

**19BIT0292**

**Bhaumik Tandan**

**ASSESSMENT-3**

**ADVANCED JAVA**

**PROGRAMMING**

ITE2005

L25+L26

**Q1)** Develop a java application for demonstrating the two way chat using UDP

**CODE**

import java.util.Scanner;

import java.net.DatagramPacket;

import java.net.DatagramSocket;

import java.net.SocketException;

import java.lang.Thread;

import java.net.InetAddress;

class Receive extends Thread{

DatagramSocket socket;

public Receive(DatagramSocket socket){

this.socket = socket;

}

public void run()

{

DatagramPacket packet =null;

byte buffer[];

try{

while(true){

buffer=new byte[1024];

packet=new DatagramPacket(buffer,1024);

socket.receive(packet);

String str = new String(buffer);

System.out.println("Message Received: "+str);

}

}

catch(Exception e){

System.out.println("Error in receiving packet");

}

}

}

class Send extends Thread{

DatagramSocket socket;

int port;

Send(DatagramSocket socket,int port){

this.socket = socket;

this.port=port;

}

public void run()

{

while(true)

{

try{

Scanner s=new Scanner(System.in);

String str=s.nextLine();

byte buffer[]=str.getBytes();

DatagramPacket packet = new DatagramPacket(buffer,buffer.length,InetAddress.getLocalHost(),this.port);

socket.send(packet);

}

catch(Exception e){

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

}

}

}

}

class Q1

{

public static void main(String[] a)

{

Scanner s=new Scanner(System.in);

System.out.print("Enter your port number: ");

int port=s.nextInt();

System.out.print("Enter the other port: ");

int receiver\_port=s.nextInt();

try{

DatagramSocket ds=new DatagramSocket(port);

Receive r=new Receive(ds);

r.start();

Send send=new Send(ds,receiver\_port);

send.start();

}

catch(SocketException e){

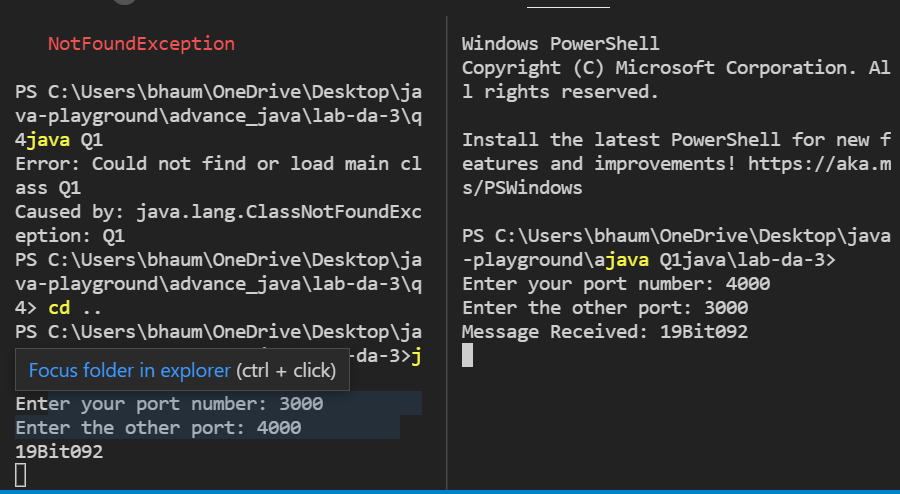
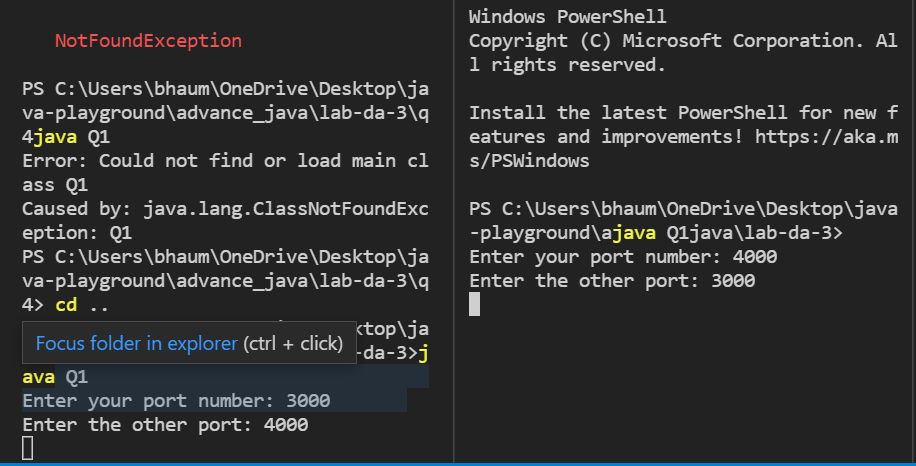
System.out.println("Error in creating socket: "+e);

