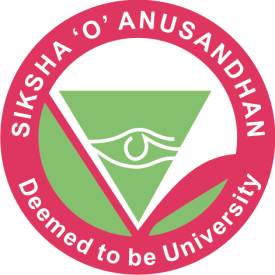
SIKSHA ‘O’ ANUSANDHAN



**DEEMED TO BE UNIVERSITY**

**Admission Batch: Session: 2020-2021**

End-Term Project Report Computer Science Workshop 1 (CSE 3141)

***Submitted by***

Name: **ASHUTOSH DASH**

Registration No.: **1941012274**

Branch: **CSE**

Semester: **3RD**  Section: **Q**

Department of Computer Science & Engineering

Faculty of Engineering & Technology (ITER)

**Jagamohan Nagar, Jagamara, Bhubaneswar, Odisha – 751030**

**CONTENT**

|  |  |  |
| --- | --- | --- |
| **S.No.** | **Title** | **Page Number** |
| **01** | **Problem Statement** | **1** |
| **02** | **Problem analysis** | **2 to 3** |
| **03** | **Flow Diagram** | **4** |
| **04** | **Program Code** | **5 to 11** |
| **05** | **Snapshot of input and output** | **12 to 13** |
| **06** | **Author details and date of editing** | **14** |

1. **Problem Statement**

Write a java program for reading and writing Serialized data using TCP/IP Server and Client Socket programing. Client sending requests to server and a server for accepting those requests and responding, replying back to the clients. In this program you have to illustrate how to add new student details and display the details. Details must include student id, name, roll no, course, mobile number, email id. There should be a unique student id for every student and also, there should not be the same roll no, mobile number and email id for each of the student.

1. **Problem Analysis**

A client creates a socket at it’s end of transmission, and strive to connect the socket to server. When a connection is established, server creates a socket at it’s end and, client and server can now ready communicate through writing and reading methods.

1. An object of **ServerSocket** is instantiated, and desired port number is specified, on which connection is going to take place.

2. The **accept** method of **ServerSocket** is invoked, in order to hold the server in listening mode. This method won’t resume until a client is connected to the server through the given port number.

3. Now, on client side, an object of **Socket** is instantiated, and desired port number and IP address is specified for the connection.

4. An attempt is made, for connecting the client to the server using the specified IP address and port number. If attempt is successful, client is provided with a **Socket** that is capable of communicating to the respective server, with write and read methods. If unsuccessful, desired exception is raised.

5. Since a client is connected to the server, **accept** method on the server side resumes, providing a **Socket** that is capable of communicating to the connected client.

6. Once the communication is completed, terminate the sockets on both, the server and the client side.

**Client Side Programming**

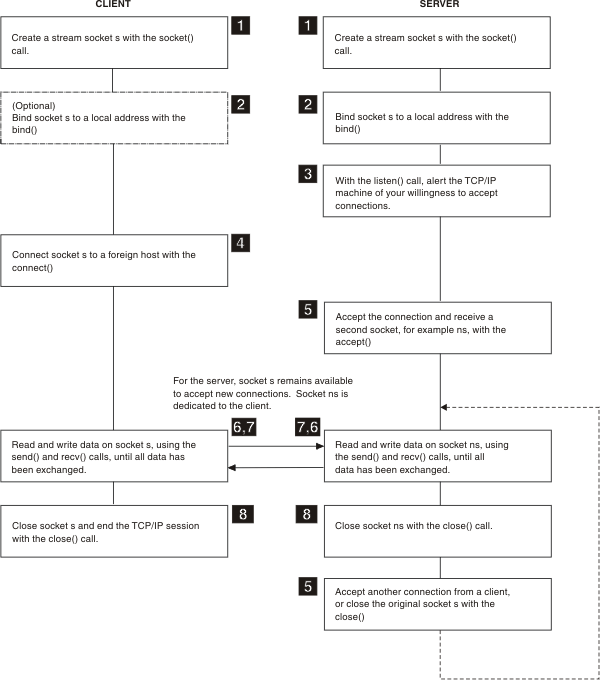
For the Client machine, we need to now establish the socket connection. Socket Connection is when the two machines have information about each other’s network location (IP Address) and the TCP port.

**Server Side Programming**

For the server side programming, a ServerSocket is required, that will wait for the client in the listening mode, on a particular TCP port. This Server Socket holds until a Client Socket is connected successfully. As soon as the client is connected, another Socket comes to the existence that will enable data sharing between the respective client and server. A temporary Socket is created to handle the request from that particular client and our main server socket will be free again, to listen to other requests. Hence we have ServerSocket and Socket on the server side.

