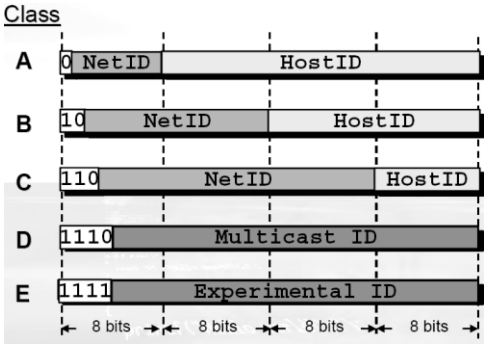
CpSc 360, Section 2, Fall 2008, Quiz #2

Name:_____ Clemson ID:_____

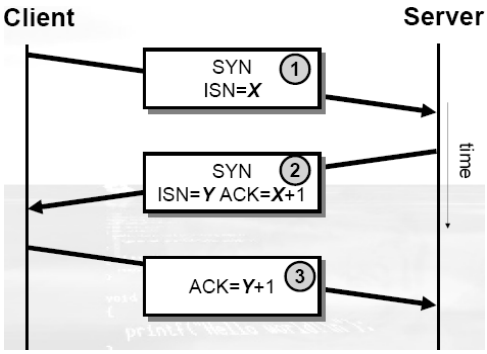
Question #1 (20 points):
Are the following statements **true** or **false**?

1. Every Ethernet interface has a unique 48 bit address. (TRUE)
2. IP provides connectionless, unreliable delivery of IP datagrams. (TRUE)
3. TCP initial sequence number starts at 1. (FALSE)
4. socket() allocates resources needed for a communication endpoint. (TRUE)
5. bind() will fill the local and remote IP addresses of the file descriptor associated with the specific socket. (FALSE)
6. TCP client uses connect() to send its IP address and port to the server. (TRUE)
7. TCP server uses accept() fill the remote IP address and port fields of the file descriptor structure opened by socket(). (FALSE)
8. TCP uses send and receive buffers to accomplish flow control. (TRUE)

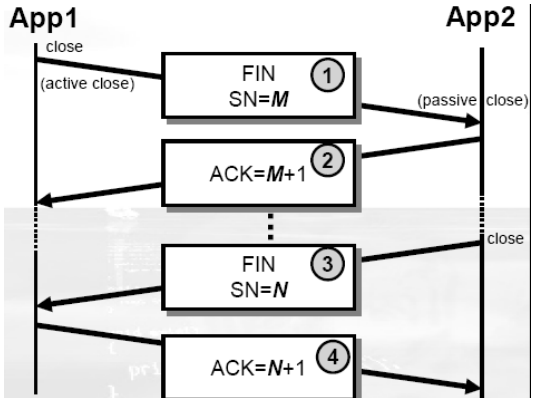
Question #2 (20 points):
Draw a diagram to show the 5 classes of IP addresses.



Question #3 (20 points):
Draw a diagram to describe the TCP 3-way handshake.



Question #4 (20 points):
Use a diagram to explain the normal close of a TCP connection.



Question #5 (20 points):
There are a few serious problems with the following TCP server program. Please point them out.

```
#include <stdio.h>     /* standard C i/o facilities */
#include <unistd.h>    /* Unix System Calls */
#include <sys/types.h> /* system data type definitions */
#include <sys/socket.h> /* socket specific definitions */
#include <netinet/in.h> /* INET constants and stuff */
#include <arpa/inet.h> /* IP address conversion stuff */

int main() {
  int sock_fd, sd, addrlen,length, n;
  struct sockaddr_in skaddr,  from;
  char buff[100];

  if ((sock_fd = socket( PF_INET, SOCK_STREAM, 0 )) < 0) {
    perror("Problem creating socket\n");
    exit(1);
  }

  skaddr.sin_family = AF_INET;
  skaddr.sin_addr.s_addr = htonl(INADDR_ANY);
  skaddr.sin_port = htons(0);

  if (bind(sock_fd, (struct sockaddr *) &skaddr, sizeof(skaddr))<0) {
    exit(0);
  }

  if (listen(sock_fd,5) < 0 ) {
    exit(1);
  }

  while (1) {

    if ( (sd = accept( sock_fd, (struct sockaddr*) &from, &addrlen)) < 0) {
      exit(1);
    }

    while ((n=read(sock_fd,buff,100))>0) {
      write(1,buff,n);
    }

    close(sock_fd);
  }
}
```

Missing a statement:
addrlen = sizeof(from);

"sd" should be used here, not "sock_fd".