

UNIT NO	List of Practical	Date	Signature
<u>1</u>	Write a program for implementing Client Server communication model using TCP		
<u>2</u>	Write a program for implementing the Client Server communication model using UDP.		
<u>3</u>	Write a program to show the object communication using RMI.		
<u>4</u>	Show the implementation of web services.		
<u>5</u>	Study and implementation of Infrastructure as a Service. and study of cloud computing Architecture		
<u>6</u>	Installation and Configuration of virtualization using KVM.		
<u>7</u>	Study and implementation of Infrastructure as a Service		
<u>8</u>	Study and implementation of Storage as a Service		
<u>9</u>	Study and implementation of Identity Management		
<u>10</u>	Study Cloud Security management		
<u>11</u>	Write a program for the web feed.		
<u>12</u>	Study and implementation of Single-Sign-On.		
<u>13</u>	User Management in the Cloud		
<u>14</u>	Case study on Amazon EC2/Microsoft Azure/Google Cloud Platform		

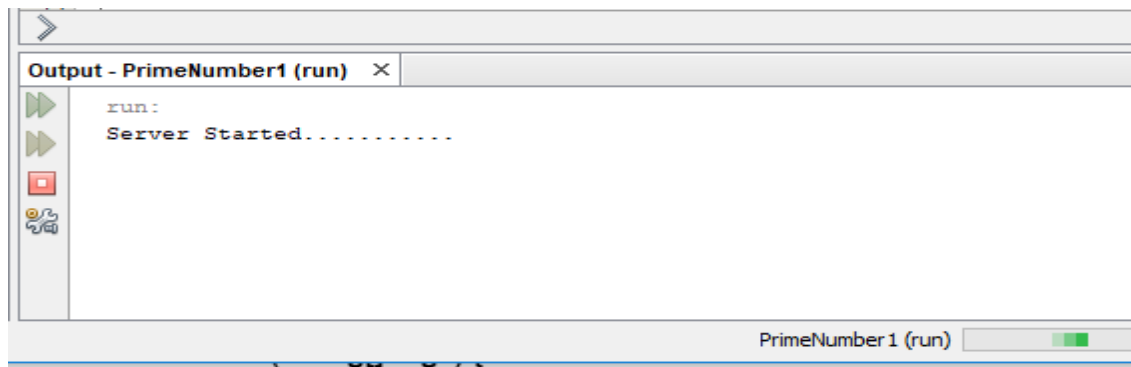
PRACTICAL 1

Date: 05-12-24

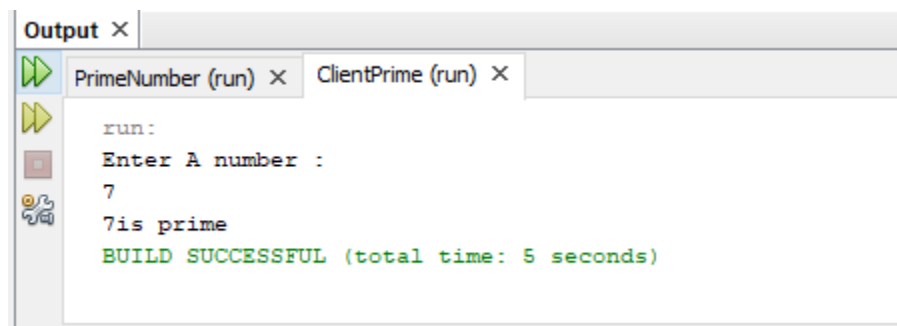
Aim: Write a program for implementing a client server communication model using TCP.

a) A client-server based program using TCP to find if the number entered is prime or not. Server Code:

```
package primenumber;
import java.net.*; import
java.io.*;
public class PrimeNumber{
    public static void main(String[] args){
        try{
            ServerSocket ss = new ServerSocket(8001);
            System.out.println("Server Started..... ");
            Socket s = ss.accept();
            DataInputStream in = new DataInputStream(s.getInputStream()); int x
            = in.readInt();
            DataOutputStream otc = new DataOutputStream(s.getOutputStream()); boolean
            isPrime = true;
            if (x <= 1) {
                isPrime = false;
            } else {
                for (int i = 2; i <= x / 2; i++) {
                    if (x % i == 0) {
                        isPrime = false;
                        break;
                    }
                }
            }
            if (isPrime) {
                otc.writeUTF(x + " is prime");
            } else {
                otc.writeUTF(x + " is not prime");
            }
            s.close();
            ss.close();
        } catch (Exception e) {
            System.out.println(e.toString());
        }
    }
}
```