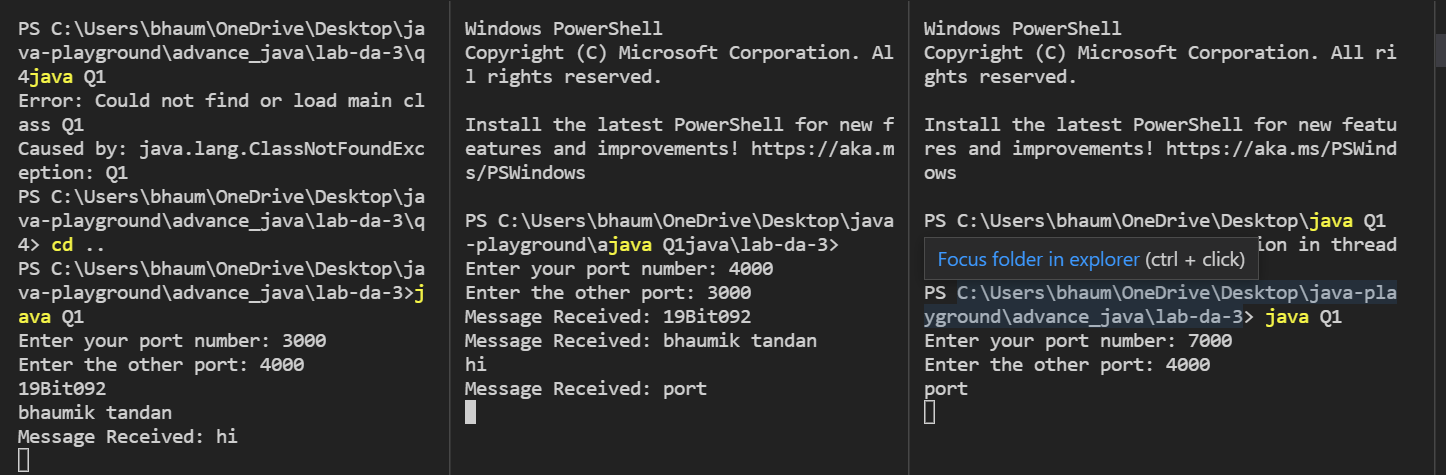
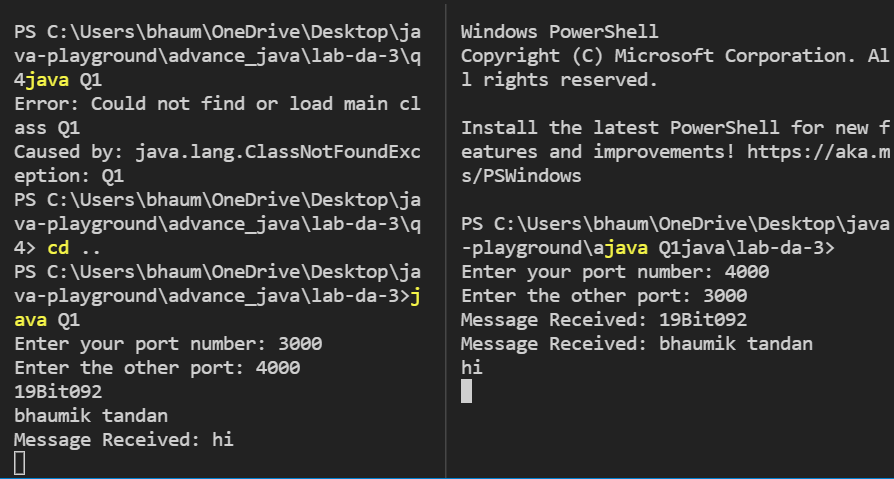
}

