RAMNIRANJAN JHUNJHUNWALA COLLEGE
GHATKOPAR (W), MUMBAI - 400 086

DEPARTMENT OF INFORMATION TECHNOLOGY
2020 - 2021

M.Sc.(I.T.) SEM I
Distributed System

Name : Sneha Ramchandra Pawar.
Roll No. : 11

Hindi Vidya Prachar Samiti's

RAMNIRANJAN JHUNJHUNWALA COLLEGE

(AUTONOMOUS)

Opposite Ghatkopar Railway Station, Ghatkopar West, Mumbai-400086

CERTIFICATE

This is to certify that Miss. **Sneha Ramchandra Pawar** with Roll No. **11** has successfully completed the necessary course of experiments in the subject of **Distributed Systems** during the academic year **2019 – 2020** complying with the requirements of **RAMNIRANJAN JHUNJHUNWALA COLLEGE OF ARTS, SCIENCE AND COMMERCE**, for the course of **M.Sc. (IT)** semester -I.

____________________ ____________________

Internal Examiner External Examiner

______________________ ________________

Head of Department College Seal

INDEX

Practical 1

Write a program for implementing Client Server Communication Model.

Client Server communication model:
Client–server model is a distributed application structure that partitions tasks or workloads between the providers of a resource or service, called servers, and service requesters, called clients. Often clients and servers communicate over a computer network on separate hardware, but both client and server may reside in the same system. A server host runs one or more server programs, which share their resources with clients. A client does not share any of its resources, but it requests content or service from a server. Clients therefore initiate communication sessions with servers, which await incoming requests. Examples of computer applications that use the client–server model are Email, network printing, and the World Wide Web.

The client-server characteristic describes the relationship of cooperating programs in an application. The server component provides a function or service to one or many clients, which initiate requests for such services. Servers are classified by the services they provide. For example, a web server serves web pages and a file server serves computer files. A shared resource may be any of the server computer's software and electronic components, from programs and data to processors and storage devices. The sharing of resources of a server constitutes a service.

Q. 1. A) Write a program for implementing Client Server Communication Model. (Request a server to tell whether the entered number is prime or not.)

CODE

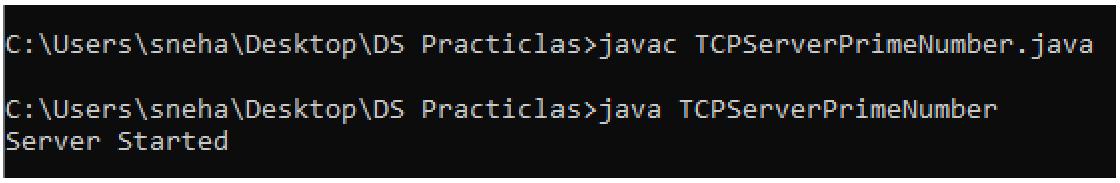
TCPServerPrimeNumber.java

```
import java.net.*;
import java.io.*;

public class TCPServerPrimeNumber
{
   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());
         int y = x / 2;
         if (x == 1)
         {
            otc.writeUTF(x + "is not a prime number");
            System.exit(0);               else
         {
            int flag = 0;
            for (int i = 2; i <= y; i++)
            {
                if (x % i == 0)
                {
                    otc.writeUTF(x + " is not prime number");
            }

                    flag = 1;
                    break;

                }
                if (flag == 0) { otc.writeUTF(x + " is prime number"); }

              }

            }
        }

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

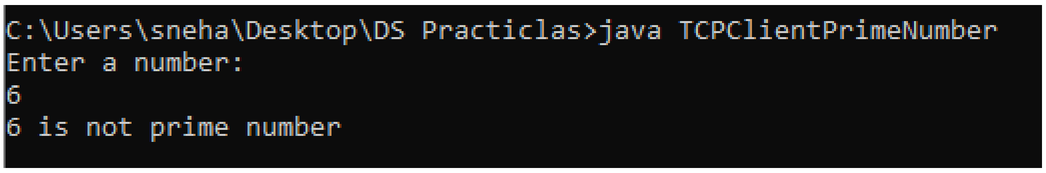
TCPClientPrimeNumber.java

```
import java.net.*;
import java.io.*;

public class TCPClientPrimeNumber
{
    public static void main(String args[])
    {
        try
        {
            Socket cs = new Socket("LocalHost", 8001);
            BufferedReader input = new BufferedReader(new InputStreamReader(System.in));
            System.out.println("Enter a number: ");
            int x = Integer.parseInt(input.readLine());
            DataOutputStream out = new DataOutputStream(cs.getOutputStream());
            out.writeInt(x);
            DataInputStream in = new DataInputStream(cs.getInputStream());
            System.out.println(in.readUTF());
            cs.close();
        }
        catch (Exception e)
        {
            System.out.println(e.toString());
        }
    }
}
```

OUTPUT :-



```
C:\Users\sneha\Desktop\DS Practiclas>javac TCPServerPrimeNumber.java

C:\Users\sneha\Desktop\DS Practiclas>java TCPServerPrimeNumber
Server Started
```



```
C:\Users\sneha\Desktop\DS Practiclas>java TCPClientPrimeNumber
Enter a number:
6
6 is not prime number
```

Q. 1. B) Write a program for implementing Client Server Communication Model. (A Client Server TCP based Chatting Application.)

CODE

TCPServerChat.java

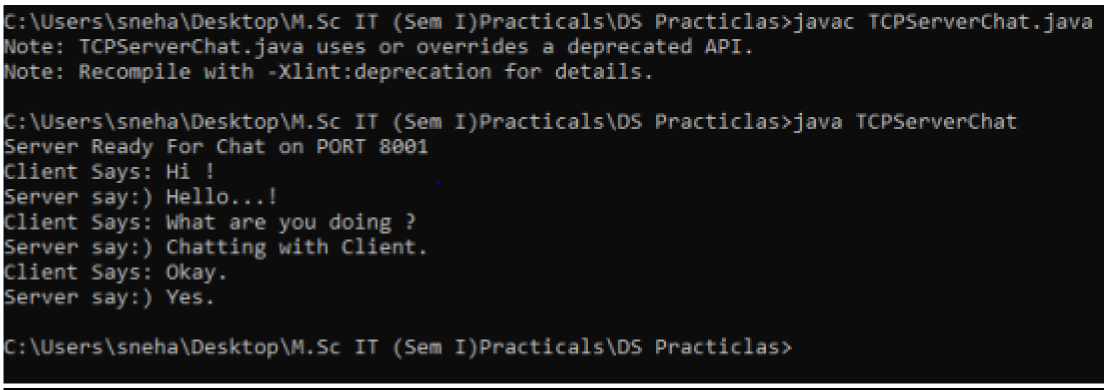
```
import java.net.*;
import java.io.*;
public class TCPServerChat{
 public static void main(String args[]){
  try{
      final int PORT=8001;
      ServerSocket serverSocket=new ServerSocket(PORT);
      System.out.println("Server Ready For Chat on PORT " + PORT);

      Socket socket=serverSocket.accept();
      BufferedReader inputBReader = new BufferedReader (new InputStreamReader(System.in));
          DataOutputStream outputStream = new
 DataOutputStream(socket.getOutputStream());
          DataInputStream inputStream = new
 DataInputStream(socket.getInputStream());
          String receive;
          String send;

          while ((receive = inputStream.readLine()) != null) {
               if(receive.toLowerCase().equals("stop"))
              break;
           System.out.println("Client Says: " + receive);
           System.out.print("Server say:) ");
           send =inputBReader.readLine();
           outputStream.writeBytes(send + "\n");
           }
            inputBReader.close();
            inputStream.close();
            outputStream.close();
            serverSocket.close();
}
 catch (Exception e)
{
  System.out.println(e.toString());
}
}
```

```
}
```

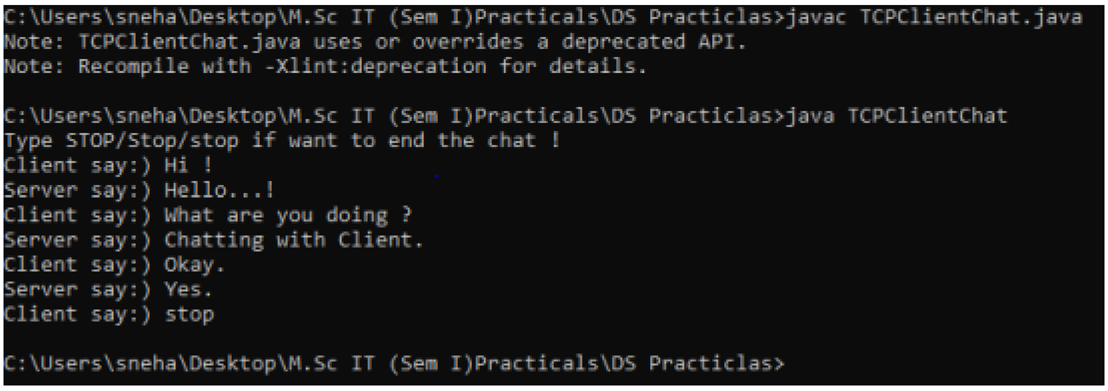
TCPClientChat .java

```
import java.net.*;
import java.io.*;

public class TCPClientChat {
      public static void main(String args[]){
           try {
                 final int PORT=8001;
                 final String HOST = "LocalHost";
                 Socket clientSocket = new Socket(HOST,PORT);
                 BufferedReader       inpBufferedReader       =       new       BufferedReader(new
InputStreamReader(System.in));
                 DataOutputStream outputStream = new DataOutputStream(clientSocket.getOutputStream());
                 DataInputStream inputStream = new DataInputStream(clientSocket.getInputStream());
                 String send;
                  System.out.println("Type STOP/Stop/stop if want to end the chat !");
                  System.out.print("Client say:) ");
                  while ((send =inpBufferedReader.readLine()) !=null) {
                  outputStream.writeBytes(send + "\n");
                  if (send.toLowerCase().equals("stop"))
                   break;
                  System.out.println("Server say:) " +
                    inputStream.readLine());
                       System.out.print("Client say:) ") ;
                    }
                     inpBufferedReader.close();
                     inputStream.close();
                     outputStream.close();
                     clientSocket.close();
                   }
                     catch(Exception e)
                           {
                              System.out.println(e.toString());
                      }
                       }
                        }
```

OUTPUT :-



```
C:\Users\sneha\Desktop\M.Sc IT (Sem I)Practicals\DS Practiclas>javac TCPServerChat.java
Note: TCPServerChat.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.

C:\Users\sneha\Desktop\M.Sc IT (Sem I)Practicals\DS Practiclas>java TCPServerChat
Server Ready For Chat on PORT 8001
Client Says: Hi !
Server say:) Hello...!
Client Says: What are you doing ?
Server say:) Chatting with Client.
Client Says: Okay.
Server say:) Yes.