OUTPUT:Client-code:

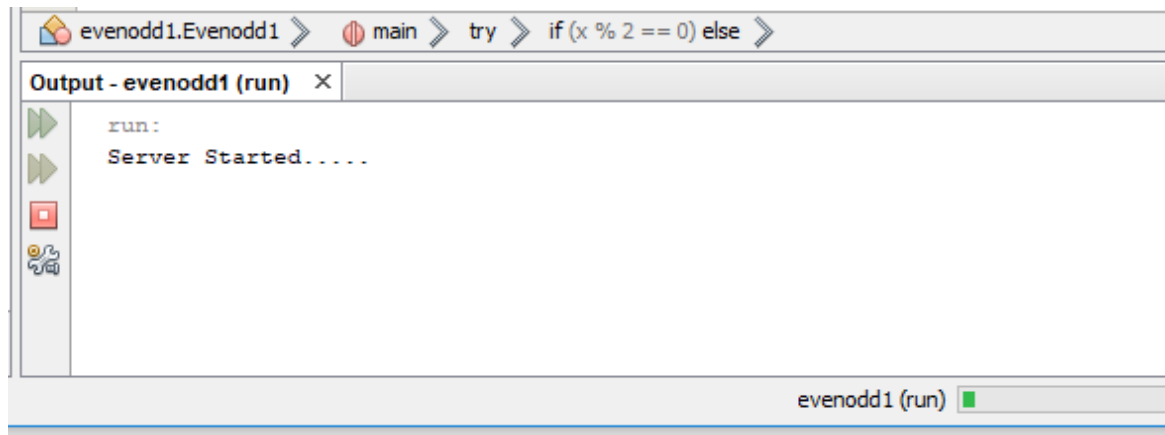
```
package clientprime;
import java.io.*;
import java.net.*;
public class ClientPrime {
    public static void main(String[] args) { try{
        Socket cs = new Socket("localhost",8001);
        BufferedReader infu = new BufferedReader(new InputStreamReader(System.in));
        System.out.println("Enter A number :");
        int a = Integer.parseInt(infu.readLine());
        DataOutputStream out = new DataOutputStream(cs.getOutputStream());
        out.writeInt(a);
        DataInputStream in = new DataInputStream(cs.getInputStream());
        System.out.println(in.readUTF());
        cs.close();
    }
    catch(Exception e){
        System.out.println(e.toString());
    }
    }
}
```

OUTPUT:

b) A client-server based program using TCP to find if the number entered is Even or Odd. Server code:

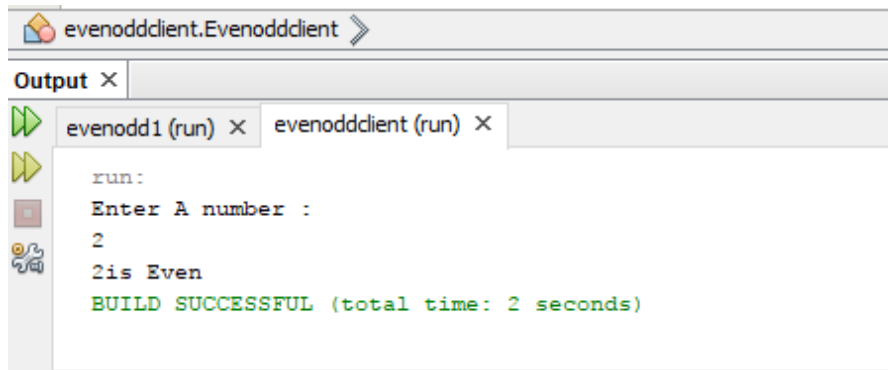
```
package evenodd1;
import java.net.*;
import java.io.*;
public class Evenodd1 {
    public static void main(String[] args) throws IOException {
        try{
            ServerSocket ss = new ServerSocket(8002);
            System.out.println("Server Started ..... ");
            Socket s = ss.accept();
            DataInputStream in = new DataInputStream(s.getInputStream()); int x
            = in.readInt();
            DataOutputStream otc = new DataOutputStream(s.getOutputStream()); if(x %
            2==0)
            {
                otc.writeUTF(x + "is Even");
            }
            else{
                otc.writeUTF(x + "is odd");
            }
        }
        catch(Exception e){
            System.out.println(e.toString());
        }
    }
}
```