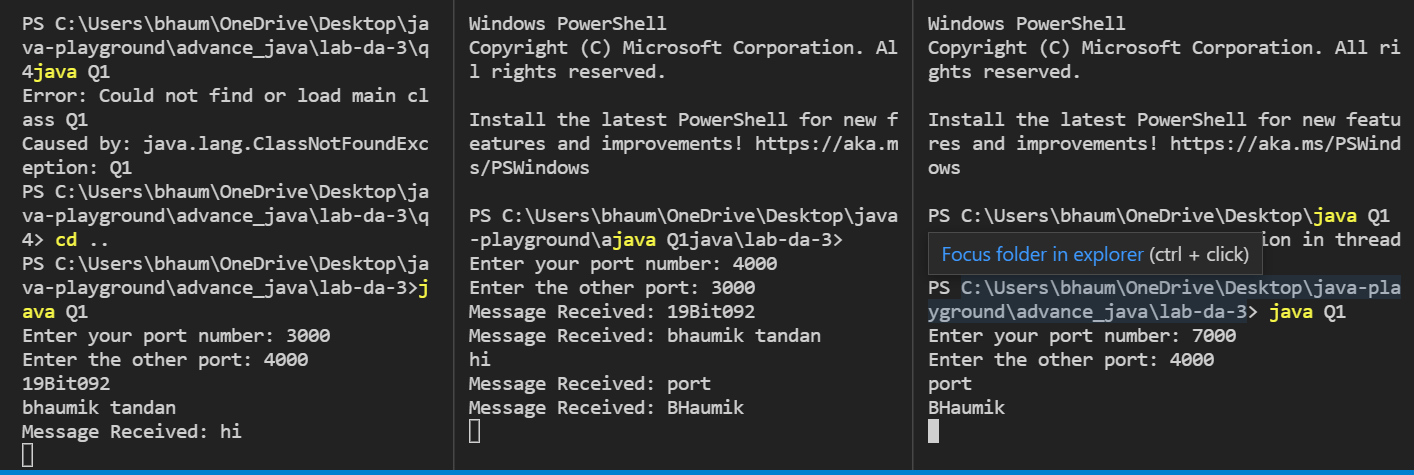
}

}

**OUTPUT**

****

****

****

**Q2)** Demonstrate RMI based String manipulation application

**Server.java**

**CODE**

import java.rmi.\*;

import java.rmi.registry.\*;

import java.rmi.server.\*;

class StringManipulator extends UnicastRemoteObject implements StringManipulatorInterface{

StringManipulator() throws RemoteException

{

super();

}

public String add(String s1, String s2) throws RemoteException

{

return s1 + s2;

}

public String toLowerCase(String s) throws RemoteException

{

return s.toLowerCase();

}

public String toUpperCase(String s) throws RemoteException

{

return s.toUpperCase();

}

public String substring(String s,int StartIndex, int LastIndex) throws RemoteException

{

return s.substring(StartIndex, LastIndex);

}

}

public class Server

{

public static void main(String args[])

{

try

{

StringManipulator obj = new StringManipulator();

LocateRegistry.createRegistry(1800);

Naming.rebind("rmi://localhost:1800"+"/stringManipulator",obj);

}

catch(Exception ae)

{

System.out.println(ae);

}

}

}

**Client.java**

**CODE**

import java.rmi.\*;

public class Client {

public static void main(String args[]){

try{

StringManipulatorInterface stub=(StringManipulatorInterface)Naming.lookup("rmi://localhost:1800/stringManipulator");

System.out.println("ADD: "+stub.add("Bhaumik ","Tandan"));

System.out.println("To upper: "+stub.toUpperCase("bhaumik"));

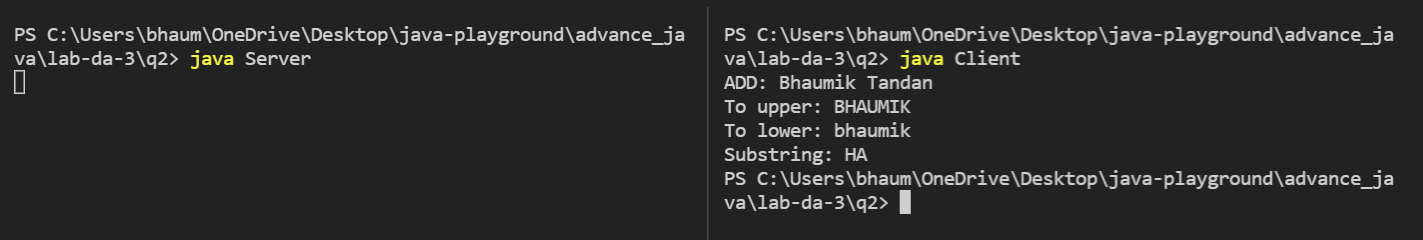
System.out.println("To lower: "+stub.toLowerCase("BHAUMIK"));