C:\Users\sneha\Desktop\M.Sc IT (Sem I)Practicals\DS Practiclas>
```



```
C:\Users\sneha\Desktop\M.Sc IT (Sem I)Practicals\DS Practiclas>javac TCPClientChat.java
Note: TCPClientChat.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.

C:\Users\sneha\Desktop\M.Sc IT (Sem I)Practicals\DS Practiclas>java TCPClientChat
Type STOP/Stop/stop if want to end the chat !
Client say:) Hi !
Server say:) Hello...!
Client say:) What are you doing ?
Server say:) Chatting with Client.
Client say:) Okay.
Server say:) Yes.
Client say:) stop

C:\Users\sneha\Desktop\M.Sc IT (Sem I)Practicals\DS Practiclas>
```

Practical 2

Write a program to show the object communication using RMI.

RMI (Remote Method Invocation):

The **RMI** (Remote Method Invocation) is an API that provides a mechanism to create distributed application in java. The RMI allows an object to invoke methods on an object running in another JVM.

The RMI provides remote communication between the applications using two objects *stub* and *skeleton*.

RMI uses stub and skeleton object for communication with the remote object.
A **remote object** is an object whose method can be invoked from another JVM.

stub

The stub is an object, acts as a gateway for the client side. All the outgoing requests are routed through it. It resides at the client side and represents the remote object. When the caller invokes method on the stub object, it does the following tasks:

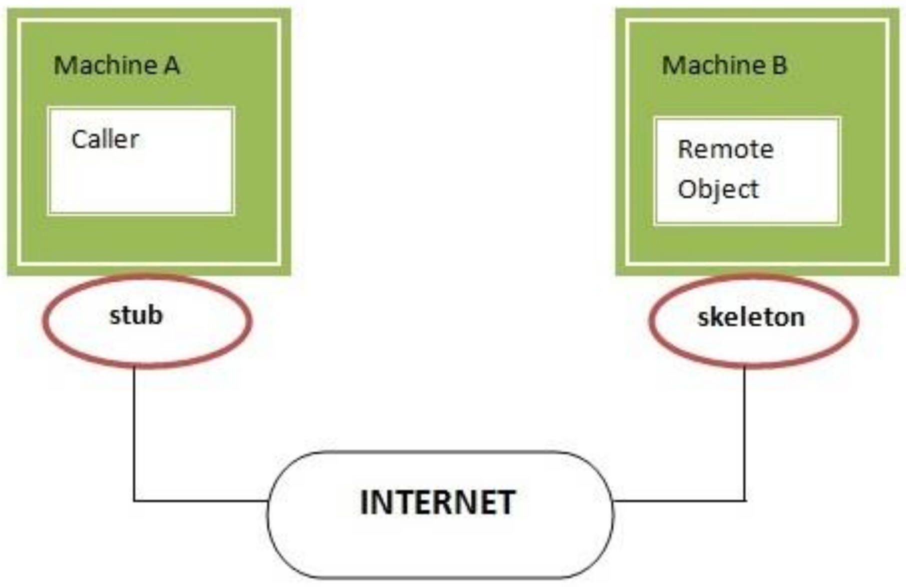
1. It initiates a connection with remote Virtual Machine (JVM),
2. It writes and transmits (marshals) the parameters to the remote Virtual Machine (JVM),
3. It waits for the result
4. It reads (unmarshals) the return value or exception, and
5. It finally, returns the value to the caller.

skeleton

The skeleton is an object, acts as a gateway for the server side object. All the incoming requests are routed through it. When the skeleton receives the incoming request, it does the following tasks:

1. It reads the parameter for the remote method
2. It invokes the method on the actual remote object, and
3. It writes and transmits (marshals) the result to the caller.

In the Java 2 SDK, an stub protocol was introduced that eliminates the need for skeletons.



If any application performs these tasks, it can be distributed application.

1. The application need to locate the remote method
2. It need to provide the communication with the remote objects, and
3. The application need to load the class definitions for the objects.

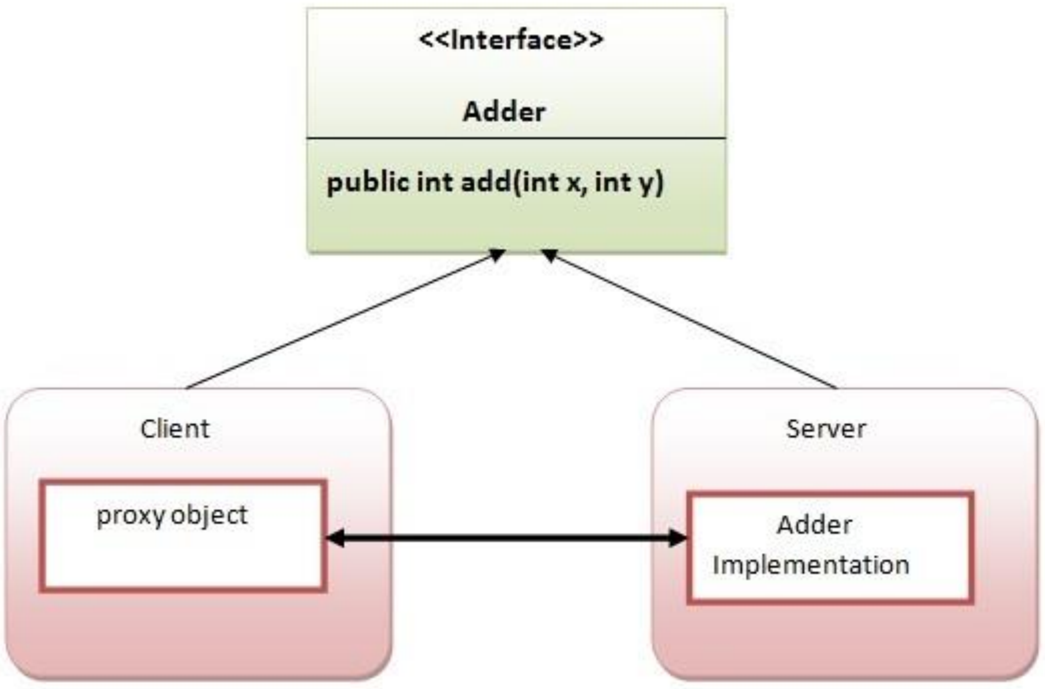
The RMI application have all these features, so it is called the distributed application.

The is given the 6 steps to write the RMI program.

1. Create the remote interface
2. Provide the implementation of the remote interface
3. Compile the implementation class and create the stub and skeleton objects using the rmic tool
4. Start the registry service by rmiregistry tool
5. Create and start the remote application
6. Create and start the client application

RMI Example

In this example, we have followed all the 6 steps to create and run the rmi application. The client application need only two files, remote interface and client application. In the rmi application, both client and server interacts with the remote interface. The client application invokes methods on the proxy object, RMI sends the request to the remote JVM. The return value is sent back to the proxy object and then to the client application.



Q. **2. A) Write a program to show the object communication using RMI. (RMI Based application program to display current date and time).**

CODE:

InterDate.java

```
import java.rmi.*;
public interface InterDate extends Remote
{
public String display() throws RemoteException;
}
```

ServerDate.java

```
import java.rmi.*;
import java.rmi.registry.LocateRegistry;
import java.rmi.registry.Registry;
import java.rmi.server.*;
import java.util.*;
public class ServerDate extends UnicastRemoteObject implements InterDate{
    public ServerDate() throws RemoteException
            {
             }
    public String display() throws RemoteException
```

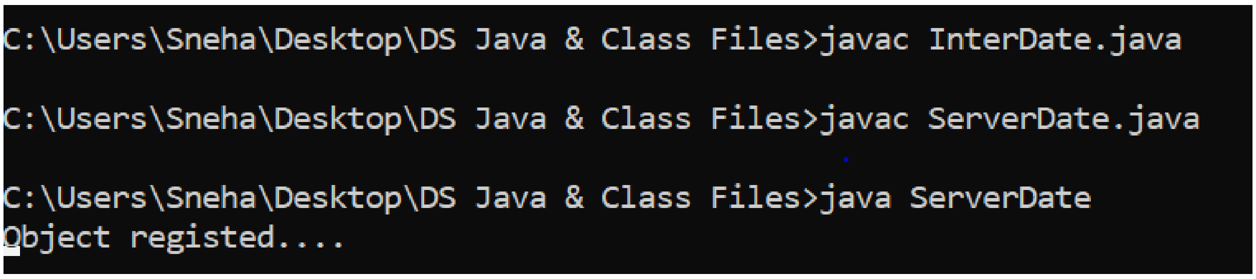
```
        {
         String str = "";
         Date d = new Date();
         str = d.toString();
         return str;
      }

      public static void main(String args[]) throws RemoteException
      {
         try
         {
            ServerDate s1= new ServerDate();
             Registry reg = LocateRegistry.createRegistry(5678);
            reg.rebind("DS",s1);
            System.out.println("Object registed....");
         }
         catch(RemoteException e){
             System.out.println("exception"+e);
         }
      }
}
```

<u>ClientDate.java</u>

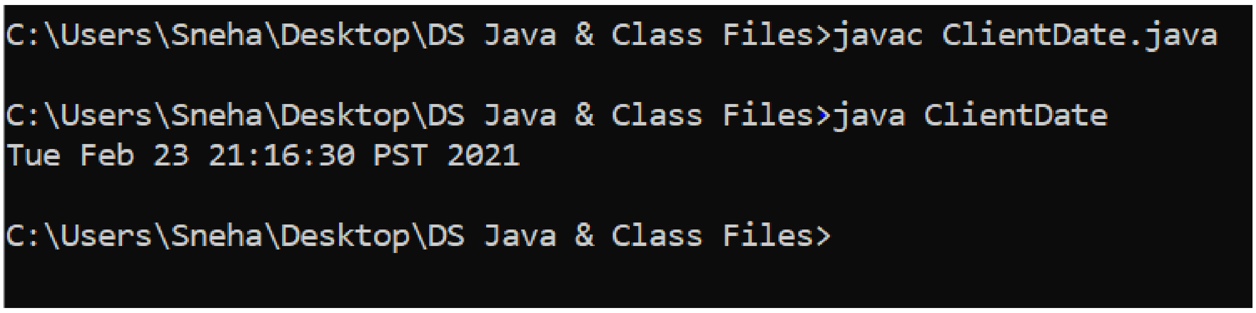
```
import java.rmi.*;
import java.rmi.registry.LocateRegistry;
import java.rmi.registry.Registry;
import java.rmi.server.*;
import java.util.*;
public class ServerDate extends UnicastRemoteObject implements InterDate{
   public ServerDate() throws RemoteException
             {
              }
   public String display() throws RemoteException
        {
         String str = "";
         Date d = new Date();
         str = d.toString();
         return str;
      }

      public static void main(String args[]) throws RemoteException
      {
         try
         {
            ServerDate s1= new ServerDate();
```