OUTPUT:



Client-code:

```
package evenoddclient;
import java.net.*; import
java.io.*;
public class Evenoddclient {
    public static void main(String[] args) { try{
        Socket cs = new Socket("LocalHost",8002);
        BufferedReader infu = new BufferedReader(new InputStreamReader(System.in));
        System.out.println("Enter A number :");
        int a = Integer.parseInt(infu.readLine());
        DataOutputStream out = new DataOutputStream(cs.getOutputStream());
        out.writeInt(a);
        DataInputStream in = new DataInputStream(cs.getInputStream());
        System.out.println(in.readUTF());
        cs.close();
    }
    catch(Exception e){
        System.out.println(e.toString());
    }
}
}
```

OUTPUT:

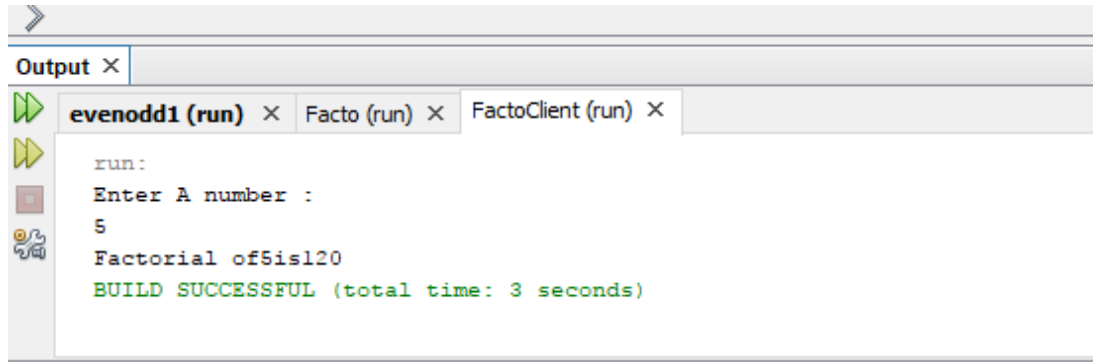
c) A client-server based program using TCP to find the factorial of the number entered. Server code:

```
package facto; import
java.io.*; import
java.net.*; public class
Facto {
    public static void main(String[] args) { try{
        ServerSocket ss = new ServerSocket(8003);
        System.out.println("Server Started..... ");
        Socket s = ss.accept();
        DataInputStream in = new DataInputStream(s.getInputStream()); int x
        = in.readInt();
        DataOutputStream otc = new DataOutputStream(s.getOutputStream()); long
        factorial=1;
        for(int i=1;i<=x;i++){
            factorial *= i;
        }
        otc.writeUTF("Factorial of" + x+"is"+factorial);
    }
    catch(Exception e){
        System.out.println(e.toString());
    }
}
}
```

Client-code package

```
factoclient; import
java.io.*; import
java.net.*;
public class FactoClient {
    public static void main(String[] args) { try{
        Socket cs = new Socket("LocalHost",8003);
        BufferedReader infu = new BufferedReader(new InputStreamReader(System.in));
        System.out.println("Enter A number :");
        int a = Integer.parseInt(infu.readLine());
        DataOutputStream out = new DataOutputStream(cs.getOutputStream());
        out.writeInt(a);
        DataInputStream in = new DataInputStream(cs.getInputStream());
        System.out.println(in.readUTF());
        cs.close();
    }
    catch(Exception e){
```

```
        System.out.println(e.toString());  
    }  
}  
}
```

OUTPUT:

PRACTICAL 2

Date: 12-12-24

Aim: Write a program for implementing the Client-Server communication model using UDP.

