**Project 3**

**Due Sunday 2/21/2016 by 11:59 PM**

**No delay is accepted**

**For each day late submission 2 points penalty is incurred.**

**Copy/paste your Server classes and a client class in this document and drop it into the folder *Project 3 on the blackboard.***

**If you have any question please email me at mbadii@pace.edu**

*Do not delete anything from this file. Just write your answer under the word: Answer.*

Write Multicast client-server classes to do the following:

There are number of clients. Each one send a integer to the server.

The server adds these integers.

When the server receives a zero from a client it displays the sum of the integers and terminates.

For example if the first client sends 10, the second client sends 20, and the third client sends zero the server displays 30 and terminates.

To test your project make two folders one for the server and one for one client. Run the server first. Run the client several times and each time change the integer that the client is sending. In the last run of the client send zero.

**Answer**:

**The server class**:

import java.io.\*;

import java.net.\*;

import java.util.ArrayList;

import java.util.List;

import java.util.Scanner;

public class MyServer {

static DatagramSocket clientSocket ;

static DatagramPacket datagram;

static byte [] buff ;

public static void main(String[] args) throws IOException {

int receivedNumber = 0;

boolean flag = false;

String receivedMessage = "";

int sum =0;

try {

final int MAX\_LEN = 100;

InetAddress host = InetAddress.getByName("localhost");

clientSocket = new DatagramSocket(6456,host);

while(receivedNumber != 0 || flag == false)

{

flag = true;

byte [] incomingData = new byte[MAX\_LEN];

datagram = new DatagramPacket(incomingData, incomingData.length);

clientSocket.receive(datagram);

receivedMessage= new String(datagram.getData()).trim();

receivedNumber = Integer.parseInt(receivedMessage);

System.out.println("Number received from client:"+receivedNumber);

List<Integer> list = new ArrayList<Integer>();

list.add(receivedNumber);

for(int i=0;i<list.size();i++)

{

sum = sum+list.get(i);

}

}

String msge = Integer.toString(sum);

System.out.println("Sum is :"+msge);

byte[] buff = msge.getBytes();

datagram.setData(buff);

datagram.setLength(buff.length);

clientSocket.send(datagram);

clientSocket.close();

}

catch (IOException e) {

System.out.println("Error: " + e);

System.exit(0);

}

catch (NumberFormatException ne) {

System.out.println("Error: " + ne);

System.exit(0);

}

}

}

**The client class**:

import java.io.\*;

import java.net.\*;

import java.util.\*;

public class MyClient {

public static void main(String[] args) throws IOException, InterruptedException {

Scanner keyboard = new Scanner(System.in);

DatagramSocket serverSocket = null;

try {

final int MAX\_LEN = 100;

serverSocket= new DatagramSocket();

String msg ="" ;

InetAddress ip = InetAddress.getByName("localhost");

System.out.println("Enter number to send to server:");

msg = keyboard.nextLine();

int number = Integer.parseInt(msg);

byte[] buff = msg.getBytes();

DatagramPacket datagram=new DatagramPacket(buff,buff.length,ip,6456);

serverSocket.send(datagram);

if(number == 0)

{

byte[] buffer = new byte[MAX\_LEN];

datagram = new DatagramPacket(buffer, buffer.length);

serverSocket.receive(datagram);

String message = new String(buffer);

System.out.println("Sum received from server: " + message);

}

serverSocket.close();

} catch (IOException e) {

System.out.println("Error: " + e);

System.exit(0);

}

catch(NumberFormatException ne)

{

System.out.println("Error: " + ne);

System.exit(0);

}

}

}

**STEPS TO RUN THE PROGRAM**

1.Start MyServer.java first

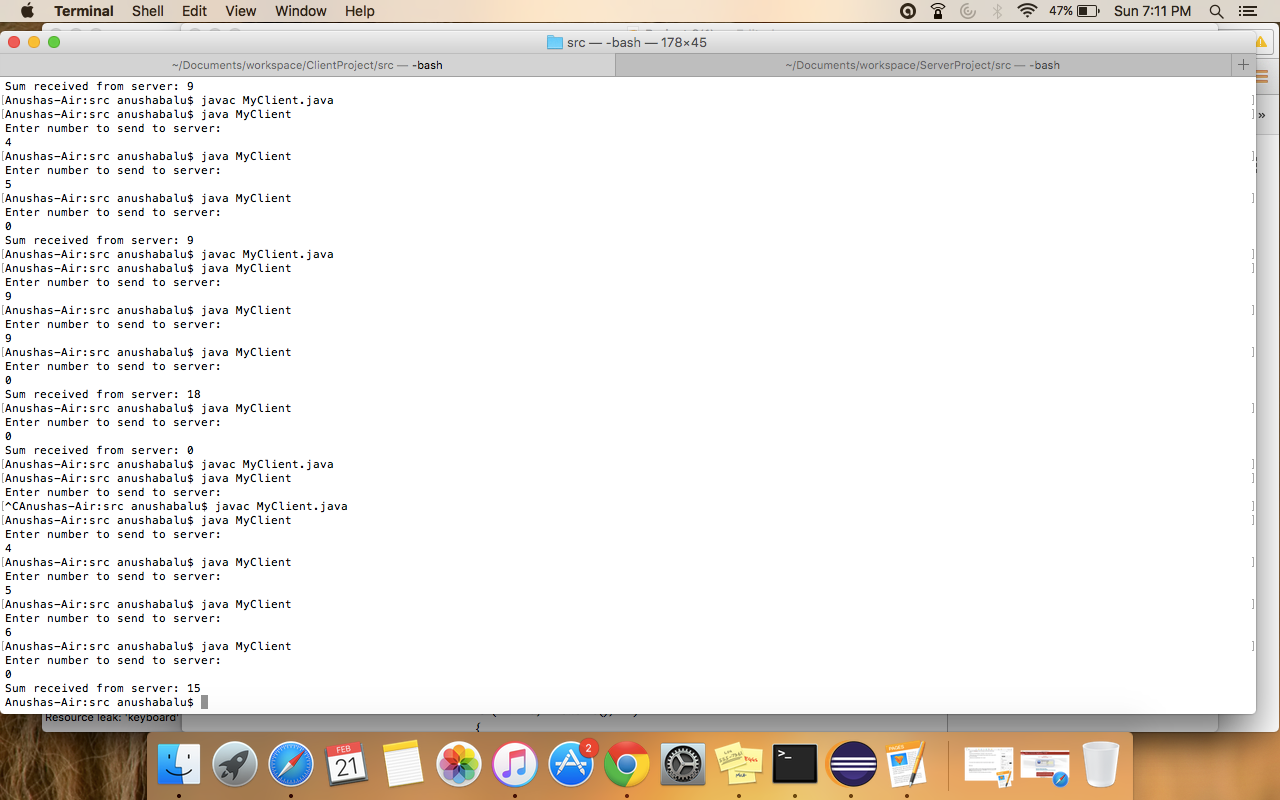
2.Start MyClient.java and enter a random integer.

3.The MyClient will terminate after you entering an integer.(As it is expected from the given question)

4.Start MyClient.java again and enter another integer and the step 3 will happen again.Repeat it how many ever times you want.

5.Do step 2 again and enter 0 and the server displays the sum of numbers that you entered till 0,sends it to client and terminates.The client displays the sum received from server and terminates.

**OUTPUT:**

**MyClient:**

**MyServer:**