```
                Registry reg = LocateRegistry.createRegistry(5678);
               reg.rebind("DS",s1);
               System.out.println("Object registed....");
            }
            catch(RemoteException e){
                System.out.println("exception"+e);
            }
      }
}
```

OUTPUT:



```
C:\Users\Sneha\Desktop\DS Java & Class Files>javac InterDate.java

C:\Users\Sneha\Desktop\DS Java & Class Files>javac ServerDate.java

C:\Users\Sneha\Desktop\DS Java & Class Files>java ServerDate
Object registed....
```



```
C:\Users\Sneha\Desktop\DS Java & Class Files>javac ClientDate.java

C:\Users\Sneha\Desktop\DS Java & Class Files>java ClientDate
Tue Feb 23 21:16:30 PST 2021

C:\Users\Sneha\Desktop\DS Java & Class Files>
```

Q. 2. B) Write a program to show the object communication using RMI. (RMI based application program that converts digits to words, e.g., 123 will be converted to one two three).

CODE

InterConvert.java

```
import java.rmi.*;
public interface InterConvert extends Remote
{
            public String display(String n) throws RemoteException;
}
```

ServerConvert.java

```
import java.rmi.*;
import java.rmi.registry.LocateRegistry;
import java.rmi.registry.Registry;
import java.rmi.server.*;
import java.util.*;
public class ServerConvert extends UnicastRemoteObject implements InterConvert{
     public ServerConvert() throws RemoteException
     {
     }
     public String display(String n) throws RemoteException
     {
          String str1 = "";
          for(int i=0;i<n.length();i++)
               {
                                   int ascii=n.charAt(i);
                                   switch(ascii)
                                   {
                                                       case 48:
                                                       str1 +="ZERO";
                                                       break;
                                                       case 49:
                                                       str1 +="ONE";
                                                       break;
                                                       case 50:
                                                       str1 +="TWO";
                                                       break;
                                                       case 51:
                                                       str1 +="THREE";
                                                       break;
                                                       case 52:
                                                       str1 +="FOUR";
                                                       break;
                                                       case 53:
                                                       str1 +="FIVE";
                                                       break;
                                                       case 54:
                                                       str1 +="SIX";
                                                       break;
                                                       case 55:
                                                       str1 +="SEVEN";
                                                       break;
                                                       case 56:
                                                       str1 +="EIGHT";
                                                       break;
```

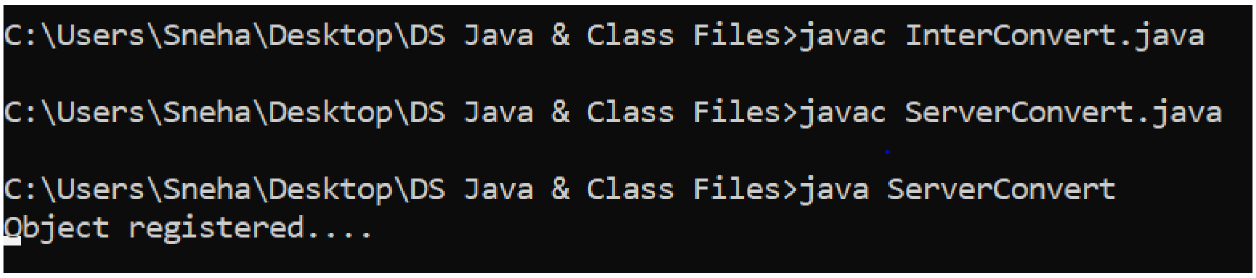
```
                                        case 57:
                                        str1 +="NINE";
                                        break;
                }
               }
          return str1;
     }
     public static void main(String args[]) throws RemoteException{
          try{
               ServerConvert s1= new ServerConvert();
               Registry reg = LocateRegistry.createRegistry(5678);
               reg.rebind("DS",s1);
               System.out.println("Object registered....");
          }
          catch(RemoteException e){
                System.out.println("exception"+e);
          }
     }
}
```

ClientConver.java

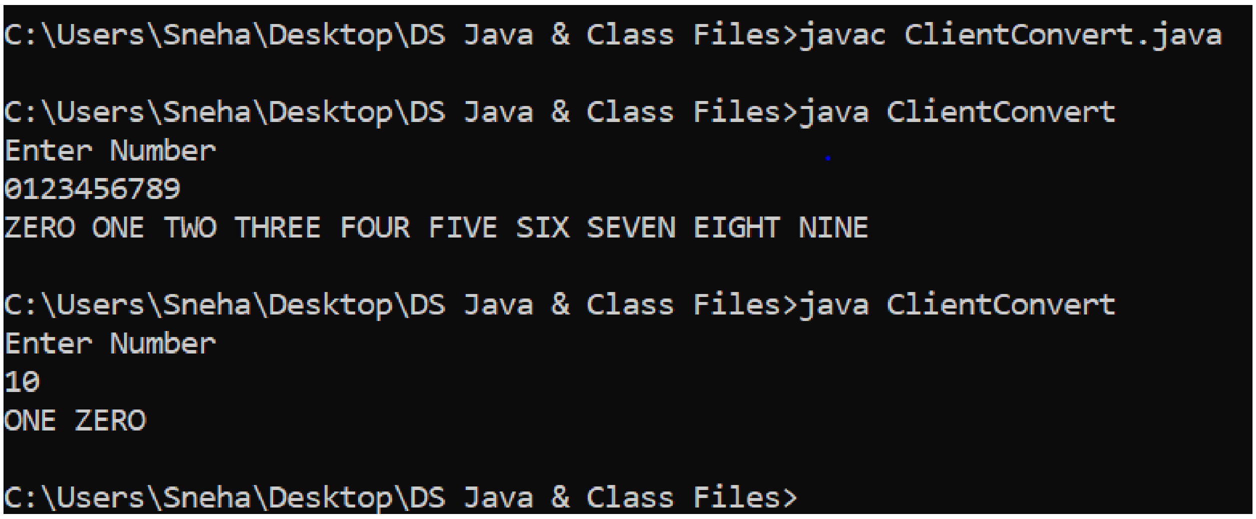
```
import java.rmi.*;
import java.io.*;
import java.rmi.registry.LocateRegistry;
import java.rmi.registry.Registry;
import java.rmi.server.*;
import java.util.*;
import java.io.*;
import java.rmi.*;
public class ClientConvert{
     public static void main(String args[]) throws RemoteException{
          ClientConvert c1 = new ClientConvert();
          c1.connectRemote();
     }
     private void connectRemote() throws RemoteException{
          try{
BufferedReader info=new BufferedReader(new InputStreamReader(System.in));
                         System.out.println("Enter Number");
                        String s1=info.readLine();
                         String s2;
               Registry reg = LocateRegistry.getRegistry("localhost",5678);
                        InterConvert h1 = (InterConvert)reg.lookup("DS");
                        s2 = h1.display(s1);
```

```
                    System.out.println(s2);
                }
                catch(Exception e){
                System.out.println(e);
            }
        }
    }
```

OUTPUT:



```
C:\Users\Sneha\Desktop\DS Java & Class Files>javac InterConvert.java

C:\Users\Sneha\Desktop\DS Java & Class Files>javac ServerConvert.java

C:\Users\Sneha\Desktop\DS Java & Class Files>java ServerConvert
Object registered....
```

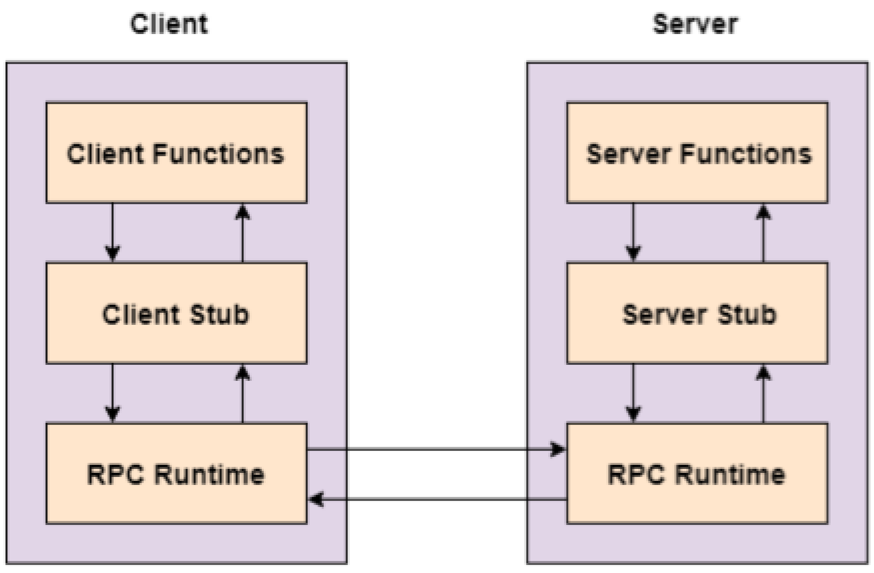


```
C:\Users\Sneha\Desktop\DS Java & Class Files>javac ClientConvert.java

C:\Users\Sneha\Desktop\DS Java & Class Files>java ClientConvert
Enter Number
0123456789
ZERO ONE TWO THREE FOUR FIVE SIX SEVEN EIGHT NINE

C:\Users\Sneha\Desktop\DS Java & Class Files>java ClientConvert
Enter Number
10
ONE ZERO