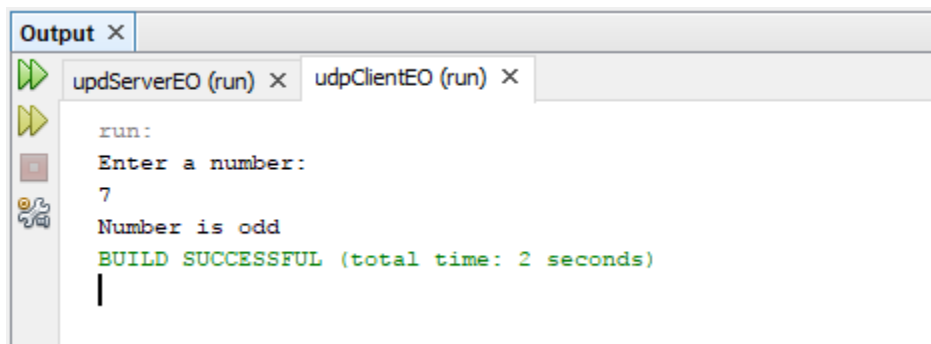
a) Client-Server based program UDP to find the number entered is even or odd.

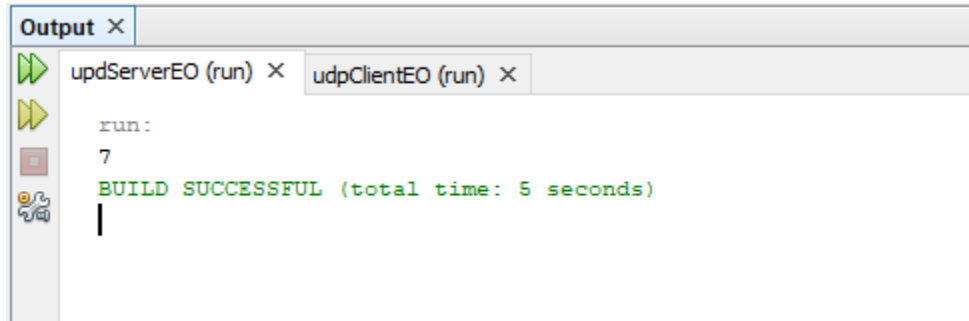
Server Code:

```
package evenoddsServer;
import java.io.*;
import java.net.*;
public class EvenOddServer {
    public static void main(String[] args) { try
    {
        DatagramSocket ds = new DatagramSocket(2000); byte
        b[] = new byte[1024];
        DatagramPacket dp = new DatagramPacket(b,b.length); ds.receive(dp);
        String str = new String(dp.getData(),0,dp.getLength());
        System.out.println(str);
        int a = Integer.parseInt(str);
        String s = new String();
        if(a%2==0)
            s = "Number is even"; else
            s = "Number is odd"; byte
        b1[] = new byte[1024]; b1 =
        s.getBytes();
        DatagramPacket dp1 = new
        DatagramPacket(b1,b1.length,InetAddress.getLocalHost(),1000); ds.send(dp1);
    }
    catch(Exception e)
    {
        e.printStackTrace();
    }
    }
}
```

Client Code:

```
package evenoddcclient1;
import java.io.*;
import java.net.*;
public class EvenOddClient1 {
    public static void main(String[] args) { try
    {
        DatagramSocket ds = new DatagramSocket(1000);
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
        System.out.println("Enter a number :");
        String num = br.readLine();
        byte b[] = new byte[1024];
        b= num.getBytes();
        DatagramPacket dp = new
DatagramPacket(b,b.length,InetAddress.getLocalHost(),2000); ds.send(dp);
        byte b1[] = new byte[1024];
        DatagramPacket dp1 =new DatagramPacket(b1,b1.length);
        ds.receive(dp1);
        String str = new String(dp1.getData(),0,dp1.getLength());
        System.out.println(str);
    }
    catch(Exception e)
    {
        e.printStackTrace();
    }
    }
}
```

OUTPUT:



b) Client-server based program using UDP to find factorial of entered number.

