```
// $Id: oclib.h,v 1.13 2019-09-19 17:08:34-07 - - $

// Bilingual file useable as a header file for both oc and g++.

#ifndef __OCLIB_H__
#define __OCLIB_H__

#ifdef __cplusplus
extern "C" {
using string = char*;
#endif

#define SUCCESS 0
#define FAILURE 1
#define BOOL int
#define TRUE 1
#define FALSE 0
#define EOF (-1)

#define assert(expr) {if (not (expr)) fail (#expr, __FILE__, __LINE__);}

void fail (string expr, string file, int line);

void putchr (int chr);
void putint (int num);
void putstr (string str);

int getchr();
string getstr();
string getln();

#ifdef __cplusplus
}
#endif

#endif

```

```
// $Id: octypes.h,v 1.4 2019-09-16 14:36:17-07 - - $

// Type definitiions to compile oc programs with g++.

#ifndef __OCDEFS_H__
#define __OCDEFS_H__

#include <type_traits>

using string = char*;

template <typename type>
using ptr = std::enable_if_t<std::is_class<type>::value,type*>;

template <typename type>
struct array {
   using array_value_type = type;
   type* data {};
   array() = default;
   array (type* that) { data = that; }
   array& operator= (type* that) { data = that; return *this; }
   type& operator[] (int i) { return data[i]; }
};

template <typename type>
std::enable_if_t<std::is_class<type>::value,ptr<type>>
alloc() {
   return new type();
}

template <typename type>
array<typename type::array_value_type>
alloc (int size) {
   auto result = new typename type::array_value_type [size] {};
   using result_t = array<typename type::array_value_type>*;
   return *reinterpret_cast<result_t> (&result);
}

template <typename type>
std::enable_if_t<std::is_same<type,string>::value,string>
alloc (int size) {
   return new char[size] {};
}

#endif

```

```c
// $Id: oclib.c,v 1.7 2019-09-19 17:08:34-07 - - $

#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define not !
#define nullptr 0
#define string char*

#include "oclib.h"

void fail (string expr, string file, int line) {
   fprintf (stderr, "%s:%d: assert (%s) failed\n", file, line, expr);
   abort();
}

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

void putchr (int chr) { printf ("%c", chr); }
void putint (int num) { printf ("%d", num); }
void putstr (string str) { printf ("%s", str); }

int getchr() { return getchar(); }

static char get_buffer[0x1000];

string getstr (void) {
   static char format[16];
   sprintf (format, "%%%zds", sizeof get_buffer - 1);
   int count = scanf (format, get_buffer);
   return count != 1 ? nullptr : strdup (get_buffer);
}

string getln (void) {
   string result = fgets (get_buffer, sizeof get_buffer, stdin);
   return result == nullptr ? nullptr : strdup (result);
}

