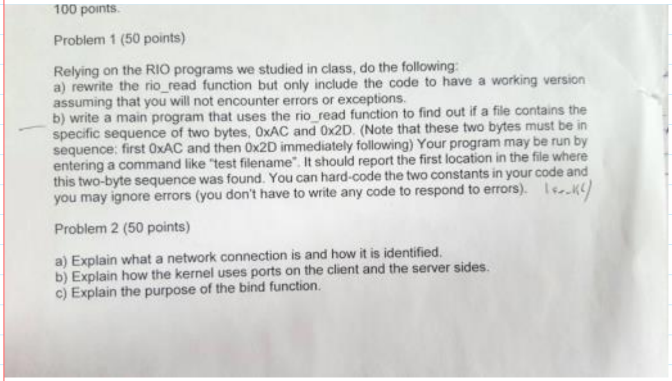
Sunday, February 12, 2017 4:01 PM



100 points.

Problem 1 (50 points)

Relying on the RIO programs we studied in class, do the following:
a) rewrite the rio_read function but only include the code to have a working version assuming that you will not encounter errors or exceptions.
b) write a main program that uses the rio_read function to find out if a file contains the specific sequence of two bytes, 0xAC and 0x2D. (Note that these two bytes must be in sequence: first 0xAC and then 0x2D immediately following) Your program may be run by entering a command like "test filename". It should report the first location in the file where this two-byte sequence was found. You can hard-code the two constants in your code and you may ignore errors (you don't have to write any code to respond to errors).

Problem 2 (50 points)

a) Explain what a network connection is and how it is identified.
b) Explain how the kernel uses ports on the client and the server sides.
c) Explain the purpose of the bind function.

① a)
```
ssize_t rio_read (rio_t * rio, void * buf, size_t n) {
    size_t nleft = n;
    size_t nread;
    char * buf_ptr = buf;

    while (rio -> rio_cnt <= 0) {
        if ((rio->rio_cnt = read (rio -> rio_fp, rio -> rio_buf, sizeof (rio -> rio_buf)) < 0) {

            //errors
        }
        else if (nread == 0) {
            return 0;
        }
        else {
          rio -> rio_bufptr = rio -> rio_buf;
        }
    }

    size_t num = n;
    if (rio -> rio_cnt < num) {
       num = rio -> rio_cnt;
    }
    memcpy (buf_ptr, rio -> rio_bufptr; num);
    rio -> rio_bufptr += num;
    rio -> rio_cnt -= num;

    return num
}
```

b)
```
int main (int argc, char ** argv) {

    if (argc < 2) {
       exit(0);

    char * filename;
    filename = argv[1];

    rio_t rio;
    int fp;
    fp = open (filename, O_rdonly);

    rio_readinitb (&rio, fp);

    int foundFirst = 0, foundSequence = 0;
    char buf [2];
int location = 0;
int n = 0;
while ( rio_read ( &fp, buf, 2) > 0){
  if (foundFirst == 0){
     if (*buf == '0xAC'){
        foundFirst = 1;
        location = n;
     }
  }
  else {
     if (*buf == '0x2D'){ ← foundSequence = 1;
        break; // found sequence
     }
     else{
        foundFirst = 0;
        location = 0;
     }
  }
  n++;
}

if (found Sequence == 1){

  printf ("The sequence was found at byte %d in file.", location);

}
else {
  printf ( "The sequence was not found in the file.");

}

  exit (0);
}
```

2

a) and b) A network connection is a reliable, full duplex stream that two processes can use for communication. This means that information can be sent both ways, and the order in which messages are sent is preserved. A connection is identified by the socket addresses of each of the processes. A socket address is identified by an IP address which signifies the identity of the machine and a port that identifies the service that is being connected to. A port can either be ephemeral which is randomly assigned by a Kernel for the client machine to get responses or a well-known port for a server that a client knows is used for a specific service (e.g port 80 is usually used for web servers). Therefore, the connection is identified by (client IP, client port : server IP, server port).

c) The bind function is used to associate a given address structure with a socket that has been created. Unlike connect (for clients), which issues a connection request for a given address structure, the bind function only tells the socket the structure of the address (IP of the server and port #). The server then can call the listen function to tell the socket that it is a server socket and should expect incoming connections instead of a client socket.