**Server is listening on port : 2205**is the proof of concept that, once the request is arrived, the server diverts it onto another thread and gets ready for listening to another request. And extracting and processing the data takes place on another thread in parallel.

1. **Flow Diagram**



1. **Program Code**

**Client:**

import java.io.\*;

import java.net.\*;

public class Client {

public static void main(String[] data) {

***//data is taken as command line argument***

String ipAddress=data[0];

int portNumber=Integer.parseInt(data[1]);

String name=data[2];

int rollno=Integer.parseInt(data[3]);

String course=data[4];

long phno=Long.parseLong(data[5]);

String email=data[6];

long id=Long.parseLong(data[7]);

String request=name+" , "+rollno+" , "+course+" , "+phno+" , "+email+" , "+id+'#';

***//# acts as a terminator***

try {

Socket socket=new Socket(ipAddress , portNumber);

***//Socket is initialized and attempt is made for connecting to the server***

***//Declaring other properties and streams***

OutputStream outputStream;

OutputStreamWriter outputStreamWriter;

InputStream inputStream;

InputStreamReader inputStreamReader;

StringBuffer stringBuffer;

String response;

int x;

***//retrieving output Stream and its writer, for sending request or acknowledgement***

outputStream=socket.getOutputStream();

outputStreamWriter=new OutputStreamWriter(outputStream);

outputStreamWriter.write(request);

outputStreamWriter.flush(); ***// request is sent***

***// retrieving input stream and its reader, for receiving acknowledgement or response***

inputStream=socket.getInputStream();

inputStreamReader=new InputStreamReader(inputStream);

stringBuffer=new StringBuffer();

while(true) {

x=inputStreamReader.read();

if(x=='#' || x==-1) break; // reads till the terminator

stringBuffer.append((char)x);

}

response=stringBuffer.toString();

System.out.println(response);

socket.close(); ***//closing the connection***

}

catch(Exception exception) {

***// Raised in case, connection is refused or some other technical issue***

System.out.println(exception);

}

}

}

**Server:**

import java.io.\*;

import java.net.\*;

class RequestProcessor extends Thread { ***//for multi-threaded server***

private Socket socket;

RequestProcessor(Socket socket) {

this.socket=socket;

start(); ***// will load the run method***

}

public void run() {

try {

***//Declaring properties and streams***

OutputStream outputStream;

OutputStreamWriter outputStreamWriter;

InputStream inputStream;

InputStreamReader inputStreamReader;

StringBuffer stringBuffer;

String response;

String request;

int x;

int temp1,temp2,temp3,temp4,temp5;

String part1,part2,part3,part4,part5,part6;

String name;

int rollno;

String course;

long phno;

String email;

long id;

***//getting input stream and its reader, for reading request or acknowledgement***

inputStream=socket.getInputStream();

inputStreamReader=new InputStreamReader(inputStream);

stringBuffer=new StringBuffer();

while(true) {

x=inputStreamReader.read();

if(x=='#' || x==-1) break; ***//reads until terminator***

stringBuffer.append((char)x);

}

request=stringBuffer.toString();

System.out.println("Request : "+request);

***//parsing and extracting Request data***

temp1=request.indexOf(",");

temp2=request.indexOf(",",temp1+1);

temp3=request.indexOf(",",temp2+1);

temp4=request.indexOf(",",temp3+1);

temp5=request.indexOf(",",temp4+1);

part1=request.substring(0,temp1);

part2=request.substring(temp1+1,temp2);

part3=request.substring(temp2+1,temp3);

part4=request.substring(temp3+1,temp4);

part5=request.substring(temp4+1,temp5);

part6=request.substring(temp5+1);

name=part1;

rollno=Integer.parseInt(part2);

course=part3;

phno=Long.parseLong(part4);

email=part5;

id=Long.parseLong(part6);

System.out.println("Name : "+name);

System.out.println("Roll number : "+rollno);

System.out.println("Course : "+course);

System.out.println("Phone number : "+phno);

System.out.println("Email : "+email);

System.out.println("ID : "+id);

***// handle data***

***//sending response***

response="Data saved#";

***//get output stream and its writer, for sending response or acknowledgement***

outputStream=socket.getOutputStream();

outputStreamWriter=new OutputStreamWriter(outputStream);

outputStreamWriter.write(response);

outputStreamWriter.flush(); ***// response sent***

System.out.println("Response sent");

socket.close(); ***//terminating connection***

}

catch(Exception exception) {

System.out.println(exception);

}

}

}

public class Server {

private ServerSocket serverSocket;

private int portNumber;

Server(int portNumber) {

this.portNumber=portNumber;

try {

***//Initiating ServerSocket with TCP port***

serverSocket=new ServerSocket(this.portNumber);

startListening();

}

catch(Exception e) {

System.out.println(e);

