```
CS3650: select, synchronous I/O multiplexing

1. select interfaces

  a) select

    * int select(int nfds, fd_set *restrict readfds,
                 fd_set *restrict writefds, fd_set *restrict errorfds,
                 struct timeval *restrict timeout);

      select() examines the I/O descriptor sets whose addresses are passed in
      readfds, writefds, and errorfds to see if some of their descriptors are
      ready for reading, are ready for writing, or have an exceptional
      condition pending, respectively.

  b) fd_set manipulation

    * FD_ZERO(fd_set *set);             Clear all entries from the set.
    * FD_SET(int fd, fd_set *set);      Add fd to the set.
    * FD_CLR(int fd, fd_set *set);      Remove fd from the set.
    * FD_ISSET(int fd, fd_set *set);    Return true if fd is in the set.


2. An example - a chat server

    // Below is a code snippet using select()

    int fds[2] = {0, 0};
    fds[0] = ...                    // socket connection 1
    fds[1] = ...                    // socket connection 2

    fd_set readfds;

    while(1) {
        FD_ZERO(&readfds);
        for (int i=0; i<2; i++) {
            FD_SET(fds[i], &readfds);
        }

        int maxfd = ...             // Q: what is the maxfd?

        select(maxfd+1, &readfds, NULL, NULL, NULL);

        for (int i=0; i<2; i++) {
            if (FD_ISSET(fds[i], &readfds)) {
                print(fds[i], ...); // print msg received
            }
        }
    }
    ... // wrap up and exit
```



```
CS3650: I/O and device driver

1. An example of I/O instructions:
   Setting the cursor position

  The code below is excerpted from WeensyOS's k-hardware.c. It
  uses I/O instructions to set a blinking cursor in the upper left of
  the screen.

  // console_show_cursor(cpos)
  //   Move the console cursor to position 'cpos',
  //   which should be between 0 and 80 * 25.

  void console_show_cursor(int cpos) {
    if (cpos < 0 || cpos > CONSOLE_ROWS * CONSOLE_COLUMNS)
      cpos = 0;

    outb(0x3D4, 14); // Command 14 = upper byte of position
    outb(0x3D5, cpos / 256); // row
    outb(0x3D4, 15); // Command 15 = lower byte of position
    outb(0x3D5, cpos % 256); // column

  }
```

```
27
28  2. Memory-mapped I/O
29
30    a. Here is a 32-bit PC's physical memory map:
31
32        +------------------+ <- 0xFFFFFFFF (4GB)
33        |      32-bit      |
34        |  memory mapped   |
35        |     devices      |
36        |                  |
37        /\/\/\/\/\/\/\/\/\/\
38
39        /\/\/\/\/\/\/\/\/\/\
40        |                  |
41        |      Unused      |
42        |                  |
43        +------------------+ <- depends on amount of RAM
44        |                  |
45        |                  |
46        |  Extended Memory |
47        |                  |
48        |                  |
49        +------------------+ <- 0x00100000 (1MB)
50        |     BIOS ROM     |
51        +------------------+ <- 0x000F0000 (960KB)
52        |  16-bit devices, |
53        |  expansion ROMs  |
54        +------------------+ <- 0x000C0000 (768KB)
55        |   VGA Display    |
56        +------------------+ <- 0x000A0000 (640KB)
57        |                  |
58        |    Low Memory    |
59        |                  |
60        +------------------+ <- 0x00000000
61
62     [Credit to Frans Kaashoek, Robert Morris, and
63     Nickolai Zeldovich for this picture]
64
```



```
65
66     b. Loads and stores to the device memory "go to hardware".
67
68       An example is in the console printing code from WeensyOS.
69       Here is an excerpt from link/shared.ld:
70
71       /* Compare the address below to the map above. */
72       PROVIDE(console = 0xB8000);
73
74       This is an excerpt from lib.c; notice how it uses the address
75       "console":
76
77       /*
78       * prints a character to the console at the specified
79       * cursor position in the specified color.
80       * Question: what is going on in the check
81       * if (c == '\n')
82       * ?
83       * Hint: '\n' is "C" for "newline" (the user pressed enter).
84       */
85       static void console_putc(printer* p, unsigned char c, int color) {
86         console_printer* cp = (console_printer*) p;
87         if (cp->cursor >= console + CONSOLE_ROWS * CONSOLE_COLUMNS) {
88           cp->cursor = console;
89         }
90         if (c == '\n') {
91           int pos = (cp->cursor - console) % 80;
92           for (; pos != 80; pos++) {
93             *cp->cursor++ = ' ' | color;
94           }
95         } else {
96           *cp->cursor++ = c | color;
97         }
98       }
```