```
#include <iostream>
#include <string>
#include <WS2tcpip.h>
#pragma comment(lib, "ws2_32.lib");
using namespace std;

void main() {
	//initialize winsock
	WSADATA wsData;
	WORD ver = MAKEWORD(2, 2);
	int wsOK = WSAStartup(ver, &wsData);
	if (wsOK != 0) {
		cerr << "Can't Initialize winsock! Quitting" << endl;
		WSACleanup();
		return;
	}
	//Create a socket
	SOCKET listening = socket(AF_INET, SOCK_STREAM, 0);
	if (listening == INVALID_SOCKET) {//-1
		cerr << "Can't create a socket" << endl;
	}
	//Bind the ip address and port to a socket

	sockaddr_in hint;
	hint.sin_family = AF_INET;
	hint.sin_port = htons(54000);

	//inet_pton(AF_INET, "127.0.0.1", &hint.sin_addr);
	hint.sin_addr.s_addr = INADDR_ANY;  //inet_pton ... also work
	/*
 Binding listening socket to the special IP address INADDR_ANY allows your program
 to work without knowing the IP address of the machine it was running on, or,
 in the case of a machine with multiple network interfaces, it allowed your server
 to receive packets destined to any of the interfaces.
	*/

	bind(listening, (sockaddr*)&hint, sizeof(hint));

	//Tell winsock the socket is for listening
	listen(listening, SOMAXCONN);
	//The maximum length of the queue of pending connections.

	//wait for a connection
	sockaddr_in client;
	int clientsize = sizeof(client);
	SOCKET clientSocket = accept(listening, (sockaddr*)&client, &clientsize);//actual socket
for communicaitons
	//if (clientSocket == INVALID_SOCKET) { ....; return; }
```

```
	char host[NI_MAXHOST]; //client's remote name
	char service[NI_MAXSERV]; //service (i.e., port) the client is connected on
	ZeroMemory(host, NI_MAXHOST);
	ZeroMemory(service, NI_MAXSERV);
	if (getnameinfo((sockaddr*)&client, sizeof(client), host, NI_MAXHOST, service,
NI_MAXSERV, 0) == 0) {
		cout << host << "connected on port " << service << endl;
	}
	else {
		inet_ntop(AF_INET, &client.sin_addr, host, NI_MAXHOST);
		//opposite to inet_pton
		cout << host << " connected on port " <<
			ntohs(client.sin_port) << endl;//opposite to htons
	}
	//close listening socket
	closesocket(listening);
	//while loop; accept and echo message back to client
	char buf[4096];
	while (true) {
		ZeroMemory(buf, 4096);
		//wait for client to send data
		int bytesReceived = recv(clientSocket, buf, 4096, 0);
		if (bytesReceived == SOCKET_ERROR) {
			cerr << "Error in recv(). Quitting" << endl;
			break;
		}
		if (bytesReceived == 0) {
			cout << "client disconnected " << endl;
			break;
		}
		send(clientSocket, buf, bytesReceived + 1, 0);
	}
	//close the socket
	closesocket(clientSocket);
	//cleanup winsock
	WSACleanup();
}
```