Server Code package

serverfacto; import

java.io.*; import

java.net.*;

public class ServerFacto {

public static void main(String[] args) { try{

DatagramSocket ds = new DatagramSocket(2000); byte b

[] = new byte[1024];

DatagramPacket dp = new DatagramPacket(b,b.length); ds.receive(dp);

String str = new String(dp.getData(),0,dp.getLength());

System.out.println(str);

int num = Integer.parseInt(str);

long factorial = calculateFactorial(num);

String result = "Factorial of" + num + "is" + factorial; byte

b1[] = result.getBytes();

DatagramPacket dp1 = new DatagramPacket(b1,b1.length,InetAddress.getLocalHost(),1000);

ds.send(dp1);

}

catch(Exception e)

{

e.printStackTrace();

}

}

public static long calculateFactorial(int num)

{

long factorial =1; for(int

i=1;i<=num;i++){

factorial*=i;

}

return factorial;

```

    }
}

```

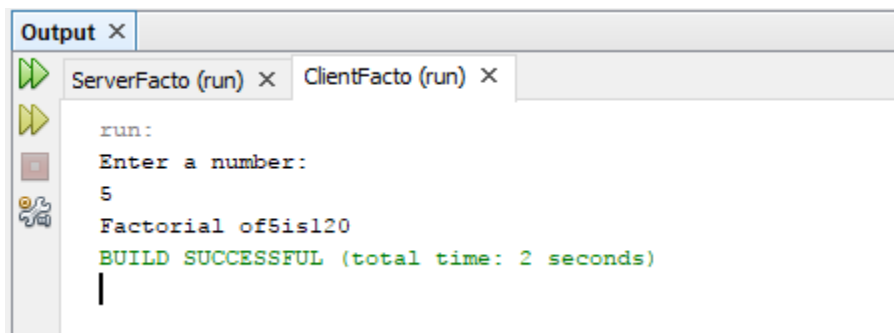
Client Code package

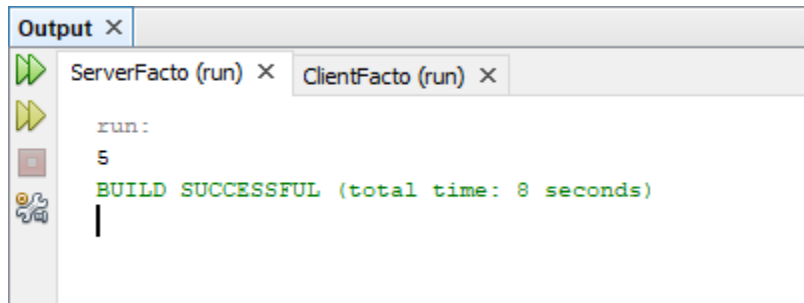
```

clientfacto; import
java.io.*; import
java.net.*;
public class ClientFacto {
    public static void main(String[] args) { try
    {
        DatagramSocket ds=new DatagramSocket(1000);
        BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
        System.out.println("Enter a number: ");
        String num=br.readLine(); byte[]
        b=num.getBytes();
        DatagramPacket dp=new
        DatagramPacket(b,b.length,InetAddress.getLocalHost(),2000); ds.send(dp);
        byte[] b1=new byte[1024];
        DatagramPacket dp1=new DatagramPacket(b1,b1.length);
        ds.receive(dp1);
        String result=new String(dp1.getData(),0,dp1.getLength());
        System.out.println(result);
    }
    catch(Exception e)
    {
        e.printStackTrace();
    }
    }
}

```

OUTPUT:





- c) Client-server based program to implement simple calculator operations addition, subtraction, multiplication and division.

