```
 1: // $Id: oclib.h,v 1.13 2019-09-19 17:08:34-07 - - $
 2:
 3: // Bilingual file useable as a header file for both oc and g++.
 4:
 5: #ifndef __OCLIB_H__
 6: #define __OCLIB_H__
 7:
 8: #ifdef __cplusplus
 9: extern "C" {
10: using string = char*;
11: #endif
12:
13: #define SUCCESS 0
14: #define FAILURE 1
15: #define BOOL int
16: #define TRUE 1
17: #define FALSE 0
18: #define EOF (-1)
19:
20: #define assert(expr) {if (not (expr)) fail (#expr, __FILE__, __LINE__);}
21:
22: void fail (string expr, string file, int line);
23:
24: void putchr (int chr);
25: void putint (int num);
26: void putstr (string str);
27:
28: int getchr();
29: string getstr();
30: string getln();
31:
32: #ifdef __cplusplus
33: }
34: #endif
35:
36: #endif
37:
```

```
 1: // $Id: octypes.h,v 1.4 2019-09-16 14:36:17-07 - - $
 2:
 3: // Type definitiions to compile oc programs with g++.
 4:
 5: #ifndef __OCDEFS_H__
 6: #define __OCDEFS_H__
 7:
 8: #include <type_traits>
 9:
10: using string = char*;
11:
12: template <typename type>
13: using ptr = std::enable_if_t<std::is_class<type>::value,type*>;
14:
15: template <typename type>
16: struct array {
17:    using array_value_type = type;
18:    type* data {};
19:    array() = default;
20:    array (type* that) { data = that; }
21:    array& operator= (type* that) { data = that; return *this; }
22:    type& operator[] (int i) { return data[i]; }
23: };
24:
25: template <typename type>
26: std::enable_if_t<std::is_class<type>::value,ptr<type>>
27: alloc() {
28:    return new type();
29: }
30:
31: template <typename type>
32: array<typename type::array_value_type>
33: alloc (int size) {
34:    auto result = new typename type::array_value_type [size] {};
35:    using result_t = array<typename type::array_value_type>*;
36:    return *reinterpret_cast<result_t> (&result);
37: }
38:
39: template <typename type>
40: std::enable_if_t<std::is_same<type,string>::value,string>
41: alloc (int size) {
42:    return new char[size] {};
43: }
44:
45: #endif
46:
```

```
 1: // $Id: oclib.c,v 1.7 2019-09-19 17:08:34-07 - - $
 2:
 3: #include <stdio.h>
 4: #include <stdlib.h>
 5: #include <string.h>
 6:
 7: #define not !
 8: #define nullptr 0
 9: #define string char*
10:
11: #include "oclib.h"
12:
13: void fail (string expr, string file, int line) {
14:    fprintf (stderr, "%s:%d: assert (%s) failed\n", file, line, expr);
15:    abort();
16: }
17:
18: void* xcalloc (int nelem, int size) {
19:    void* result = calloc (nelem, size);
20:    assert (result != nullptr);
21:    return result;
22: }
23:
24: void putchr (int chr) { printf ("%c", chr); }
25: void putint (int num) { printf ("%d", num); }
26: void putstr (string str) { printf ("%s", str); }
27:
28: int getchr() { return getchar(); }
29:
30: static char get_buffer[0x1000];
31:
32: string getstr (void) {
33:    static char format[16];
34:    sprintf (format, "%%%zds", sizeof get_buffer - 1);
35:    int count = scanf (format, get_buffer);
36:    return count != 1 ? nullptr : strdup (get_buffer);
37: }
38:
39: string getln (void) {
40:    string result = fgets (get_buffer, sizeof get_buffer, stdin);
41:    return result == nullptr ? nullptr : strdup (result);
42: }
43:
```

```
  1: 0000000000000000 b .bss
  2: 0000000000000000 n .comment
  3: 0000000000000000 d .data
  4: 0000000000000000 r .eh_frame
  5: 0000000000000000 n .note.GNU-stack
  6: 0000000000000000 r .rodata.str1.1
  7: 0000000000000000 t .text
  8:                  U _IO_getc
  9:                  U __isoc99_scanf
 10:                  U __strdup
 11:                  U abort
 12:                  U calloc
 13: 0000000000000000 T fail
 14:                  U fgets
 15: 0000000000000000 b format.3364
 16:                  U fprintf
 17: 0000000000000020 b get_buffer
 18: 0000000000000090 T getchr
 19: 00000000000000f0 T getln
 20: 00000000000000a0 T getstr
 21: 0000000000000000 a oclib.c
 22:                  U printf
 23:                  U putchar
 24: 0000000000000060 T putchr
 25: 0000000000000070 T putint
 26: 0000000000000080 T putstr
 27:                  U sprintf
 28:                  U stderr
 29:                  U stdin
 30: 0000000000000030 T xcalloc
```

```
  1:         .file   "oclib.c"
  2:         .text
  3:         .section        .rodata.str1.1,"aMS",@progbits,1
  4: .LC0:
  5:         .string "%s:%d: assert (%s) failed\n"
  6:         .text
  7:         .p2align 4,,15
  8:         .globl  fail
  9:         .type   fail, @function
 10: fail:
 11: .LFB32:
 12:         .cfi_startproc
 13:         subq    $8, %rsp
 14:         .cfi_def_cfa_offset 16
 15:         movq    %rdi, %r8
 16:         movq    stderr(%rip), %rdi
 17:         movl    %edx, %ecx
 18:         xorl    %eax, %eax
 19:         movq    %rsi, %rdx
 20:         movl    $.LC0, %esi
 21:         call    fprintf
 22:         call    abort