System.out.println("Substring: "+stub.substring("BHAUMIK",1,3));

}catch(Exception e){}

}

}

****

**Q3)** Develop a TCP based multithreaded server to compute the addition of complex number. Let the clients pass the complex number as inputs to the server.

**Server.java**

**CODE**

import java.io.\*;

import java.net.\*;

class Server {

public static void main(String[] args)

{

ServerSocket server = null;

try {

server = new ServerSocket(1234);

server.setReuseAddress(true);

while (true) {

Socket client = server.accept();

System.out.println("New client connected"

+ client.getInetAddress()

.getHostAddress());

ClientHandler clientSock

= new ClientHandler(client);

new Thread(clientSock).start();

}

}

catch (IOException e) {

e.printStackTrace();

}

finally {

if (server != null) {

try {

server.close();

}

catch (IOException e) {

e.printStackTrace();

}

}

}

}

private static class ClientHandler implements Runnable {

private final Socket clientSocket;

public ClientHandler(Socket socket)

{

this.clientSocket = socket;

}

public void run()

{

PrintWriter out = null;

BufferedReader in = null;

try {

out = new PrintWriter(

clientSocket.getOutputStream(), true);

in = new BufferedReader(

new InputStreamReader(

clientSocket.getInputStream()));

String line;

while ((line = in.readLine()) != null) {

System.out.printf(

" Sent from the client: %s\n",

line);

out.println(line);

}

}

catch (IOException e) {

e.printStackTrace();

}

finally {

try {

if (out != null) {

out.close();

}

if (in != null) {

in.close();

clientSocket.close();

}

}

catch (IOException e) {

e.printStackTrace();

}

}

}

}

}

**Client.java**

**CODE**

import java.io.\*;

import java.net.\*;

import java.util.\*;

class Client {

public static void main(String[] args)

{

try (Socket socket = new Socket("localhost", 1234)) {

PrintWriter out = new PrintWriter(

socket.getOutputStream(), true);

BufferedReader in

= new BufferedReader(new InputStreamReader(

socket.getInputStream()));

Scanner sc = new Scanner(System.in);

String line = null;

while (!"exit".equalsIgnoreCase(line)) {

line = sc.nextLine();

out.println(line);

out.flush();

System.out.println("Server replied "

+ in.readLine());

}

sc.close();

}

catch (IOException e) {

e.printStackTrace();

}

}

}

**Q4)** Using ServletOutStream and ServletInputStream handle the display of images in the web application.

**CODE**

package com.javatpoint;

import java.io.\*;

import javax.servlet.\*;

import javax.servlet.http.\*;

public class Q4 extends HttpServlet {

public void doGet(HttpServletRequest request,HttpServletResponse response)

throws IOException

{

response.setContentType("image/jpeg");

ServletOutputStream out;

out = response.getOutputStream();

FileInputStream fin = new FileInputStream("https://www.puma-catchup.com/wp-content/uploads/2022/03/gallery-RDH-750x417.jpg");

BufferedInputStream bin = new BufferedInputStream(fin);

BufferedOutputStream bout = new BufferedOutputStream(out);

int ch =0; ;

while((ch=bin.read())!=-1)

{

bout.write(ch);

}

bin.close();