Server Code:

```
package calserver;
import java.io.*;
import java.net.*;
public class CalServer {
    public static void main(String[] args) { try
    {
        DatagramSocket ds = new DatagramSocket(2000); byte[]
        b = new byte[1024];
        DatagramPacket dp = new DatagramPacket(b, b.length); ds.receive(dp);
        String str = new String(dp.getData(), 0, dp.getLength());
        System.out.println("Received data: " + str);

        // Parse the received data (e.g., "5 + 3") String[]
        parts = str.split(" ");
        double num1 = Double.parseDouble(parts[0]); String
        operation = parts[1];
        double num2 = Double.parseDouble(parts[2]);

        // Perform the operation
        double result = 0; switch
        (operation) {
            case "+":
                result = num1 + num2;
                break;
            case "-":
                result = num1 - num2;
                break;
            case "*":
                result = num1 * num2;
```

```

        break;
    case "/":
        if (num2 != 0) {
            result = num1 / num2;
        } else {
            throw new ArithmeticException("Division by zero");
        }
        break;
    default:
        throw new IllegalArgumentException("Invalid operation: " + operation);
}

// Send the result back to the client String
response = "Result: " + result; byte[] b1 =
response.getBytes();
DatagramPacket dp1 = new DatagramPacket(b1, b1.length, dp.getAddress(), dp.getPort());
ds.send(dp1);
} catch (Exception e) {
    e.printStackTrace();
}
}
}

```

Client Code:

```

package calclient;
import java.io.*;
import java.net.*;
public class CalClient {
    public static void main(String[] args) { try
    {
        DatagramSocket ds = new DatagramSocket(1000);
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

        // Input two numbers and an operation from the user
        System.out.println("Enter the first number: "); String
        num1 = br.readLine(); System.out.println("Enter the
        operation (+, -, *, /): "); String operation =
        br.readLine(); System.out.println("Enter the second
        number: "); String num2 = br.readLine();

        // Combine input into a single string
        String data = num1 + " " + operation + " " + num2; byte[]

```

```
b = data.getBytes();

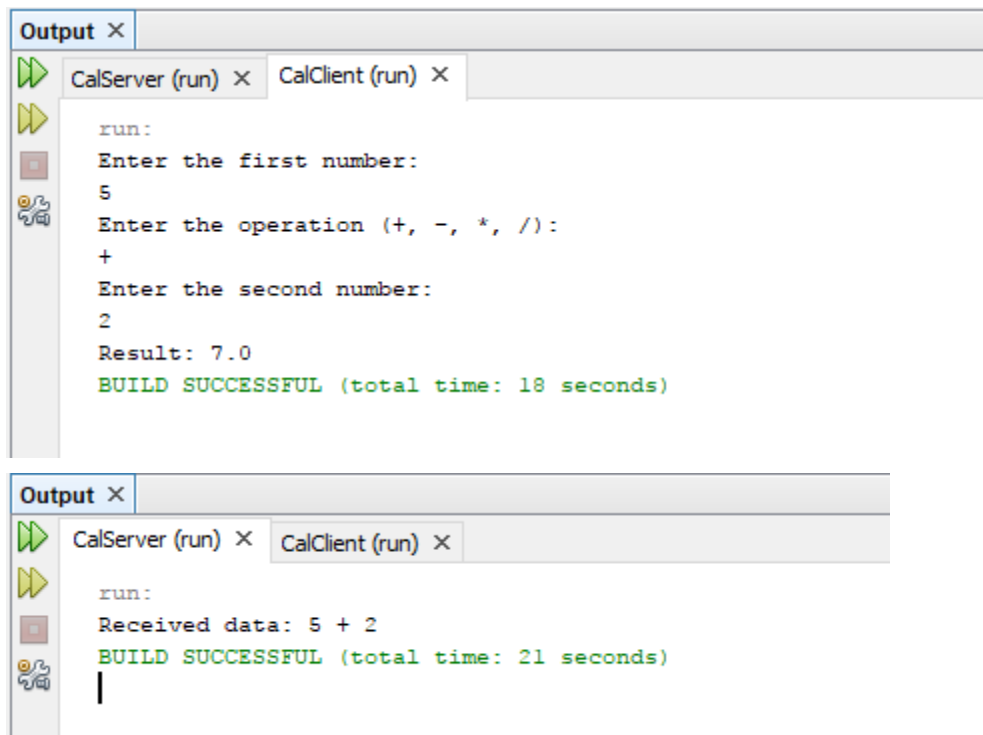
// Send the data to the server
DatagramPacket dp = new DatagramPacket(b, b.length,
InetAddress.getLocalHost(), 2000);
ds.send(dp);

// Receive the result from the server
byte[] b1 = new byte[1024];
DatagramPacket dp1 = new DatagramPacket(b1, b1.length);
ds.receive(dp1);
String response = new String(dp1.getData(), 0, dp1.getLength());
System.out.println(response);

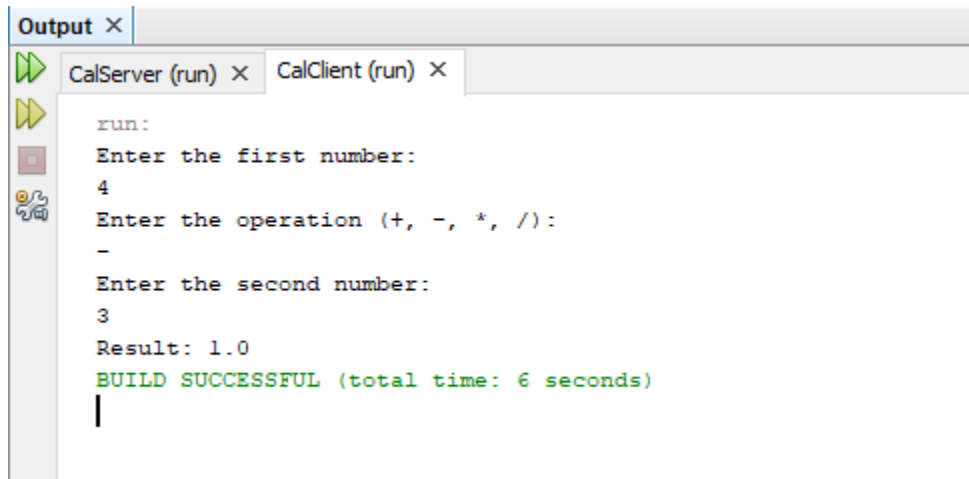
} catch (Exception e) {
    e.printStackTrace();
}
}
```

