**PRACTICAL 2**

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

**Practical 2A: A client server based program using UDP to find if the number entered is even or odd.**

**Code:**

**1.udpServerEO.java**

/\*Program which finds entered number is even or odd \*/

import java.io.\*;

import java.net.\*;

public class udpServerEO

{

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();

}

}

}

**2. udpClientEO.java**

import java.io.\*;

import java.net.\*;

public class udpClientEO

{

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();

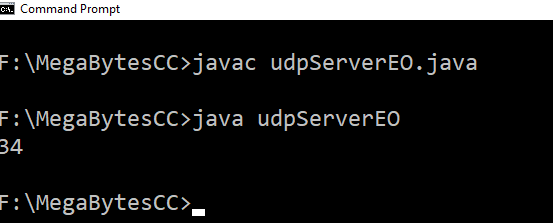
}

}

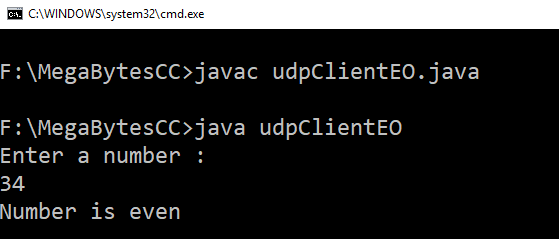
}

Output:

**Server**

****

**Client**

****

**Practical 2B: A client server based program using UDP to find the factorial of the entered number.**

**Code:**

1. **udpServerFact.java**

/\*Program which calculate factorial of a number\*/

import java.io.\*;

import java.net.\*;

public class udpServerFact {

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);

int f = 1, i;

String s= new String();

for(i=1;i<=a;i++) {

f=f\*i;

}

s=Integer.toString(f);

String str1 = "The Factorial of " + str + " is : " + f;

byte b1[] = new byte[1024];

b1 = str1.getBytes();

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

ds.send(dp1);

} catch(Exception e) {

e.printStackTrace();

}

}

}

1. **udpClientFact.java**

import java.io.\*;

import java.net.\*;

public class udpClientFact {

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) {

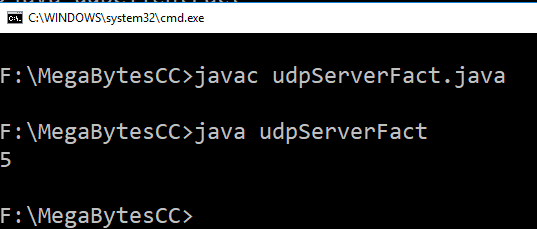
e.printStackTrace();

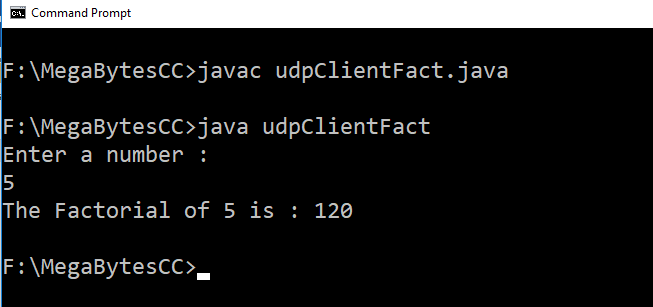
}

}

}

Output:





**Practical 2C: A program to implement simple calculator operations like addition, subtraction, multiplication and division.**

**Code:**

1. **RPCClient.java**

import java.io.\*;

import java.net.\*;

class RPCClient {

RPCClient()

{

try {

InetAddress ia = InetAddress.getLocalHost();

DatagramSocket ds = new DatagramSocket();

DatagramSocket ds1 = new DatagramSocket(1300);

System.out.println("\nRPC Client\n");

System.out.println("Enter method name and parameter like add 3 4\n");

while (true) {

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

String str = br.readLine();

byte b[] = str.getBytes();

DatagramPacket dp = new

DatagramPacket(b,b.length,ia,1200);

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

ds1.receive(dp);

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

System.out.println("\nResult = " + s + "\n");

}

} catch (Exception e) {

e.printStackTrace();

}

}

public static void main(String[] args) {

new RPCClient();

}

}

1. **RPCServer.java**

import java.util.\*;

import java.net.\*;

class RPCServer {

DatagramSocket ds;

DatagramPacket dp;

String str, methodName, result;

int val1, val2;

RPCServer() {

try {

ds = new DatagramSocket(1200);

byte b[] = new byte[4096];

while (true) {

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

ds.receive(dp);

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

if (str.equalsIgnoreCase("q")) {

System.exit(1);

} else {

StringTokenizer st = new StringTokenizer(str, " "); // Added a space as the delimiter

int i = 0;

while (st.hasMoreTokens()) {

String token = st.nextToken();

methodName = token;

val1 = Integer.parseInt(st.nextToken());

val2 = Integer.parseInt(st.nextToken());

}

}

System.out.println(str);

InetAddress ia = InetAddress.getLocalHost();

if (methodName.equalsIgnoreCase("add")) {

result = "" + add(val1, val2);

} else if (methodName.equalsIgnoreCase("sub")) {

result = "" + sub(val1, val2);

} else if (methodName.equalsIgnoreCase("mul")) {

result = "" + mul(val1, val2);

} else if (methodName.equalsIgnoreCase("div")) {

result = "" + div(val1, val2);

}

byte b1[] = result.getBytes();

DatagramSocket ds1 = new DatagramSocket();

DatagramPacket dp1 = new DatagramPacket(b1, b1.length, InetAddress.getLocalHost(), 1300);

System.out.println("result :" + result + "\n");

ds1.send(dp1);

ds1.close(); // Close the DatagramSocket after sending the response.