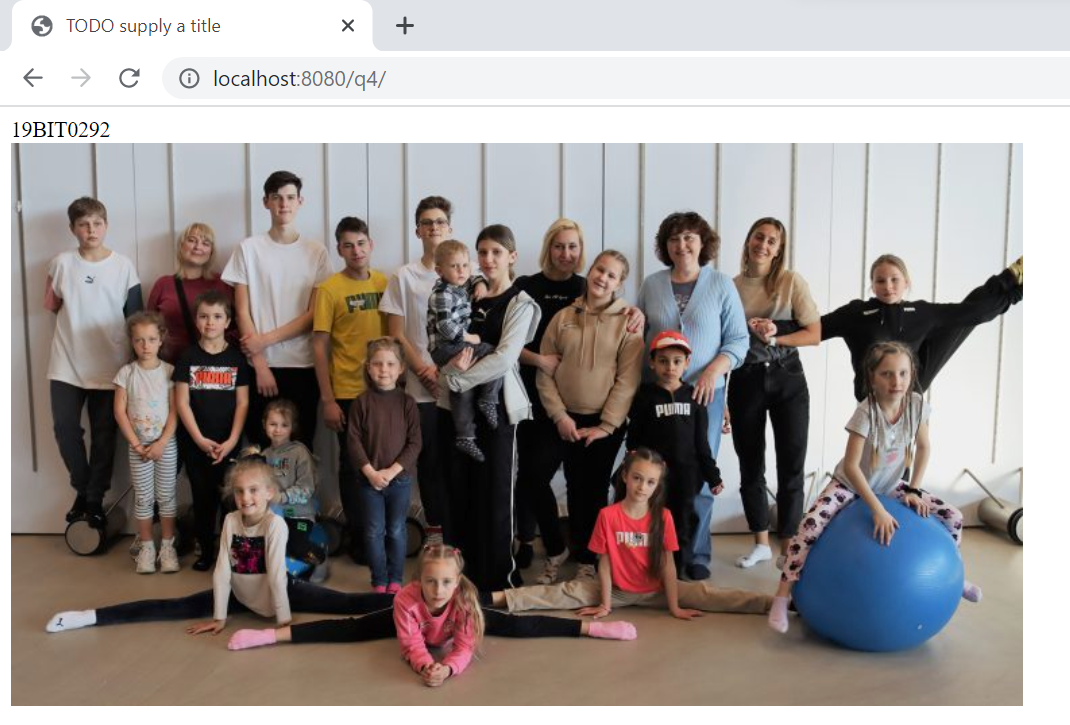
fin.close();

bout.close();

out.close();

}

}

****

**Q5)** Develop a simple JDBC application to display the result of pattern matching query executed over the ‘book’ table in a tabular format.

**CODE**

import java.sql.\*;

public class connect {

Connection con = null;

public static Connection connectDB()

{

try {

Class.forName("com.mysql.jdbc.Driver");

Connection con = DriverManager.getConnection(

"jdbc:mysql://localhost:3306/book",

"root", "");

return con;

}

catch (SQLException e) {

System.out.println(e);

}

}

}

import java.sql.\*;

public class Q5 {

public static void main(String[] args)

{

Connection con = null;

PreparedStatement p = null;

ResultSet rs = null;

con = connect.connectDB();

try {

String sql = "select \* from books";

p = con.prepareStatement(sql);

rs = p.executeQuery();

System.out.println("id\t\tname\t\temail");

while (rs.next()) {

int id = rs.getInt("id");

String name = rs.getString("name");

String email = rs.getString("email");

System.out.println(id + "\t\t" + name

+ "\t\t" + email);

}

}

catch (SQLException e) {

System.out.println(e);

}

}

}

**Q6)** Develop a Servlet application to display the various cookies stored in the client machine.

**CODE**

import java.io.\*;

import javax.servlet.\*;

import javax.servlet.http.\*;

public class ReadCookies extends HttpServlet {

public void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

Cookie cookie = null;

Cookie[] cookies = null;

cookies = request.getCookies();

response.setContentType("text/html");

PrintWriter out = response.getWriter();

String title = "Reading Cookies Example";

String docType =

"<!doctype html public \"-//w3c//dtd html 4.0 " +

"transitional//en\">\n";

out.println(docType +

"<html>\n" +

"<head><title>" + title + "</title></head>\n" +

"<body bgcolor = \"#f0f0f0\">\n" );

if( cookies != null ) {

out.println("<h2> Found Cookies Name and Value</h2>");

for (int i = 0; i < cookies.length; i++) {

cookie = cookies[i];

out.print("Name : " + cookie.getName( ) + ", ");

out.print("Value: " + cookie.getValue( ) + " <br/>");

}

} else {

out.println("<h2>No cookies founds</h2>");

}

out.println("</body>");

out.println("</html>");

}

}