OUTPUT:

Addition:

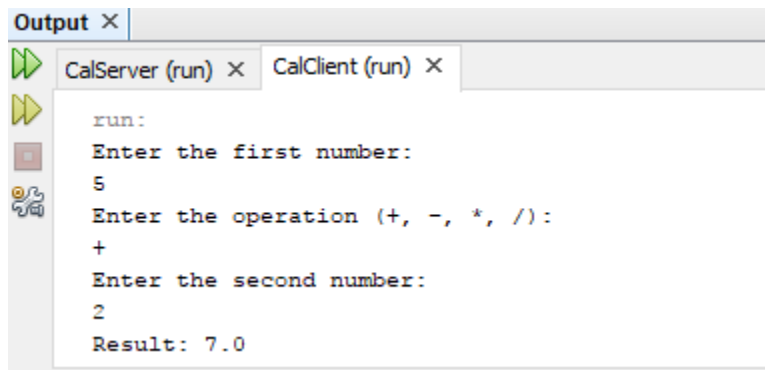


Subtraction:



The screenshot shows an IDE output window with two tabs: 'CalServer (run)' and 'CalClient (run)'. The 'CalClient (run)' tab is active, displaying the following text:

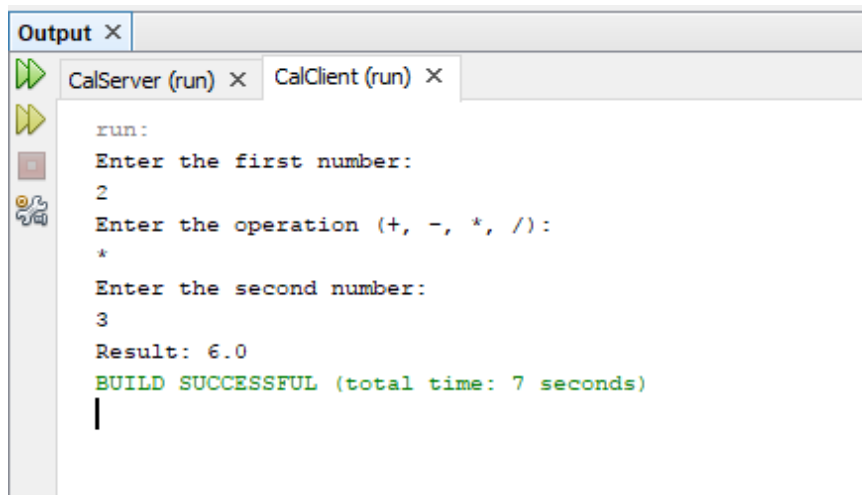
```
run:
Enter the first number:
4
Enter the operation (+, -, *, /):
-
Enter the second number:
3
Result: 1.0
BUILD SUCCESSFUL (total time: 6 seconds)
```



The screenshot shows an IDE output window with two tabs: 'CalServer (run)' and 'CalClient (run)'. The 'CalClient (run)' tab is active, displaying the following text:

```
run:
Enter the first number:
5
Enter the operation (+, -, *, /):
+
Enter the second number:
2
Result: 7.0
```

Multiplication:



The screenshot shows an IDE output window with two tabs: 'CalServer (run)' and 'CalClient (run)'. The 'CalClient (run)' tab is active, displaying the following text:

```
run:
Enter the first number:
2
Enter the operation (+, -, *, /):
*
Enter the second number:
3
Result: 6.0
BUILD SUCCESSFUL (total time: 7 seconds)
```



```

Output X
CalServer (run) X CalClient (run) X
run:
Received data: 2 * 3
BUILD SUCCESSFUL (total time: 11 seconds)

```

Division:

```

Output X
CalServer (run) X CalClient (run) X
run:
Enter the first number:
4
Enter the operation (+, -, *, /):
/
Enter the second number:
2
Result: 2.0
BUILD SUCCESSFUL (total time: 10 seconds)

```

```

Output X
CalServer (run) X CalClient (run) X
run:
Received data: 4 / 2
BUILD SUCCESSFUL (total time: 13 seconds)

```

- d) Client-server based program to find square, square root, cube and cube root of the entered numbers.

Server code:

```

package udpserveroperations;
import java.io.*;
import java.net.*;
import java.math.*;
public class UdpServerOperations { public
    static void main (String args []){ try {
        System.out.print("Server is running..."); DatagramSocket ds
        = new DatagramSocket(2000);

```

```

        byte b[]=new byte[1024];
        DatagramPacket dp = new DatagramPacket(b,b.length); ds.receive(dp);
        String str=new String (dp.getData(),0,dp.getLength());
        System.out.println(str);
        int a=Integer.parseInt(str);
        String s= new String();
        s="The Square: "+ a*a+" \nSquare _Root: "+Math.sqrt(a)
        +"\nCube:"+a*a*a+"\nCube_Root: "+Math.cbrt(a);
        byte b1[]= new byte[1024];
        b1=s.getBytes(); DatagramPacket
        dp1= new
        DatagramPacket(b1,b1.length,InetAddress.getLocalHost(),1000); ds.send(dp1);
    }
    catch (Exception e){
        e.printStackTrace();
    }
}
}

```

Client code:

```

package udpclientoperations;
import java.io.*;
import java.net.*;
public class UdpClientOperations { public
    static void main (String args[]){
        try {
            System.out.println("Client is running...");
            BufferedReader br = new BufferedReader (new InputStreamReader(System.in));
            System.out.println("Enter Number :");
            String num= br.readLine(); byte
            b[]= new byte[1024];
            b=num.getBytes();
            DatagramSocket ds= new DatagramSocket (1000); DatagramPacket dp
            = new
            DatagramPacket(b,b.length,InetAddress.getLocalHost(),2000); ds.send(dp);
            byte b1[]= new byte[1024];
            DatagramPacket dp1= new DatagramPacket(b1,b1.length);
            ds.receive(dp1);
            String r= new String (dp1.getData(),0,dp1.getLength());
            System.out.println(r);
        }
    }
}

```