System.exit(0);

}

}

private void startListening() {

try {

Socket socket;

while(true) {

System.out.println("Server is listening on port : "+this.portNumber);

socket=serverSocket.accept(); ***// server is in listening mode***

System.out.println("Request arrived..");

***// diverting the request to processor with the socket reference***

new RequestProcessor(socket);

}

}

catch(Exception e) {

System.out.println(e);

}

}

public static void main(String[] data) {

int portNumber=Integer.parseInt(data[0]);

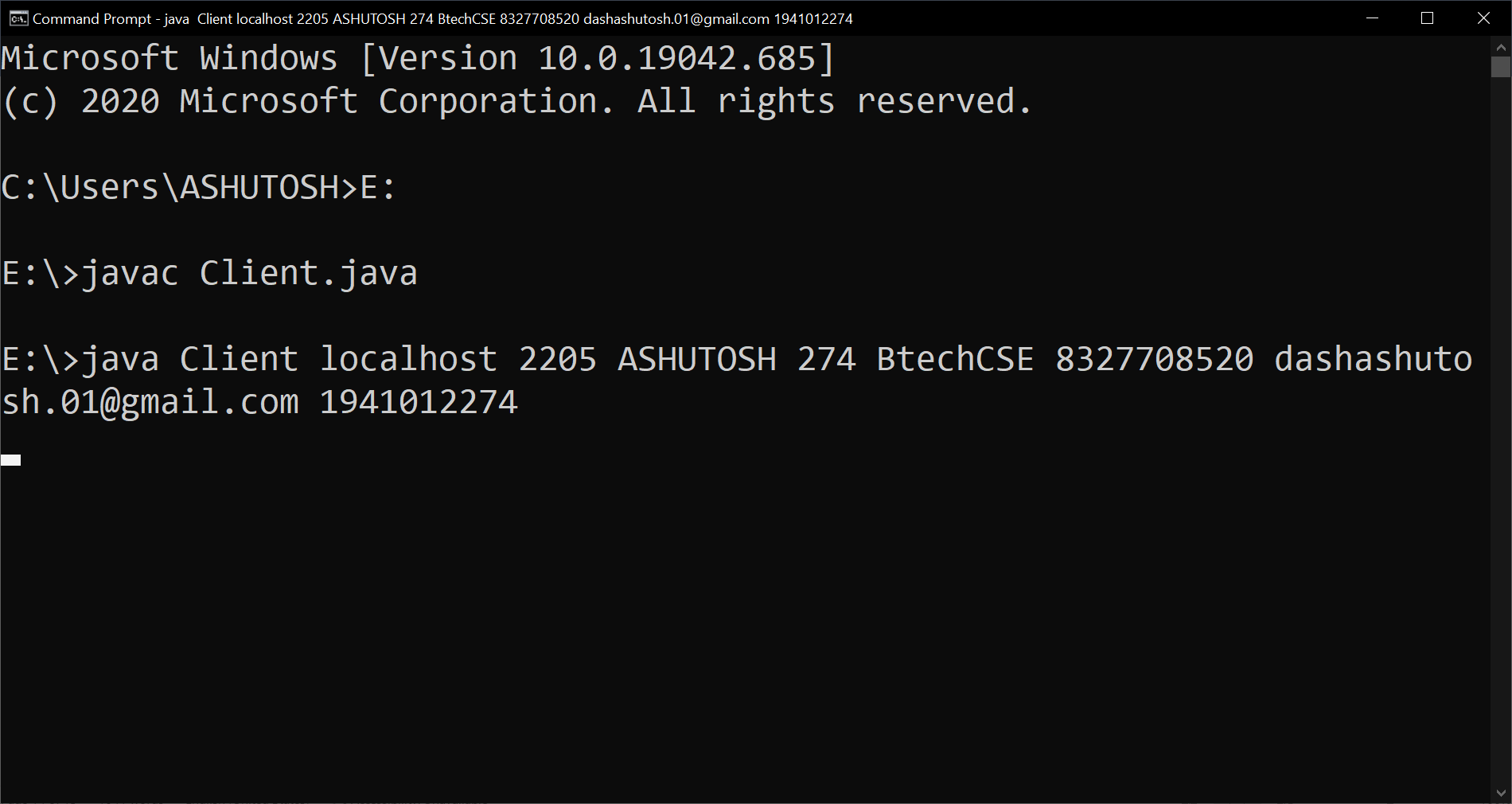
Server server=new Server(portNumber);

}

}

1. **Input & Output**







1. **Author details & Date of editing**

**Name:** ASHUTOSH DASH

**Registration no:** 1941012274

**Branch:** Computer Science Engineering

**Section:** Q

**Date:** 20.01.2021