}

} catch (Exception e) {

e.printStackTrace();

}

}

public int add(int val1, int val2) {

return val1 + val2;

}

public int sub(int val1, int val2) { // Changed val3 and val4 to val1 and val2

return val1 - val2;

}

public int mul(int val1, int val2) { // Changed val3 and val4 to val1 and val2

return val1 \* val2;

}

public int div(int val1, int val2) { // Changed val3 and val4 to val1 and val2

if (val2 != 0) {

return val1 / val2;

} else {

return 0; // You should handle division by zero error.

}

}

public static void main(String[] args) {

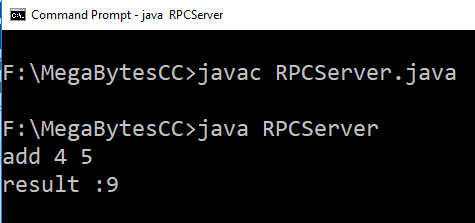
new RPCServer();

}

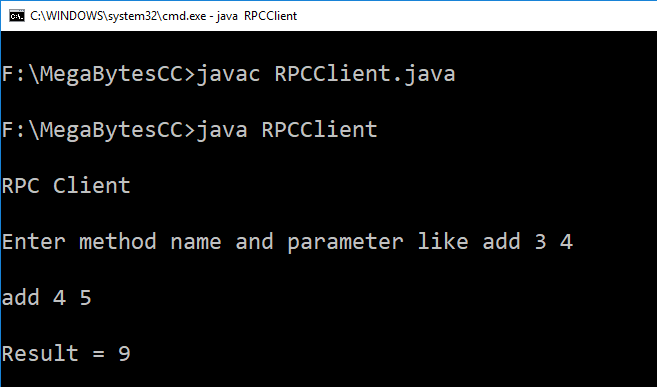
}

**Output:**

**Server**



**Client**



**Practical 2D: A program that finds the square, square root, cube and cube root of the entered number.**

**Code:**

1. **RPCNumServer.java**

import java.util.\*;

import java.net.\*;

import java.io.\*;

class RPCNumServer

{

DatagramSocket ds;

DatagramPacket dp;

String str,methodName,result;

int val;

RPCNumServer()

{

try

{

ds=new DatagramSocket(1200);

byte b[]=new byte[4096];

while(true)

{

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

ds.receive(dp);

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

if(str.equalsIgnoreCase("q")) {

System.exit(1);

}

else

{

StringTokenizer st = new StringTokenizer(str," ");

int i=0;

while(st.hasMoreTokens())

{

String token=st.nextToken();

methodName=token;

val = Integer.parseInt(st.nextToken());

}

}

System.out.println(str);

InetAddress ia = InetAddress.getLocalHost();

if(methodName.equalsIgnoreCase("square"))

{

result= "" + square(val);

}

else if(methodName.equalsIgnoreCase("squareroot"))

{

result= "" + squareroot(val);

}

else if(methodName.equalsIgnoreCase("cube"))

{

result= "" + cube(val);

}

else if(methodName.equalsIgnoreCase("cuberoot"))

{

result= "" + cuberoot(val);

}

byte b1[]=result.getBytes();

DatagramSocket ds1 = new DatagramSocket();

DatagramPacket dp1 = new

DatagramPacket(b1,b1.length,InetAddress.getLocalHost(), 1300); System.out.println("result : "+result+"\n"); ds1.send(dp1);

}

}

catch (Exception e)

{

e.printStackTrace();

}

}

public double square(int a) throws Exception

{

double ans;

ans = a\*a;

return ans;

}

public double squareroot(int a) throws Exception

{

double ans;

ans = Math.sqrt(a);

return ans;

}

public double cube(int a) throws Exception

{

double ans;

ans = a\*a\*a;

return ans;

}

public double cuberoot(int a) throws Exception

{

double ans;

ans = Math.cbrt(a);

return ans;

}

public static void main(String[] args)

{

new RPCNumServer();

}

}

1. **RPCNumClient.java**

import java.io.\*;

import java.net.\*;

class RPCNumClient

{

RPCNumClient()

{

try

{

InetAddress ia = InetAddress.getLocalHost();

DatagramSocket ds = new DatagramSocket();

DatagramSocket ds1 = new DatagramSocket(1300);

System.out.println("\nRPC Client\n");

System.out.println("1. Square of the number - square\n2. Square root of the number - squareroot\n3. Cube of the number - cube\n4. Cube root of the number - cuberoot");

System.out.println("Enter method name and the number\n");

while (true)

{

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

String str = br.readLine();

byte b[] = str.getBytes();

DatagramPacket dp = new

DatagramPacket(b,b.length,ia,1200);

ds.send(dp);

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

ds1.receive(dp);

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

System.out.println("\nResult = " + s + "\n");

}

}

catch (Exception e)

{

e.printStackTrace();

}

}

public static void main(String[] args)

{

new RPCNumClient();

}

}

**Output:**

