```
#include <stdlib.h>
#include <string.h>
#include <sys/socket.h> //for socket(), connect(), ...
#include <unistd.h>     //for close()
#include <netinet/in.h> //for internet address family
#include <arpa/inet.h>  //for sockaddr_in, inet_addr()
#include <errno.h>      //for system error numbers
#include <iostream>     //for cin, cout...



////////////////////////////
//Stanley Lim 50441072
//computer networks class
// UDP program assignment 1
// this will relay messages between the client and server
//should work for the most part, it worked on my end
////////////////////////////


#define MY_PORT_ID   11073       /* a number > 1024 */
#define SERVER_PORT_ID   11072
#define SERV_HOST_ADDR "127.0.0.1"

#define MESSAGESIZE  80

using std::cout;
using std::cerr;

main( int argc, char *argv[])
{
  int sockfd;  //socket declaration
  int serverSockfd;
  struct sockaddr_in my_addr, client_addr;  //addresses
  char serv_host_addr[20]=SERV_HOST_ADDR;
  struct sockaddr_in server_addr, relay_addr;
  int nread,        //number of bytes read
     retcode,     //returned code from a function
     addrlen;     //address length

  char msg[MESSAGESIZE];

  int my_port_id = MY_PORT_ID;  //default port number
  int server_port_id = SERVER_PORT_ID;
  if (argc == 2)   // if command line with port number, take it
  {
     my_port_id = atoi(argv[1]);
  }
```

```
// ------------------------------------------------------
// Initialization:
// ------------------------------------------------------
//cout << "Server: creating socket\n";
if ( (sockfd = socket(AF_INET, SOCK_DGRAM, 0)) < 0)
{
   cerr << "Server: socket error: " << errno << "\n";
   exit(1);
}

// cout << "Server: binding my local socket\n";
memset( &my_addr, 0, sizeof(my_addr));    // Zero out structure
my_addr.sin_family = AF_INET;        // Internet address family
my_addr.sin_port = htons(my_port_id);    //My port
my_addr.sin_addr.s_addr = htonl(INADDR_ANY);    // Any incoming interface


//initialize server socket
 if ( (serverSockfd = socket(AF_INET, SOCK_DGRAM, 0)) < 0)
{
   cerr << "Client: socket failed: "<< errno <<"\n";
   exit(1);
}


// cout << "client: binding my local socket\n";  // this is optional
memset( &server_addr, 0, sizeof(server_addr));       // Zero out structure
server_addr.sin_family = AF_INET;                // Internet address family
server_addr.sin_port = htons(0);                 // My port, 0 means any random available one
server_addr.sin_addr.s_addr = htonl(INADDR_ANY); // Any incoming interface

// ------------------------------------------------------
// binding:
// ------------------------------------------------------
if ( (bind(sockfd, (struct sockaddr *) &my_addr, sizeof(my_addr)) < 0) )
{
   cerr << "Server: bind fail: " << errno << "\n";
   exit(2);
}


//binding server socket
if ( (bind(serverSockfd, (struct sockaddr *) &server_addr, sizeof(server_addr)) < 0) )
{
   cerr << "client: bind fail: " << errno << "\n";
   exit(2);
}
memset( &server_addr, 0, sizeof(server_addr));
server_addr.sin_family = AF_INET;
```

```
server_addr.sin_port = htons(server_port_id);

if (inet_aton(serv_host_addr, &(server_addr.sin_addr))==0) // get server addr
{  // invalid server address
   cerr << "Client: Invalid server address...\n";
   exit(2);
}

while (1)
{
   // ---------------------------------------------------------
   // Wait for client's connection
   // ---------------------------------------------------------
   cout << "\n\nWaiting for client's message....\n\n";
   addrlen = sizeof(client_addr);   // need to give it the buffer size for sender's address
   nread = recvfrom(sockfd,msg, MESSAGESIZE, 0,
            (struct sockaddr *) &client_addr, (socklen_t *) &addrlen);

   if (nread >0)
   {
      cout << "Receive from client (IP:" << inet_ntoa(client_addr.sin_addr)
          << " port:" << ntohs(client_addr.sin_port)<<")\n>>";
      cout.write(msg, nread);
      cout << "\n";

      // ---------------------------------------------------------
      // prepare a message and send to client
      // ---------------------------------------------------------
      // send a message back to client
      // strcpy(msg, "Got it!!!");
      // cout << "Please type a message back to client:\n";
      // nread = read(0, msg, MESSAGESIZE);    // read from keyboard
      // msg[nread]='\0';
      retcode = sendto(serverSockfd,msg,strlen(msg)+1,0,
               (struct sockaddr *) &server_addr, sizeof(server_addr));
      if (retcode <= -1)
      {
         cerr << "client: sendto failed: " << errno << "\n";
         exit(3);
      }

   }

   //receive from server
   nread = recvfrom(serverSockfd,msg, MESSAGESIZE, 0,
            (struct sockaddr *) &server_addr, (socklen_t *) &addrlen);
  if(nread > 0){

    cout << "Receive from server (IP:" << inet_ntoa(client_addr.sin_addr)
```

```
            << " port:" << ntohs(client_addr.sin_port)<<")\n>>";
        cout.write(msg, nread);
        cout << "\n";
        cout << "sending to client from relay";
        cout << "\n";

        //send to client
        retcode = sendto(sockfd,msg,strlen(msg)+1,0,
                (struct sockaddr *) &client_addr, sizeof(client_addr));
        if (retcode <= -1)
        {
          cerr << "client: sendto failed: " << errno << "\n";
          exit(3);
        }

      }


    }
    // -------------------------------------------------------
    // Termination
    // -------------------------------------------------------
    close(sockfd);
    close(serverSockfd);
}
```