C:\Users\Sneha\Desktop\DS Java & Class Files>
```

Practical 3

Show the implementation of Remote Procedure Call (RPC).

Remote Procedure Call (RPC):
A remote procedure call is an inter process communication technique that is used for clientserver-based applications. It is also known as a subroutine call or a function call.
A client has a request message that the RPC translates and sends to the server. This request may be a procedure or a function call to a remote server. When the server receives the request, it sends the required response back to the client. The client is blocked while the server is processing the call and only resumed execution after the server is finished.
The sequence of events in a remote procedure call are given as follows:
• The client stub is called by the client
• The client stub makes a system call to send the message to the server and puts the parameters in the message.
• The message is sent from the client to the server by the client's operating system
• The message is passed to the server stub by the server operating system.
• The parameters are removed from the message by the server stub. • Then, the server procedure is called by the server stub.
A diagram that demonstrates this is as follows:



The following steps take place during a RPC:
1. A client invokes a client stub procedure, passing parameters in the usual way. The client stub resides within the client's own address space.
2. The client stub marshalls(pack) the parameters into a message. Marshalling includes converting the representation of the parameters into a standard format, and copying each parameter into the message.
3. The client stub passes the message to the transport layer, which sends it to the remote server machine.
4. On the server, the transport layer passes the message to a server stub, which demarshalls(unpack) the parameters and calls the desired server routine using the regular procedure call mechanism.

Q. 3 A) Show the implementation of Remote Procedure Call (RPC). (Find Addition, Subtraction, Multiplication, Division.)

<u>CODE</u>

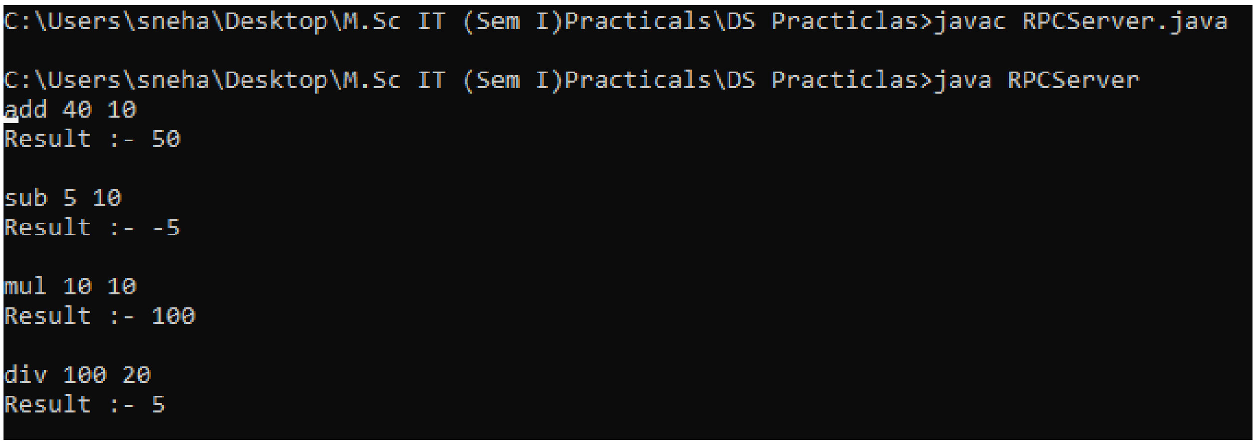
<u>RPCServer.java</u>

```
import java.io.*;
import java.net.*;
import java.util.StringTokenizer;

final 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," ");
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);
}
}
catch(Exception e)
{
e.printStackTrace();
}
}
public int add(int val1,int val2)
{
return val1+val2;
}
public int sub(int val1,int val2)
{
return val1-val2;
}
public int mul(int val1,int val2)
{
return val1*val2;
}
public int div(int val1,int val2)
{
return val1/val2;
}
public static void main(String args[])
{
new RPCServer();
}
}
```

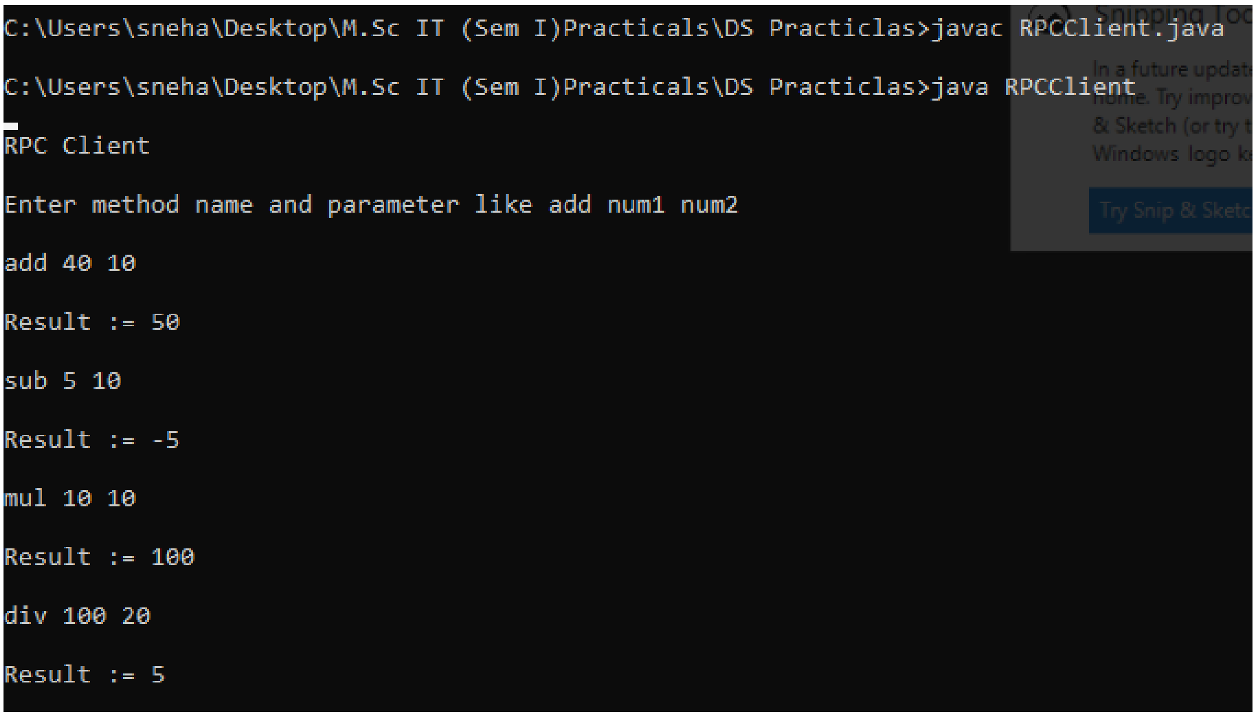
<u>RPCClient.java</u>

```
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 num1 num2\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();
}
}
```

OUTPUT :-



```
C:\Users\sneha\Desktop\M.Sc IT (Sem I)Practicals\DS Practiclas>javac RPCServer.java

C:\Users\sneha\Desktop\M.Sc IT (Sem I)Practicals\DS Practiclas>java RPCServer
add 40 10
Result :- 50

sub 5 10
Result :- -5

mul 10 10
Result :- 100

div 100 20
Result :- 5
```



```
C:\Users\sneha\Desktop\M.Sc IT (Sem I)Practicals\DS Practiclas>javac RPCClient.java

C:\Users\sneha\Desktop\M.Sc IT (Sem I)Practicals\DS Practiclas>java RPCClient

RPC Client

Enter method name and parameter like add num1 num2

add 40 10

Result := 50

sub 5 10

Result := -5

mul 10 10

Result := 100

div 100 20

Result := 5
```

Q. 3 B) Show the implementation of Remote Procedure Call(RPC). (Find Square, Square Root, Cube, Cube Root of a number.)

CODE

RPCServer1.java

```
import java.io.*;
import java.net.*;
import java.util.StringTokenizer;

final public class RPCServer1
{
DatagramSocket ds;
DatagramPacket dp;
String str,methodname,result;
int val1;
RPCServer1()
{
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
{
```

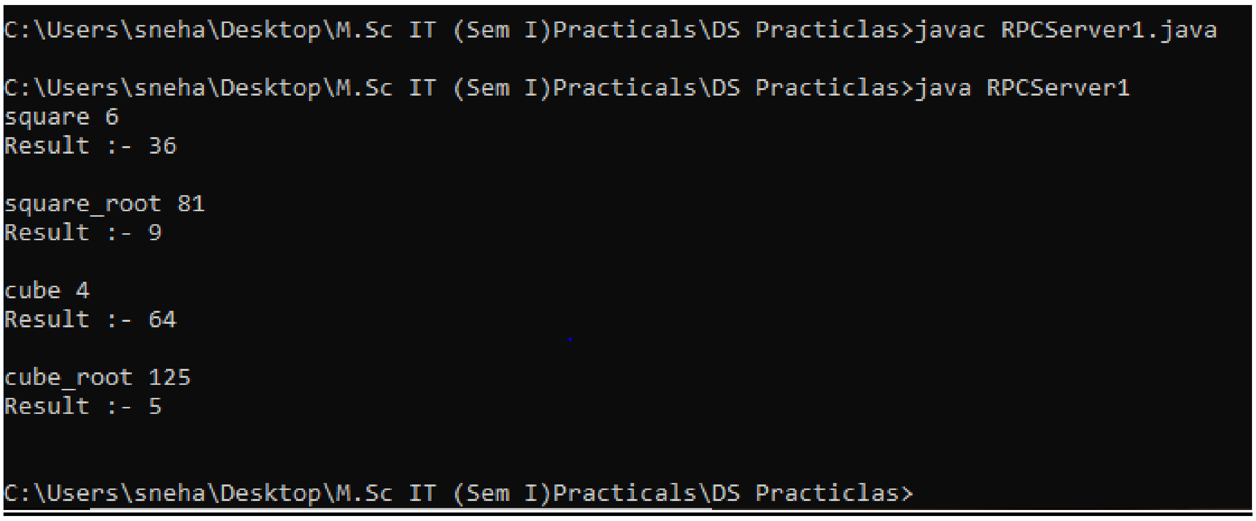
```
StringTokenizer st=new StringTokenizer(str," ");
int i=0;
while(st.hasMoreTokens())
{
String token=st.nextToken();
methodname=token;
val1 = Integer.parseInt(st.nextToken());
//val2 = Integer.parseInt(st.nextToken());
}
}
System.out.println(str);
InetAddress ia = InetAddress.getLocalHost();
if(methodname.equalsIgnoreCase("square"))
{
result = "" +square(val1);
}
else if(methodname.equalsIgnoreCase("square_Root"))
{
result = "" +square_Root(val1);
}
else if(methodname.equalsIgnoreCase("cube"))
{
result = ""   +cube(val1);
}
else if(methodname.equalsIgnoreCase("cube_Root"))
{
result = "" +cube_Root(val1);
}
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();
}
}
catch(Exception e)
{
e.printStackTrace();
}
}
public int square(int val1)
{
return (int) Math.pow(val1,2);
}
public int square_Root(int val1)
{
return   (int) Math.sqrt(val1);
}
public int cube(int val1)
{
return (int)Math.pow(val1,3);
}
public int cube_Root(int val1)
{
return (int) Math.cbrt(val1);
}
public static void main(String args[])
```

