–/\*

CL-9

BE-9 R9

Roll No. 43144

**Assignment 1 A**

Power Calculation: Design a distributed application which consist of a client server communication using TCP, UDP & RMI techniques in Java. Multiple clients can simultaneously connect to the server and send messages of the format -> (a, b) where a and bare integers and server returns the value a^b (a raised to b).

\*/

**// Power Calculation: using TCP**

**//Client.java**

import java.io.DataInputStream;

import java.io.DataOutputStream;

import java.io.IOException;

import java.net.InetAddress;

import java.net.Socket;

import java.net.UnknownHostException;

import java.util.Scanner;

public class Client

{

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

{

InetAddress ip = InetAddress.getLocalHost();

int port = 4444;

Scanner sc = new Scanner(System.in);

// Step 1: Open the socket connection.

Socket s = new Socket(ip, port);

// Step 2: Communication-get the input and output stream

DataInputStream dis = new DataInputStream(s.getInputStream());

DataOutputStream dos = new DataOutputStream(s.getOutputStream());

while (true) {

// User Menu

System.out.println(

"\n1.Power function\n2.Exit");

System.out.println("Enter the option:");

int opt = sc.nextInt();

if (opt == 2) {

break;

}

System.out.println("Enter the the first number:");

double a = sc.nextDouble();

// send the numbers to server

dos.writeDouble(a);

System.out.println("Enter the second number:");

double b = sc.nextDouble();

// send the numbers to server

dos.writeDouble(b);

double n = dis.readDouble();

System.out.println(a + " raised to " + b + " = " + n);

}

}

}

**//Server.java**

import java.io.DataInputStream;

import java.io.DataOutputStream;

import java.io.IOException;

import java.net.ServerSocket;

import java.net.Socket;

import java.util.StringTokenizer;

import java.lang.Math;

public class Server

{

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

{

// Step 1: Establish the socket connection.

ServerSocket ss = new ServerSocket(4444);

Socket s = ss.accept();

// Step 2: Processing the request.

DataInputStream dis = new DataInputStream(s.getInputStream());

DataOutputStream dos = new DataOutputStream(s.getOutputStream());

// wait for input

Double a = dis.readDouble();

Double b = dis.readDouble();

Double result = Math.pow(a, b);

System.out.println("Sending the result...");

// send the result back to the client.

dos.writeDouble(result);

}

}

// OUTPUT

Graphical user interface

Description automatically generated