int (*skipover) (int), int (*stopat) (int)) {
   int byte;
   do {
      byte = getchar();
      if (byte == EOF) return NULL;
   } while (skipover (byte));
   char buffer[0x1000];
   char* end = buffer;
   do {
      *end++ = byte;
      assert (end < &buffer[sizeof buffer]);
      *end = '\0';
      byte = getchar();
   }while (byte != EOF && ! stopat (byte));
   char* result = (char*) strdup ((char*) buffer);
   assert (result != NULL);
   return result;
}

int isfalse (int byte)   { return 0 & byte; }
int isnl (int byte)      { return byte == '\n'; }
void __putb (char byte)  { printf ("%s", byte ? "true" : "false"); }
void __putc (char byte)  { printf ("%c", byte); }
void __puti (int val)    { printf ("%d", val); }
void __puts (char* str)  { printf ("%s", str); }
void __endl (void)       { printf ("\n"); fflush (NULL); }
int __getc (void)        { return getchar(); }
char* __getw (void)      { return scan (isspace, isspace); }
char* __getln (void)     { return scan (isfalse, isnl); }
char** __getargv (void)  { return oc_argv; }
void __exit (int status) { exit (status); }

```