```
{
new RPCServer1();
}
}
```

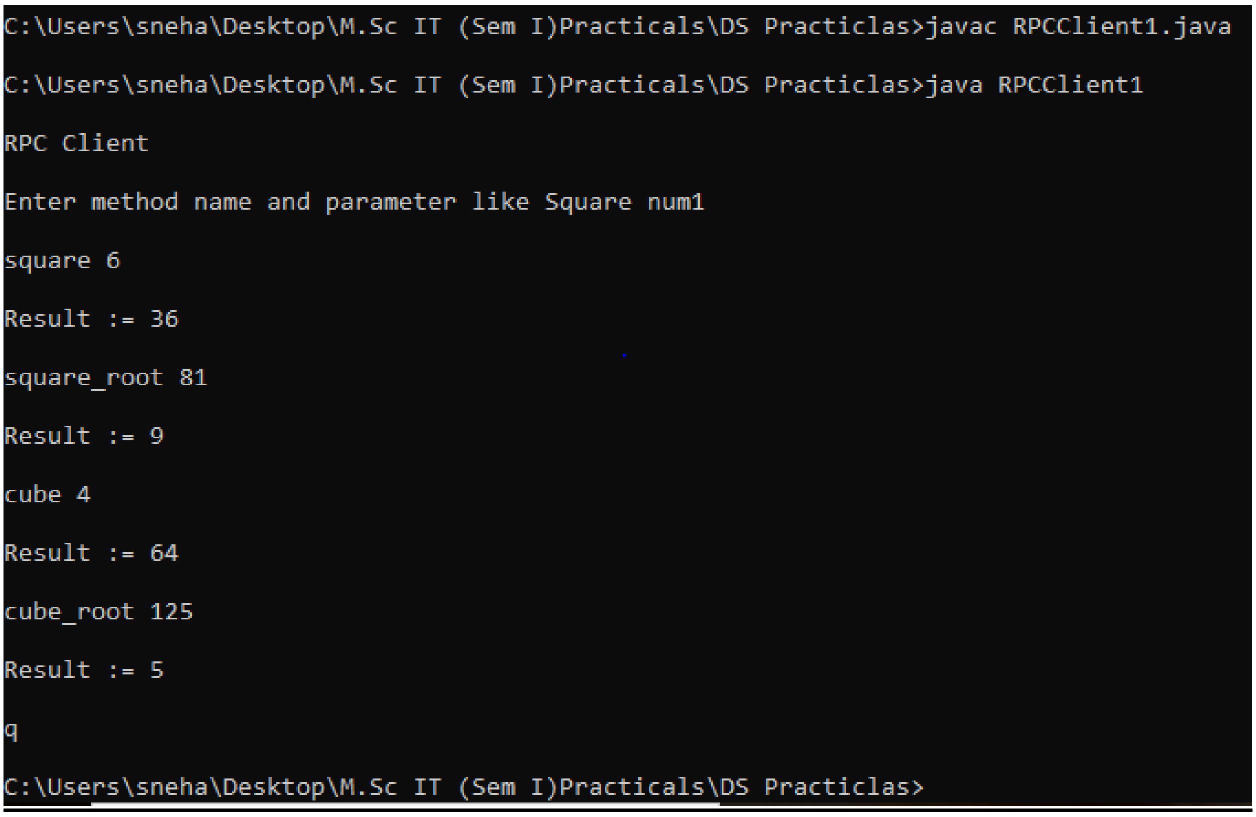
RPCClient1.java

```
import java.io.*;
import java.net.*;
class RPCClient1
{
RPCClient1()
{
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 Square num1\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 RPCClient1();
}
}
```

OUTPUT :-



```
C:\Users\sneha\Desktop\M.Sc IT (Sem I)Practicals\DS Practiclas>javac RPCServer1.java

C:\Users\sneha\Desktop\M.Sc IT (Sem I)Practicals\DS Practiclas>java RPCServer1
square 6
Result :- 36

square_root 81
Result :- 9

cube 4
Result :- 64

cube_root 125
Result :- 5


C:\Users\sneha\Desktop\M.Sc IT (Sem I)Practicals\DS Practiclas>
```



```
C:\Users\sneha\Desktop\M.Sc IT (Sem I)Practicals\DS Practiclas>javac RPCClient1.java

