Practical No. 02

1. **Write a java program to implement a Server calculator using RPC concept. (Make use of datagram)**

Program:

Server.java

```
package rpc;

import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.InetAddress;
import java.util.StringTokenizer;



public class Server {
     private DatagramSocket udpSocket;
    private int port;

    public Server(int port) {
        this.port = port;
    }
    public static int addition(int num1,int num2)
    {
     return num1+num2;
    }
    public static int substraction(int num1,int num2)
    {
     return num1-num2;
    }
    public static int multiplication(int num1,int num2)
    {
     return num1*num2;
    }
    public static int division(int num1,int num2)
```

```
{
 return num1/num2;
}

private void listen() {
 try {
          DatagramSocket udpSocket = new DatagramSocket(port);
        System.out.println("Server started at " + InetAddress.getLocalHost());
        String msg;



          byte[] buf = new byte[1024];
          DatagramPacket packet = new DatagramPacket(buf, buf.length);

          // blocks until a packet is received
          udpSocket.receive(packet);
          msg = new String(packet.getData()).trim();

          StringTokenizer str=new StringTokenizer(msg,"-");
          int mthNo=Integer.parseInt(str.nextToken());
          int num1=Integer.parseInt(str.nextToken());
          int num2=Integer.parseInt(str.nextToken());
          int result;
          if(mthNo==1)
          {
                  result=addition(num1,num2);
                  msg="Addition:"+result;
          }
          if(mthNo==2)
          {
                  result=substraction(num1,num2);
                  msg="substraction:"+result;
          }
          if(mthNo==3)
```

```
                {
                        result=multiplication(num1,num2);
                        msg="multiplication:"+result;
                }
                if(mthNo==4)
                {
                        result=division(num1,num2);
                        msg="division:"+result;
                }

                System.out.println("Message from " + packet.getAddress().getHostAddress() +
    ": " + msg);

          }
            catch(Exception e) {
                System.out.println(e.getMessage());
            }
          finally {
                //udpSocket.close();
          }
        }

        public static void main(String[] args) {
            Server client = new Server(5000);
            client.listen();
        }

    }
```

Client.java

```
package rpc;

import java.io.BufferedReader;
import java.io.InputStreamReader;
```

```
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.InetAddress;
import java.util.Scanner;

public class Client {
        DatagramSocket udpSocket;
   InetAddress serverAddress;
   int port;
   Scanner scanner;

   public Client(int port) {
      this.port = port;
   }

   public void sendReq() {
      String in;
      try {
         udpSocket = new DatagramSocket();
         InetAddress host = InetAddress.getLocalHost();
               serverAddress = InetAddress.getByName(host.getHostName());

               BufferedReader keyRead = new BufferedReader(new InputStreamReader(System.in));
               System.out.println("UDP Client started at " + InetAddress.getLocalHost());
               String paramlist="";
               System.out.println("Enter
Method:\n1.Addition:\n2.Subtraction\n3.Multiplication\n4.Devision");
               in = keyRead.readLine();
               paramlist=paramlist+in+"-";
               System.out.println("Enter Number 1:");
               in = keyRead.readLine();
```

```
                paramlist=paramlist+in+"-";
                System.out.println("Enter Number 2:");
                in = keyRead.readLine();
                paramlist=paramlist+in;
                DatagramPacket p = new DatagramPacket(paramlist.getBytes(),
paramlist.getBytes().length, serverAddress, port);
                udpSocket.send(p);


        }
        catch(Exception e) {
            System.out.println(e.getMessage());
                    }
    }

    public static void main(String[] args) {
        Client sender = new Client(5000);
        sender.sendReq();
    }



}
```

Output:

```
<terminated> Client [Java Application] C:\Program Files\Java\jdk-18.0.2.1\bin\javaw.exe ( 06-Dec-2022, 11:32:06 am -11:33:09 am ) [pid: 4176]
UDP Client started
Enter Method:
1.Addition:
2.Subtraction
3.Multiplication
4.Division
1
Enter Number 1:
5
Enter Number 2:
9
```

```
<terminated> Server [Java Application] C:\Program Files\Java\jdk-18.0.2.1\bin\javaw.exe (06-Dec-2022, 11:29:36 am - 11:33:09 am ) [pid: 13976]
Server started
Message from 127.0.0.1: Addition:14
```

```
<terminated> Client [Java Application] C:\Program Files\Java\jdk-18.0.2.1\bin\javaw.exe (` 06-Dec-2022, 11:29:36 am - 11:33:09 am) [pid: 19440]
UDP Client started
Enter Method:
1.Addition:
2.Subtraction
3.Multiplication
4.Division
3
Enter Number 1:
9
Enter Number 2:
3
```

```
<terminated> Server [Java Application] C:\Program Files\Java\jdk-18.0.2.1\bin\javaw.exe (`06-Dec-2022, 11:29:36 am - 11:33:09 am ) [pid: 13352]
Server started
Message from 127.0.0.1: multiplication:27
```

2) Write a java to implement a Date Time Server using RPC concept. (Make use of datagram)

Program:

Server.java

```
package rpcdatetime;


import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.InetAddress;
import java.time.LocalDateTime;


public class Server {
        private DatagramSocket udpSocket;
   private int port;


   public Server(int port) {
      this.port = port;
   }
   public static LocalDateTime date()
   {
        return java.time.LocalDateTime.now();
```

```
    }

    private void listen() {
        try {
                DatagramSocket udpSocket = new DatagramSocket(port);
            System.out.println("Server started at " + InetAddress.getLocalHost());
            LocalDateTime msg;


                byte[] buf = new byte[1024];
                DatagramPacket packet = new DatagramPacket(buf, buf.length);

                // blocks until a packet is received
                udpSocket.receive(packet);

                msg=date();
                System.out.println("Message from " + packet.getAddress().getHostAddress() +
": " + msg);


        }
       catch(Exception e) {
          System.out.println(e.getMessage());
       }
          finally {
                //udpSocket.close();
          }
    }

    public static void main(String[] args) {
       Server client = new Server(5000);
       client.listen();
    }
```

```
}
```

Client.java

```
package rpcdatetime;

import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.InetAddress;
import java.time.LocalDateTime;
import java.util.Scanner;

public class Client {
        DatagramSocket udpSocket;
   InetAddress serverAddress;
   int port;
   Scanner scanner;

   public Client(int port) {
      this.port = port;
   }

   public void sendReq() {
      String in;
      try {
         udpSocket = new DatagramSocket();
         InetAddress host = InetAddress.getLocalHost();
               serverAddress = InetAddress.getByName(host.getHostName());

               BufferedReader keyRead = new BufferedReader(new
InputStreamReader(System.in));
```

```
            System.out.println("UDP Client started at " + InetAddress.getLocalHost());
            String paramlist="";
            DatagramPacket p = new DatagramPacket(paramlist.getBytes(),
paramlist.getBytes().length, serverAddress, port);
            udpSocket.send(p);


        }
        catch(Exception e) {
           System.out.println(e.getMessage());
                    }
    }

    public static void main(String[] args) {
        Client sender = new Client(5000);
        sender.sendReq();
    }



}
```

Output: