#include "unp.h"

using namespace std;

int main()

{

/\* ---------- INITIALIZING VARIABLES ---------- \*/

int client, server;

int portNum = 1500;

bool isExit = false;

int bufsize = 1024;

char buffer[bufsize];

struct sockaddr\_in server\_addr;

socklen\_t size;

/\* ---------- ESTABLISHING SOCKET CONNECTION ----------\*/

/\* --------------- socket() function ------------------\*/

client = socket(AF\_INET, SOCK\_STREAM, 0);

if (client < 0)

{

cout << "\nError establishing socket..." << endl;

exit(1);

}

cout << "\n=> Socket server has been created..." << endl;

server\_addr.sin\_family = AF\_INET;

server\_addr.sin\_addr.s\_addr = htons(INADDR\_ANY);

server\_addr.sin\_port = htons(portNum);

/\* ---------- BINDING THE SOCKET ---------- \*/

/\* ---------------- bind() ---------------- \*/

if ((bind(client, (struct sockaddr\*)&server\_addr,sizeof(server\_addr))) < 0)

{

cout << "=> Error binding connection, the socket has already been established..." << endl;

return -1;

}

size = sizeof(server\_addr);

cout << "=> Looking for clients..." << endl;

/\* ------------- LISTENING CALL ------------- \*/

/\* ---------------- listen() ---------------- \*/

listen(client, 1);

/\* ------------- ACCEPTING CLIENTS ------------- \*/

/\* ----------------- listen() ------------------- \*/

int clientCount = 1;

server = accept(client,(struct sockaddr \*)&server\_addr,&size);

// first check if it is valid or not

if (server < 0)

cout << "=> Error on accepting..." << endl;

while (server > 0)

{

strcpy(buffer, "=> Server connected...\n");

write(server, buffer, bufsize);

cout << "=> Connected with the client #" << clientCount << ", you are good to go..." << endl;

cout << "\n=> Enter # to end the connection\n" << endl;u

do {

cout << "Client: ";

do {

read(server, buffer, bufsize);

cout << buffer << " ";

if (\*buffer == '#') {

return 0;

}

} while (\*buffer != '\*');

cout << "\nServer: ";

do {

cin >> buffer;

write(server, buffer, bufsize);

if (\*buffer == '#') {

return 0;

}

} while (\*buffer != '\*');

} while (!isExit);

/\* ---------------- CLOSE CALL ------------- \*/

/\* ----------------- close() --------------- \*/

// inet\_ntoa converts packet data to IP, which was taken from client

cout << "\n\n=> Connection terminated with IP " << inet\_ntoa(server\_addr.sin\_addr);

close(server);

cout << "\nGoodbye..." << endl;

isExit = false;

exit(1);

}

close(client);

return 0;

}