C:\Users\sneha\Desktop\M.Sc IT (Sem I)Practicals\DS Practiclas>java RPCClient1

RPC Client

Enter method name and parameter like Square num1

square 6

Result := 36

square_root 81

Result := 9

cube 4

Result := 64

cube_root 125

Result := 5

q

C:\Users\sneha\Desktop\M.Sc IT (Sem I)Practicals\DS Practiclas>
```

<u>Practical 4</u>

Show the implementation of Web Services

Web Services:

A Web Service is a software program that uses XML to exchange information with other software via common internet protocols. In a simple sense, Web Services are a way of interacting with objects over the internet.
A web service is
• Language Independent.
• Protocol Independent.
• Platform Independent.
• It assumes a stateless service architecture.
• Scalable (e.g. multiplying two numbers together to an entire customer-relationship management system).
• Programmable (encapsulates a task).
• Based on XML (open, text-based standard).
• Self-describing (metadata for access and use).
• Discoverable (search and locate in registries)- ability of applications and developers to search for and locate desired Web services through registries. This is based on UDDI.

Key Web Service Technologies
1. XML- Describes only data. So, any application that understands XML-regardless of the application's programming language or platform has the ability to format XML in a variety of ways (well-formed or valid). 2. SOAP- Provides a communication mechanism between services and applications. 3. WSDL- Offers a uniform method of describing web services to other programs. 4. UDDI- Enables the creation of searchable Web services registries.

Web Services Limitations
1. SOAP, WSDL, UDDI- require further development. 2. Interoperability. 3. Royalty fees. 4. Too slow for use in high-performance situations. 5. Increase traffic on networks. 6. The lack of security standards for Web services. 7. The standard procedure for describing the quality (i.e. levels of performance, reliability, security etc.) of particular Web services – management of Web services. 8. The standards that drive Web services are still in draft form (always will be in refinement).
23

9. Some vendors want to retain their intellectual property rights to certain Web services standards.

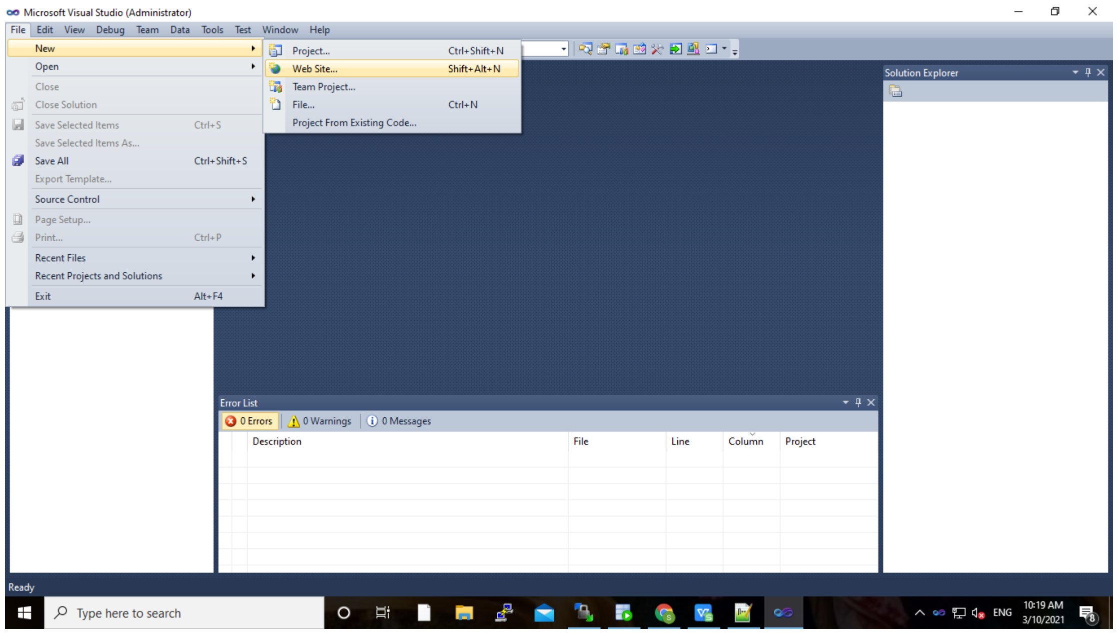
A web service can perform almost any kind of task
1. Web Portal- A web portal might obtain top news headlines from an associated press web service. 2. Weather Reporting- It can use as Weather reporting web service to display weather information in your personal website. 3. Stock Quote- It can display latest update of Share market with Stock Quote on your web site. 4. News Headline- You can display latest news update by using News Headline Web Service in your website. 5. You can make your own web service and let others use it. For example, you can make Free SMS Sending Service with footer with your companies' advertisement, so whosoever uses this service indirectly advertises your company. You can apply your ideas in N no. of ways to take advantage of it.

R. 4. A) Show the implementation of web services.

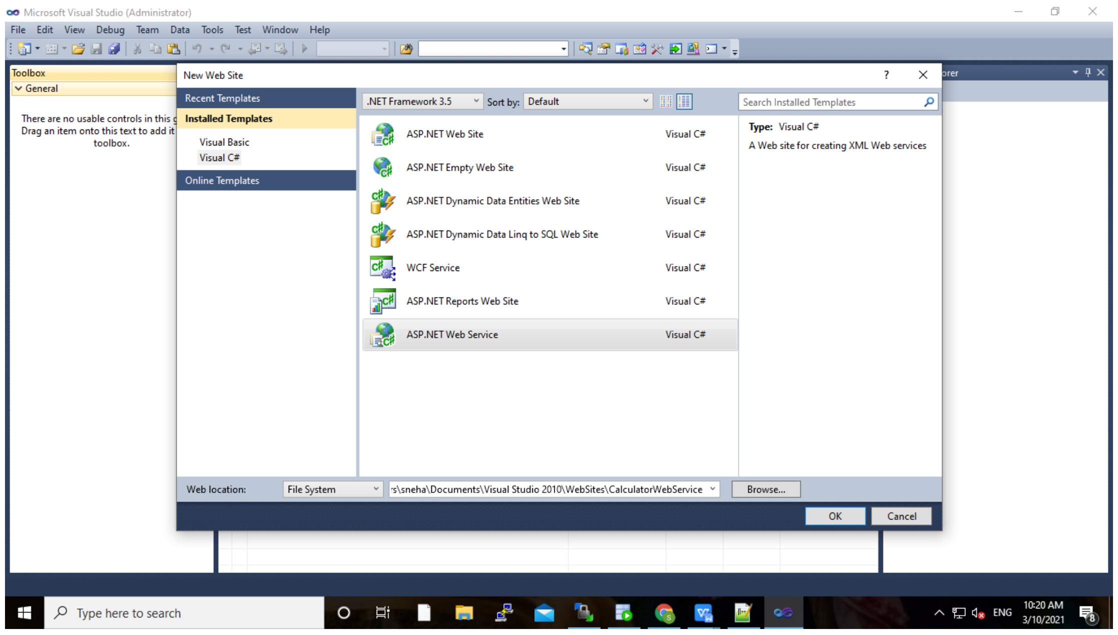
<u>Steps:</u>

1) I have used Microsoft Visual Studio 2010 Professional to implement Web Service.

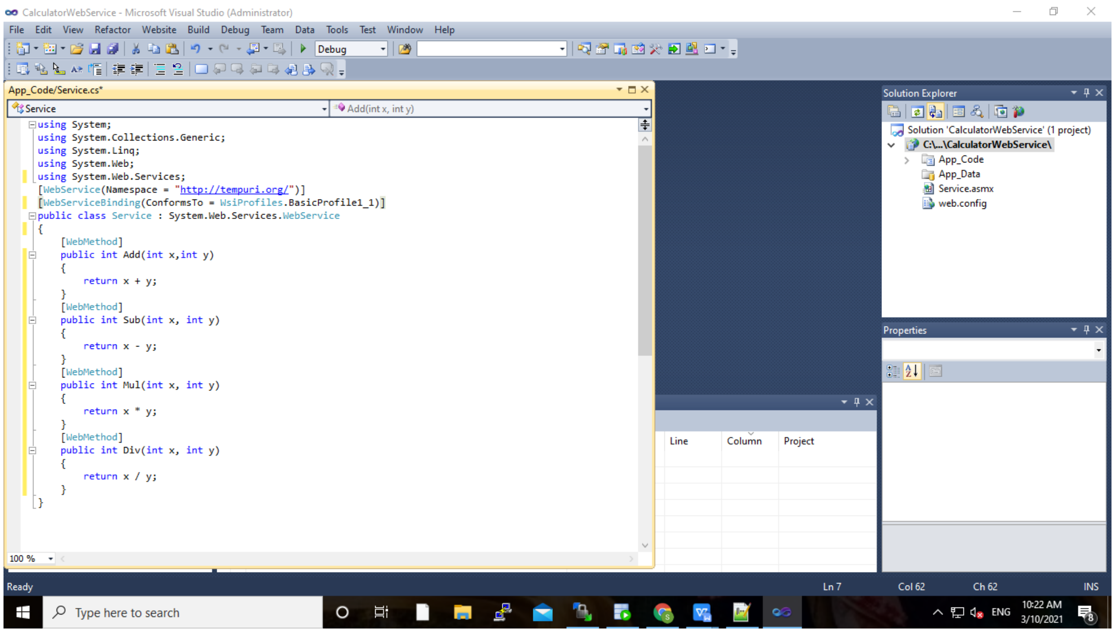
>> Click on File >> New >> Web Site.



>> Then select ASP.NET Web Service, to create a web service.

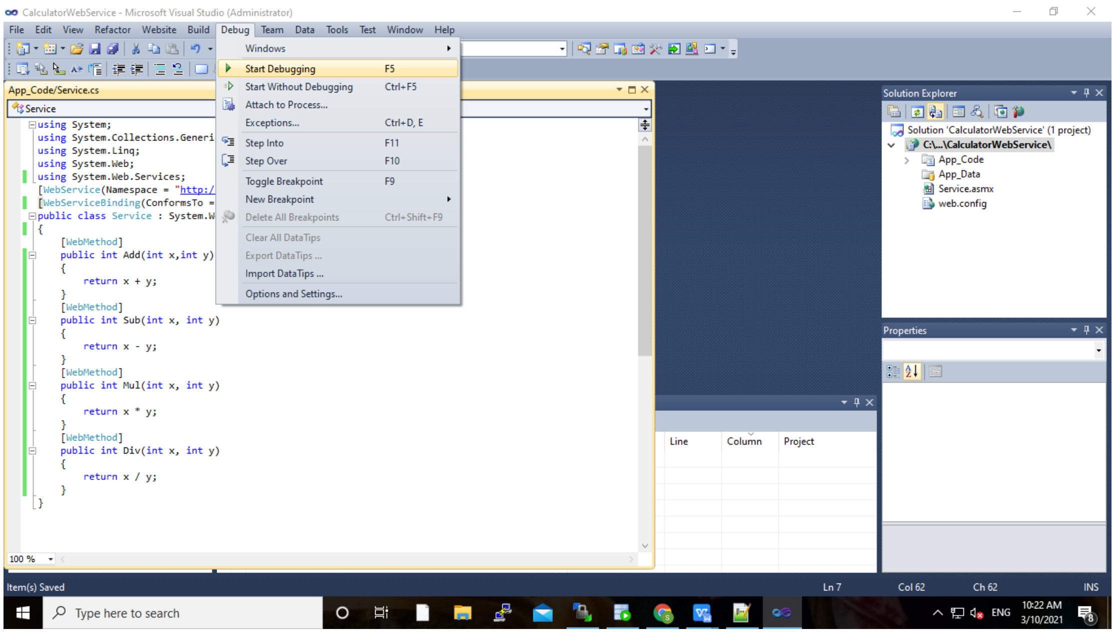

Microsoft Visual Studio (Administrator)
File Edit View Debug Team Data Tools Test Window Help
Toolbox
General
There are no usable controls in this g
Drag an item onto this text to add it
toolbox.
New Web Site
Recent Templates
Installed Templates
Visual Basic
Visual C#
Online Templates
.NET Framework 3.5
Sort by: Default
Search Installed Templates
ASP.NET Web Site Visual C#
ASP.NET Empty Web Site Visual C#
ASP.NET Dynamic Data Entities Web Site Visual C#
ASP.NET Dynamic Data Linq to SQL Web Site Visual C#
WCF Service Visual C#
ASP.NET Reports Web Site Visual C#
ASP.NET Web Service Visual C#
Type: Visual C#
A Web site for creating XML Web services
Web location: File System
's\sneha\Documents\Visual Studio 2010\WebSites\CalculatorWebService
Browse...
OK
Cancel
Type here to search
ENG
10:20 AM
3/10/2021

>> And create web methods for your web service.

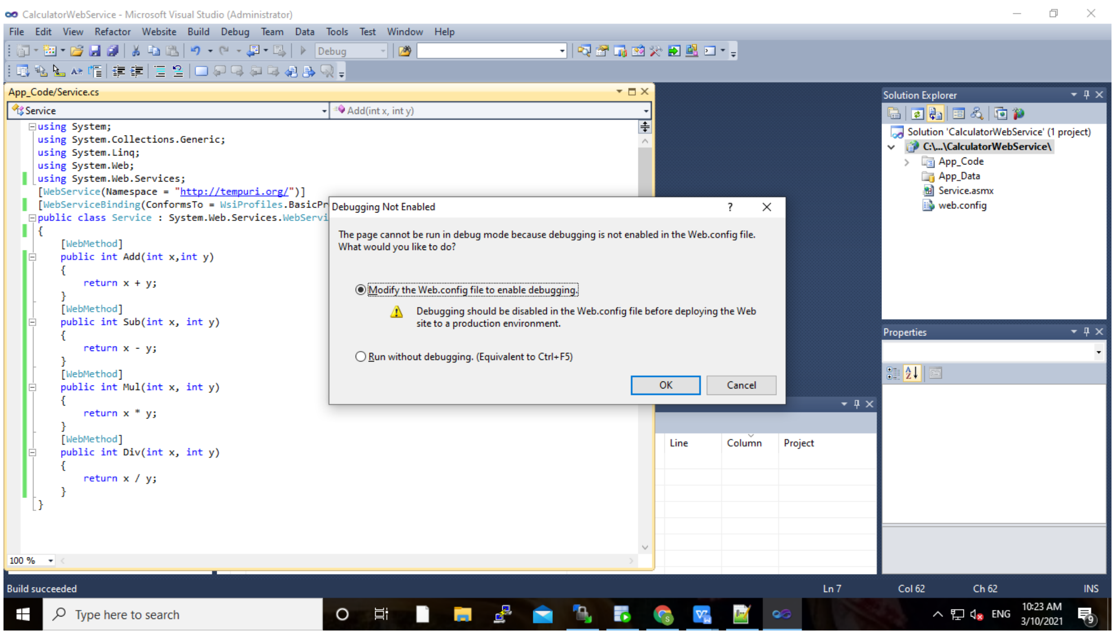

CalculatorWebService - Microsoft Visual Studio (Administrator)
File Edit View Refactor Website Build Debug Team Data Tools Test Window Help
App_Code/Service.cs*
Service
Add(int x, int y)
Solution Explorer
Solution 'CalculatorWebService' (1 project)
C:\...\CalculatorWebService\
App_Code
App_Data
Service.asmx
web.config
Properties
Line Column Project
Ready Ln 7 Col 62 Ch 62 INS
Type here to search
10:22 AM 3/10/2021

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Services;
[WebService(Namespace = "http://tempuri.org/")]
[WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]
public class Service : System.Web.Services.WebService
{
    [WebMethod]
    public int Add(int x,int y)
    {
        return x + y;
    }
    [WebMethod]
    public int Sub(int x, int y)
    {
        return x - y;
    }
    [WebMethod]
    public int Mul(int x, int y)
    {
        return x * y;
    }
    [WebMethod]
    public int Div(int x, int y)
    {
        return x / y;
    }
}
```

>> Now, debug that web service. And copy the web service path.



```
using System;
using System.Collections.Generi
using System.Linq;
using System.Web;
using System.Web.Services;
[WebService(Namespace = "http:/
[WebServiceBinding(ConformsTo =
public class Service : System.W
{
    [WebMethod]
    public int Add(int x,int y)
    {
        return x + y;
    }
    [WebMethod]
    public int Sub(int x, int y)
    {
        return x - y;
    }
    [WebMethod]
    public int Mul(int x, int y)
    {
        return x * y;
    }
    [WebMethod]
    public int Div(int x, int y)
    {
        return x / y;
    }
}
```


CalculatorWebService - Microsoft Visual Studio (Administrator)
File Edit View Refactor Website Build Debug Team Data Tools Test Window Help
App_Code/Service.cs
Service
Add(int x, int y)
Solution Explorer
Solution 'CalculatorWebService' (1 project)
C:\...\CalculatorWebService\
App_Code
App_Data
Service.asmx
web.config
Properties
Line
Column
Project
Build succeeded
Ln 7
Col 62
Ch 62
INS