```

```
0000000000000000 b .bss
0000000000000000 n .comment
0000000000000000 d .data
0000000000000000 r .eh_frame
0000000000000000 n .note.GNU-stack
0000000000000000 r .rodata.str1.1
0000000000000000 t .text
                 U _IO_getc
                 U __isoc99_scanf
                 U __strdup
                 U abort
                 U calloc
0000000000000000 T fail
                 U fgets
0000000000000000 b format.3364
                 U fprintf
0000000000000020 b get_buffer
0000000000000090 T getchr
00000000000000f0 T getln
00000000000000a0 T getstr
0000000000000000 a oclib.c
                 U printf
                 U putchar
0000000000000060 T putchr
0000000000000070 T putint
0000000000000080 T putstr
                 U sprintf
                 U stderr
                 U stdin
0000000000000030 T xcalloc
```

```
        .file   "oclib.c"
        .text
        .section        .rodata.str1.1,"aMS",@progbits,1
.LC0:
        .string "%s:%d: assert (%s) failed\n"
        .text
        .p2align 4,,15
        .globl  fail
        .type   fail, @function
fail:
.LFB32:
        .cfi_startproc
        subq    $8, %rsp
        .cfi_def_cfa_offset 16
        movq    %rdi, %r8
        movq    stderr(%rip), %rdi
        movl    %edx, %ecx
        xorl    %eax, %eax
        movq    %rsi, %rdx
        movl    $.LC0, %esi
        call    fprintf
        call    abort
        .cfi_endproc
.LFE32:
        .size   fail, .-fail
        .section        .rodata.str1.1
.LC1:
        .string "../oclib.c"
.LC2:
        .string "result != nullptr"
        .text
        .p2align 4,,15
        .globl  xcalloc
        .type   xcalloc, @function
xcalloc:
.LFB33:
        .cfi_startproc
        subq    $8, %rsp
        .cfi_def_cfa_offset 16
        movslq  %esi, %rsi
        movslq  %edi, %rdi
        call    calloc
        testq   %rax, %rax
        je      .L7
        addq    $8, %rsp
        .cfi_remember_state
        .cfi_def_cfa_offset 8
        ret
.L7:
        .cfi_restore_state
        movl    $20, %edx
        movl    $.LC1, %esi
        movl    $.LC2, %edi
        call    fail
        .cfi_endproc
.LFE33:
        .size   xcalloc, .-xcalloc
        .p2align 4,,15
        .globl  putchr
        .type   putchr, @function
putchr:
.LFB34:
        .cfi_startproc
        jmp     putchar
        .cfi_endproc
.LFE34:
        .size   putchr, .-putchr
        .section        .rodata.str1.1
.LC3:
        .string "%d"
        .text
        .p2align 4,,15
        .globl  putint
        .type   putint, @function
putint:
.LFB35:
        .cfi_startproc
        movl    %edi, %esi
        xorl    %eax, %eax
        movl    $.LC3, %edi
        jmp     printf
        .cfi_endproc
.LFE35:
        .size   putint, .-putint
        .section        .rodata.str1.1
.LC4:
        .string "%s"
        .text
        .p2align 4,,15
        .globl  putstr
        .type   putstr, @function
putstr:
.LFB36:
        .cfi_startproc
        movq    %rdi, %rsi
        xorl    %eax, %eax
        movl    $.LC4, %edi
        jmp     printf
        .cfi_endproc
.LFE36:
        .size   putstr, .-putstr
        .p2align 4,,15
        .globl  getchr
        .type   getchr, @function
getchr:
.LFB37:
        .cfi_startproc
        movq    stdin(%rip), %rdi
        jmp     _IO_getc
        .cfi_endproc
.LFE37:
        .size   getchr, .-getchr
        .section        .rodata.str1.1
.LC5:
        .string "%%%zds"
        .text
        .p2align 4,,15
        .globl  getstr
        .type   getstr, @function
getstr:
.LFB38:
        .cfi_startproc
        subq    $8, %rsp
        .cfi_def_cfa_offset 16
        movl    $.LC5, %esi
        movl    $format.3364, %edi
        xorl    %eax, %eax
        movl    $4095, %edx
        call    sprintf
        xorl    %eax, %eax
        movl    $get_buffer, %esi
        movl    $format.3364, %edi
        call    __isoc99_scanf
        cmpl    $1, %eax
        je      .L15
        xorl    %eax, %eax
        addq    $8, %rsp
        .cfi_remember_state
        .cfi_def_cfa_offset 8
        ret
        .p2align 4,,10
        .p2align 3
.L15:
        .cfi_restore_state
        movl    $get_buffer, %edi
        addq    $8, %rsp
        .cfi_def_cfa_offset 8
        jmp     __strdup
        .cfi_endproc
.LFE38:
        .size   getstr, .-getstr
        .p2align 4,,15
        .globl  getln
        .type   getln, @function
getln:
.LFB39:
        .cfi_startproc
        subq    $8, %rsp
        .cfi_def_cfa_offset 16
        movq    stdin(%rip), %rdx
        movl    $4096, %esi
        movl    $get_buffer, %edi
        call    fgets
        testq   %rax, %rax
        je      .L16
        movq    %rax, %rdi
        addq    $8, %rsp
        .cfi_remember_state
        .cfi_def_cfa_offset 8
        jmp     __strdup
        .p2align 4,,10
        .p2align 3
.L16:
        .cfi_restore_state
        xorl    %eax, %eax
        addq    $8, %rsp
        .cfi_def_cfa_offset 8
        ret
        .cfi_endproc
.LFE39:
        .size   getln, .-getln
        .local  format.3364
        .comm   format.3364,16,16
        .local  get_buffer
        .comm   get_buffer,4096,32
        .ident  "GCC: (GNU) 8.2.1 20180905 (Red Hat 8.2.1-3)"
        .section        .note.GNU-stack,"",@progbits
```