 23:         .cfi_endproc
 24: .LFE32:
 25:         .size   fail, .-fail
 26:         .section        .rodata.str1.1
 27: .LC1:
 28:         .string "../oclib.c"
 29: .LC2:
 30:         .string "result != nullptr"
 31:         .text
 32:         .p2align 4,,15
 33:         .globl  xcalloc
 34:         .type   xcalloc, @function
 35: xcalloc:
 36: .LFB33:
 37:         .cfi_startproc
 38:         subq    $8, %rsp
 39:         .cfi_def_cfa_offset 16
 40:         movslq  %esi, %rsi
 41:         movslq  %edi, %rdi
 42:         call    calloc
 43:         testq   %rax, %rax
 44:         je      .L7
 45:         addq    $8, %rsp
 46:         .cfi_remember_state
 47:         .cfi_def_cfa_offset 8
 48:         ret
 49: .L7:
 50:         .cfi_restore_state
 51:         movl    $20, %edx
 52:         movl    $.LC1, %esi
 53:         movl    $.LC2, %edi
 54:         call    fail
 55:         .cfi_endproc
 56: .LFE33:
 57:         .size   xcalloc, .-xcalloc
 58:         .p2align 4,,15
```

```
  59:         .globl  putchr
  60:         .type   putchr, @function
  61: putchr:
  62: .LFB34:
  63:         .cfi_startproc
  64:         jmp     putchar
  65:         .cfi_endproc
  66: .LFE34:
  67:         .size   putchr, .-putchr
  68:         .section        .rodata.str1.1
  69: .LC3:
  70:         .string "%d"
  71:         .text
  72:         .p2align 4,,15
  73:         .globl  putint
  74:         .type   putint, @function
  75: putint:
  76: .LFB35:
  77:         .cfi_startproc
  78:         movl    %edi, %esi
  79:         xorl    %eax, %eax
  80:         movl    $.LC3, %edi
  81:         jmp     printf
  82:         .cfi_endproc
  83: .LFE35:
  84:         .size   putint, .-putint
  85:         .section        .rodata.str1.1
  86: .LC4:
  87:         .string "%s"
  88:         .text
  89:         .p2align 4,,15
  90:         .globl  putstr
  91:         .type   putstr, @function
  92: putstr:
  93: .LFB36:
  94:         .cfi_startproc
  95:         movq    %rdi, %rsi
  96:         xorl    %eax, %eax
  97:         movl    $.LC4, %edi
  98:         jmp     printf
  99:         .cfi_endproc
 100: .LFE36:
 101:         .size   putstr, .-putstr
 102:         .p2align 4,,15
 103:         .globl  getchr
 104:         .type   getchr, @function
 105: getchr:
 106: .LFB37:
 107:         .cfi_startproc
 108:         movq    stdin(%rip), %rdi
 109:         jmp     _IO_getc
 110:         .cfi_endproc
 111: .LFE37:
 112:         .size   getchr, .-getchr
 113:         .section        .rodata.str1.1
 114: .LC5:
 115:         .string "%%%zds"
 116:         .text
```

```
117:          .p2align 4,,15
118:          .globl  getstr
119:          .type   getstr, @function
120: getstr:
121: .LFB38:
122:          .cfi_startproc
123:          subq    $8, %rsp
124:          .cfi_def_cfa_offset 16
125:          movl    $.LC5, %esi
126:          movl    $format.3364, %edi
127:          xorl    %eax, %eax
128:          movl    $4095, %edx
129:          call    sprintf
130:          xorl    %eax, %eax
131:          movl    $get_buffer, %esi
132:          movl    $format.3364, %edi
133:          call    __isoc99_scanf
134:          cmpl    $1, %eax
135:          je      .L15
136:          xorl    %eax, %eax
137:          addq    $8, %rsp
138:          .cfi_remember_state
139:          .cfi_def_cfa_offset 8
140:          ret
141:          .p2align 4,,10
142:          .p2align 3
143: .L15:
144:          .cfi_restore_state
145:          movl    $get_buffer, %edi
146:          addq    $8, %rsp
147:          .cfi_def_cfa_offset 8
148:          jmp     __strdup
149:          .cfi_endproc
150: .LFE38:
151:          .size   getstr, .-getstr
152:          .p2align 4,,15
153:          .globl  getln
154:          .type   getln, @function
155: getln:
156: .LFB39:
157:          .cfi_startproc
158:          subq    $8, %rsp
159:          .cfi_def_cfa_offset 16
160:          movq    stdin(%rip), %rdx
161:          movl    $4096, %esi
162:          movl    $get_buffer, %edi
163:          call    fgets
164:          testq   %rax, %rax
165:          je      .L16
166:          movq    %rax, %rdi
167:          addq    $8, %rsp
168:          .cfi_remember_state
169:          .cfi_def_cfa_offset 8
170:          jmp     __strdup
171:          .p2align 4,,10
172:          .p2align 3
173: .L16:
174:          .cfi_restore_state
```

```
175:          xorl    %eax, %eax
176:          addq    $8, %rsp
177:          .cfi_def_cfa_offset 8
178:          ret
179:          .cfi_endproc
180: .LFE39:
181:          .size   getln, .-getln
182:          .local  format.3364
183:          .comm   format.3364,16,16
184:          .local  get_buffer
185:          .comm   get_buffer,4096,32
186:          .ident  "GCC: (GNU) 8.2.1 20180905 (Red Hat 8.2.1-3)"
187:          .section        .note.GNU-stack,"",@progbits
```