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Services;
[WebService(Namespace = "http://tempuri.org/")]
[WebServiceBinding(ConformsTo = WsiProfiles.BasicPr
public class Service : System.Web.Services.WebServi
{
    [WebMethod]
    public int Add(int x,int y)
    {
        return x + y;
    }
    [WebMethod]
    public int Sub(int x, int y)
    {
        return x - y;
    }
    [WebMethod]
    public int Mul(int x, int y)
    {
        return x * y;
    }
    [WebMethod]
    public int Div(int x, int y)
    {
        return x / y;
    }
}
```

Debugging Not Enabled

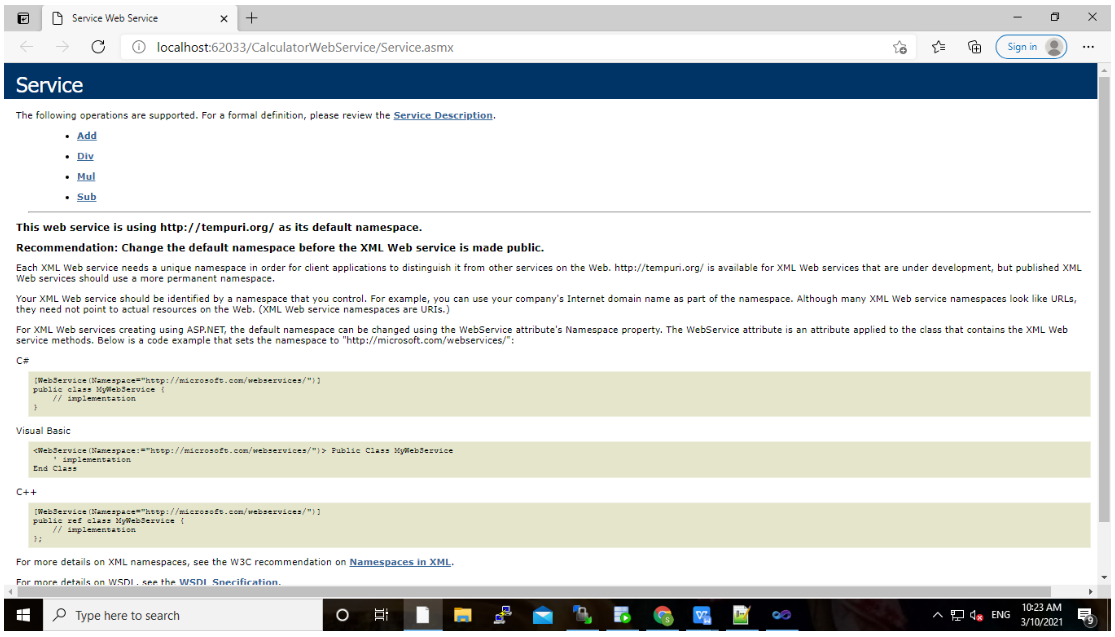
The page cannot be run in debug mode because debugging is not enabled in the Web.config file. What would you like to do?

[✓ Modify the Web.config file to enable debugging.]

Debugging should be disabled in the Web.config file before deploying the Web site to a production environment.

Run without debugging. (Equivalent to Ctrl+F5)

OK Cancel



Service

The following operations are supported. For a formal definition, please review the Service Description.

- Add
- Div
- Mul
- Sub

This web service is using http://tempuri.org/ as its default namespace.

Recommendation: Change the default namespace before the XML Web service is made public.

Each XML Web service needs a unique namespace in order for client applications to distinguish it from other services on the Web. http://tempuri.org/ is available for XML Web services that are under development, but published XML Web services should use a more permanent namespace.

Your XML Web service should be identified by a namespace that you control. For example, you can use your company's Internet domain name as part of the namespace. Although many XML Web service namespaces look like URLs, they need not point to actual resources on the Web. (XML Web service namespaces are URIs.)

For XML Web services creating using ASP.NET, the default namespace can be changed using the WebService attribute's Namespace property. The WebService attribute is an attribute applied to the class that contains the XML Web service methods. Below is a code example that sets the namespace to "http://microsoft.com/webservices/":

C#

```
[WebService(Namespace="http://microsoft.com/webservices/")]
public class MyWebService {
    // implementation
}
```

Visual Basic

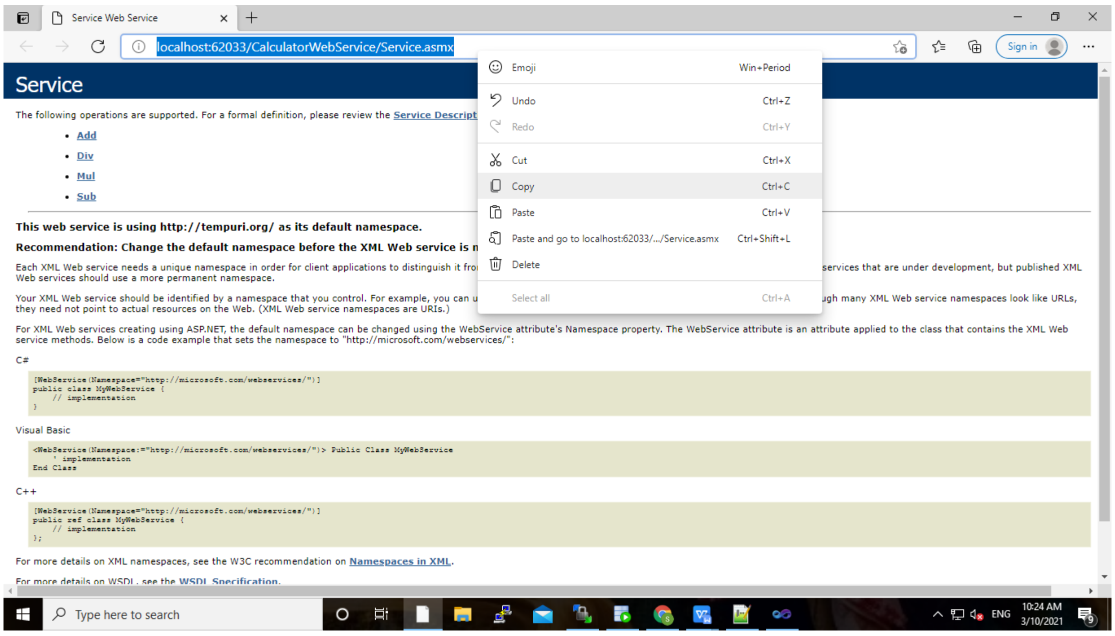
```
<WebService(Namespace:="http://microsoft.com/webservices/")> Public Class MyWebService
    ' implementation
End Class
```

C++

```
[WebService(Namespace="http://microsoft.com/webservices/")]
public ref class MyWebService {
    // implementation
};
```

For more details on XML namespaces, see the W3C recommendation on Namespaces in XML.

For more details on WSDL, see the WSDL Specification.



Service

The following operations are supported. For a formal definition, please review the Service Descript

- Add
- Div
- Mul
- Sub

This web service is using http://tempuri.org/ as its default namespace.

Recommendation: Change the default namespace before the XML Web service is m

Each XML Web service needs a unique namespace in order for client applications to distinguish it fro services that are under development, but published XML Web services should use a more permanent namespace.

Your XML Web service should be identified by a namespace that you control. For example, you can u ugh many XML Web service namespaces look like URLs, they need not point to actual resources on the Web. (XML Web service namespaces are URIs.)

For XML Web services creating using ASP.NET, the default namespace can be changed using the WebService attribute's Namespace property. The WebService attribute is an attribute applied to the class that contains the XML Web service methods. Below is a code example that sets the namespace to "http://microsoft.com/webservices/":

C#

```
[WebService(Namespace="http://microsoft.com/webservices/")]
public class MyWebService {
    // implementation
}
```

Visual Basic

