```
// ***** udpserver.cpp *****

//****************************************************************
//
// Computer Science 4/5313 Computer Networks
//
// Spring  2016
//
// Instructor:  Hung-Chi Su
//
// Assignment # n
//
// Programmer:  your name
//
// Due Date:    day-of-week, month day, year
//
// Description: This is a UDP socket program that illustrates how a client
//             (udpclient.cpp) talks to a UDP server and sends a message
//             to the server and receives a response.
//
// Editor/Platform:  vi/Linux
//
//
//
//  Input:      none/or user input
//
//  Output:     The client will display the message replied by this server
//             The server will display the message sent by a client
//
//  Compile:    g++ -o udpserver udpserver.cpp
//
//  Command:    ./udpserver [<port#>]
//
//  Note:       Change the last 4 digits of  MY_PORT_ID to
//             the last 4 digits of your student id
//
//****************************************************************

                         // header files

//#include <stdio.h>    //for printf(), ...
#include <stdlib.h>
#include <string.h>
//#include <sys/types.h>        //for data types
#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...

// !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
// change the last 4 digits of MY_PORT_ID to your last 4 digits of student ID
// !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
#define MY_PORT_ID   11072       /* a number > 1024 */

#define MESSAGESIZE  80

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

main( int argc, char *argv[])
{
  int sockfd;  //socket declaration
  struct sockaddr_in my_addr, client_addr;  //addresses
  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
  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

  // ------------------------------------------------------
  // binding:
  // ------------------------------------------------------
  if ( (bind(sockfd, (struct sockaddr *) &my_addr, sizeof(my_addr)) < 0) )
  {
      cerr << "Server: bind fail: " << errno << "\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 relay (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(sockfd,msg,strlen(msg)+1,0,
                    (struct sockaddr *) &client_addr, sizeof(client_addr));

          cout << "Sending to relay\n";
          if (retcode <= -1)
          {
              cerr << "client: sendto failed: " << errno << "\n";
              exit(3);
          }
      }
  }
  // ------------------------------------------------------
  // Termination
  // ------------------------------------------------------
  close(sockfd);
}
```