#include <ws2tcpip.h>

#include <iostream>

#include <conio.h>

#include <algorithm>

using namespace std;

#define BUFLEN 4096

#define PORT "27015"

int main ()

{

WSADATA wsaData;

SOCKET LS = INVALID\_SOCKET, CS = INVALID\_SOCKET;

struct addrinfo \*result = NULL, hints;

int iR, iSR;

string recvbuf;

char buf[BUFLEN];

iR = WSAStartup(MAKEWORD(2, 2), &wsaData);

if (iR != 0)

{

cout<<"ERROR"<<endl;

return 1;

}

ZeroMemory(&hints, sizeof(hints));

hints.ai\_family = AF\_INET;

hints.ai\_socktype = SOCK\_STREAM;

hints.ai\_protocol = IPPROTO\_TCP;

hints.ai\_flags = AI\_PASSIVE;

iR = getaddrinfo(NULL, PORT, &hints, &result);

if (iR != 0)

{

cout<<"ERROR"<<endl;

WSACleanup();

return 1;

}

LS = socket(result->ai\_family, result->ai\_socktype, result->ai\_protocol);

if (LS == INVALID\_SOCKET)

{

cout<<"ERROR "<<WSAGetLastError()<<endl;

closesocket(LS);

WSACleanup();

return 1;

}

iR = bind(LS, result->ai\_addr, (int)result->ai\_addrlen);

if (iR == SOCKET\_ERROR)

{

cout<<"ERROR "<<WSAGetLastError()<<endl;

freeaddrinfo(result);

closesocket(LS);

WSACleanup();

return 1;

}

freeaddrinfo(result);

iR = listen(LS, SOMAXCONN);

if (iR == SOCKET\_ERROR)

{

cout<<"ERROR "<<WSAGetLastError()<<endl;

closesocket(LS);

WSACleanup();

return 1;

}

CS = accept(LS, NULL, NULL);

if (CS == INVALID\_SOCKET)

{

cout<<"ERROR "<<WSAGetLastError()<<endl;

closesocket(LS);

WSACleanup();

return 1;

}

closesocket(LS);

ZeroMemory(&buf, sizeof(buf));

iR = recv(CS, buf, BUFLEN, 0);

if (iR > 0)

{

cout<<"Received Bytes : "<<iR<<endl;

recvbuf = string(buf);

cout<<recvbuf<<endl;

reverse(recvbuf.begin(), recvbuf.end());

cout<<recvbuf<<endl;

iSR = send(CS, recvbuf.c\_str(), iR, 0);

if (iSR == SOCKET\_ERROR)

{

cout<<"ERROR "<<WSAGetLastError()<<endl;

closesocket(CS);

WSACleanup();

return 1;

}

cout<<"Sent Bytes : "<<iSR<<endl;

}

else if (iR == 0)

{

cout<<"Closing connection..."<<endl;

}

else

{

cout<<"ERROR "<<WSAGetLastError()<<endl;

closesocket(CS);

WSACleanup();

return 1;

}

/\*iR = shutdown(CS, SD\_SEND);

if (iR == SOCKET\_ERROR)

{

cout<<"ERROR "<<WSAGetLastError()<<endl;

closesocket(CS);

WSACleanup();

return 1;

}\*/

//getch();

closesocket(CS);

WSACleanup();

getch();

return 0;

}