```
<WebService(Namespace:="http://microsoft.com/webservices/")> Public Class MyWebService
    ' implementation
End Class
```

C++

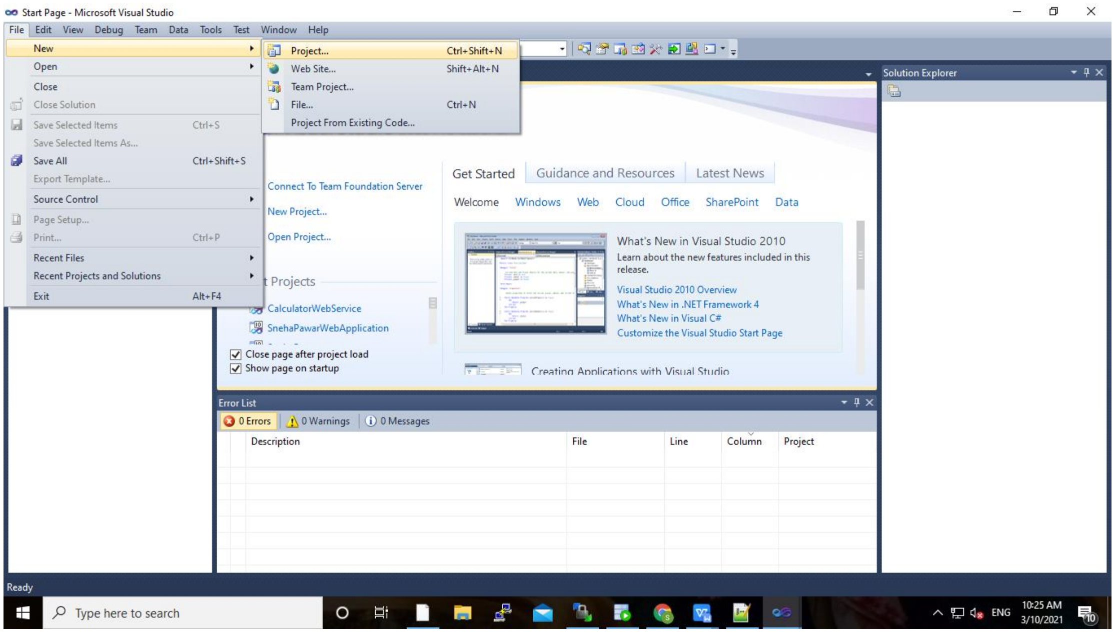
```
[WebService(Namespace="http://microsoft.com/webservices/")]
public ref class MyWebService {
    // implementation
};
```

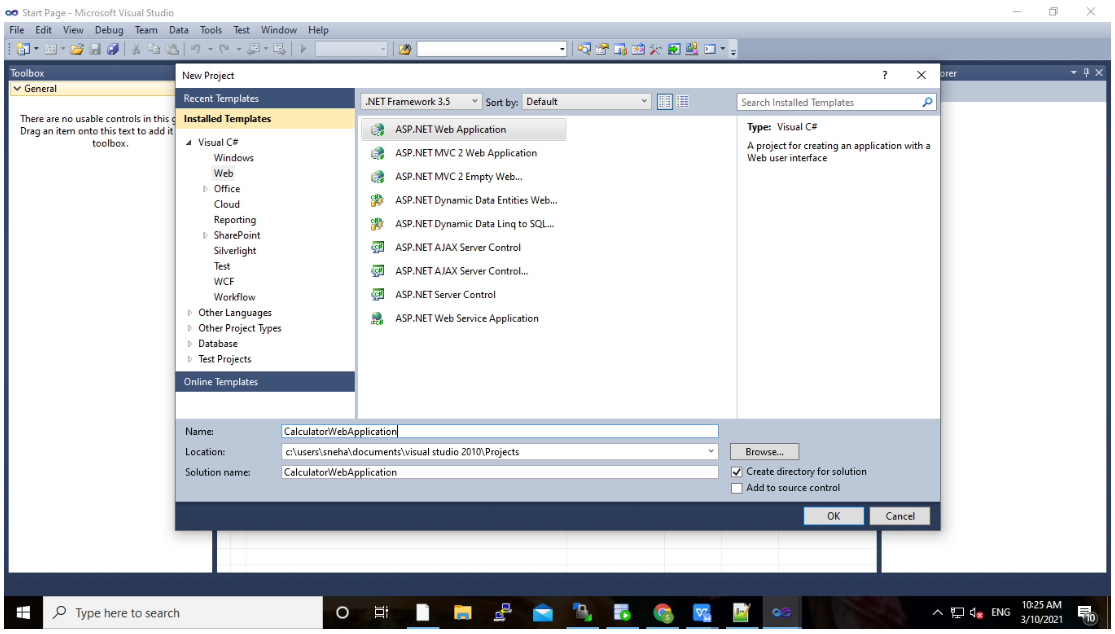
For more details on XML namespaces, see the W3C recommendation on Namespaces in XML.

For more details on WSDL, see the WSDL Specification.

Service Web Service

>> Now Create a new Web Application toconsume your web service.
>> New >> Project >> ASP.NET web application.


Start Page - Microsoft Visual Studio
File Edit View Debug Team Data Tools Test Window Help
New
Open
Close
Close Solution
Save Selected Items Ctrl+S
Save Selected Items As...
Save All Ctrl+Shift+S
Export Template...
Source Control
Page Setup...
Print... Ctrl+P
Recent Files
Recent Projects and Solutions
Exit Alt+F4
Project... Ctrl+Shift+N
Web Site... Shift+Alt+N
Team Project...
File... Ctrl+N
Project From Existing Code...
Solution Explorer
Connect To Team Foundation Server
New Project...
Open Project...
Projects
CalculatorWebService
SnehaPawarWebApplication
Close page after project load
Show page on startup
Get Started
Guidance and Resources
Latest News
Welcome Windows Web Cloud Office SharePoint Data
What's New in Visual Studio 2010
Learn about the new features included in this release.
Visual Studio 2010 Overview
What's New in .NET Framework 4
What's New in Visual C#
Customize the Visual Studio Start Page
Creating Applications with Visual Studio
Error List
0 Errors
0 Warnings
0 Messages
Description File Line Column Project
Ready
Type here to search
10:25 AM
3/10/2021


Start Page - Microsoft Visual Studio
File Edit View Debug Team Data Tools Test Window Help
Toolbox
General
New Project
Recent Templates
Installed Templates
Visual C#
Windows
Web
Office
Cloud
Reporting
SharePoint
Silverlight
Test
WCF
Workflow
Other Languages
Other Project Types
Database
Test Projects
Online Templates
ASP.NET Web Application
ASP.NET MVC 2 Web Application
ASP.NET MVC 2 Empty Web...
ASP.NET Dynamic Data Entities Web...
ASP.NET Dynamic Data Linq to SQL...
ASP.NET AJAX Server Control
ASP.NET AJAX Server Control...
ASP.NET Server Control
ASP.NET Web Service Application
Search Installed Templates
Browse...
OK
Cancel
Type here to search
ENG
10:25 AM
3/10/2021

There are no usable controls in this group. Drag an item onto this text to add it to the toolbox.

.NET Framework 3.5 Sort by: Default

Type: Visual C#

A project for creating an application with a Web user interface

Name: CalculatorWebApplication

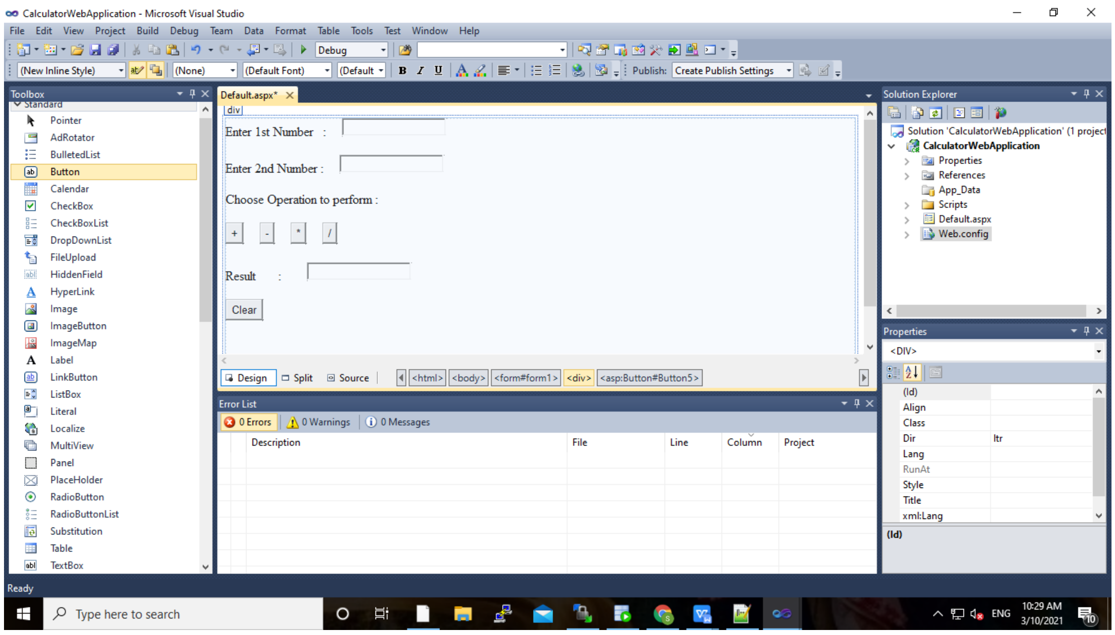
Location: c:\users\sneha\documents\visual studio 2010\Projects

Solution name: CalculatorWebApplication

☑ Create directory for solution

☐ Add to source control

>> Design the web page as per your requirement.



Enter 1st Number :

Enter 2nd Number :

Choose Operation to perform :

+ - * /

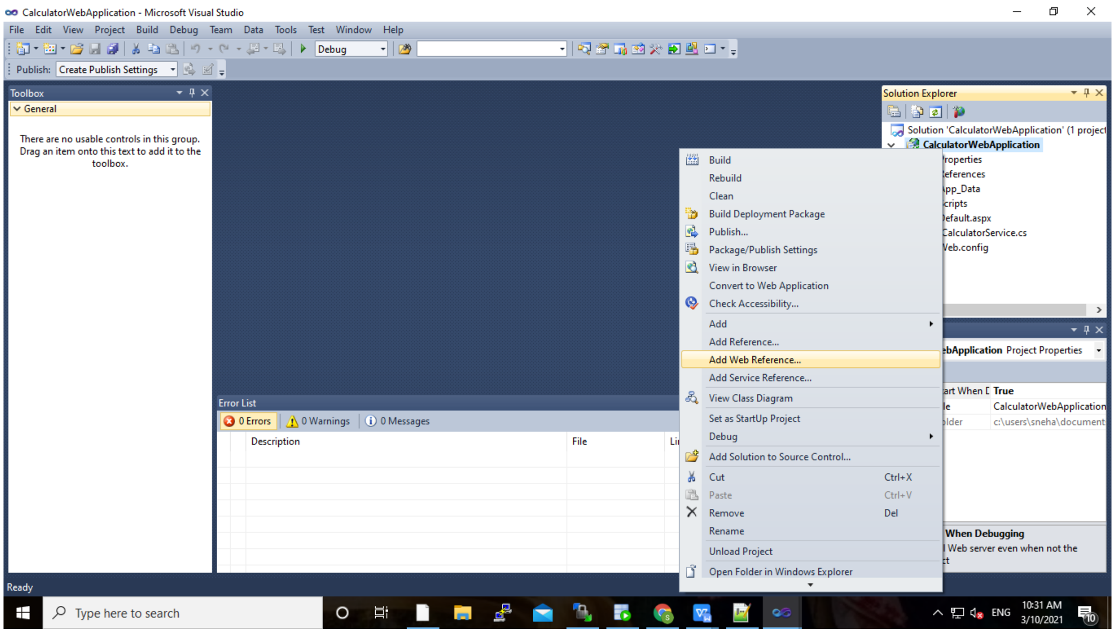
Result :

Clear

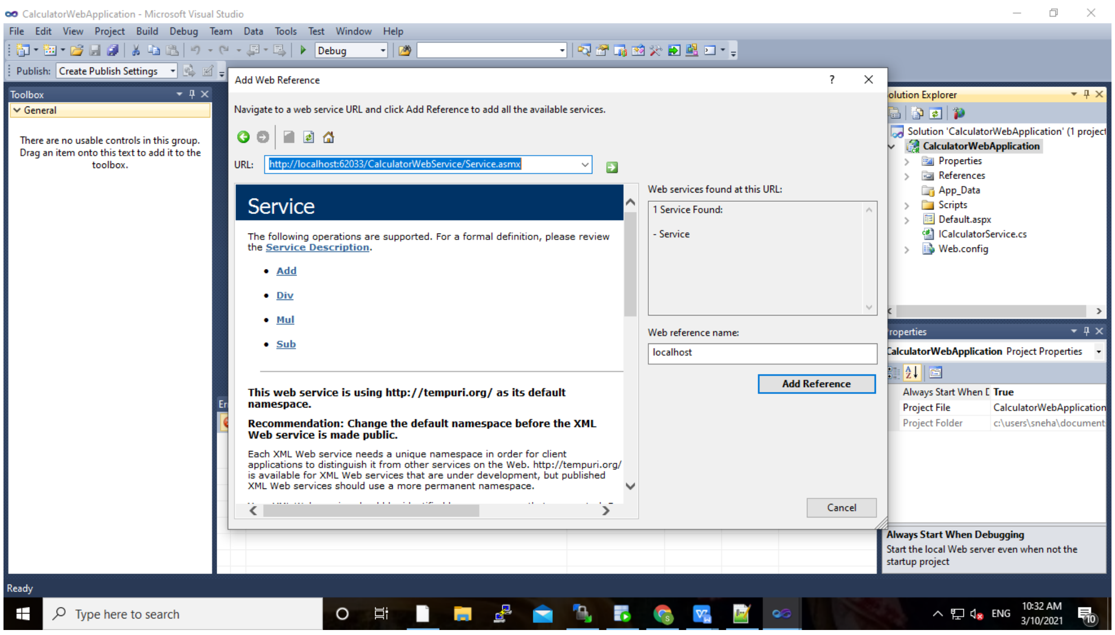
Default.aspx

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Default.aspx.cs"
Inherits="CalculatorWebApplication._Default" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
        Enter 1st Number&nbsp;&nbsp; :&nbsp;&nbsp;&nbsp;&nbsp;
        <asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>
        <br />
        <br />
        Enter 2nd Number :&nbsp;&nbsp;&nbsp;&nbsp;
        <asp:TextBox ID="TextBox2" runat="server"></asp:TextBox>
        <br />
        <br />
        Choose Operation to perform :
        <br />
        <br />
        <asp:Button ID="Button1" runat="server" Text="+" />
&nbsp;&nbsp;&nbsp;&nbsp;
        <asp:Button ID="Button2" runat="server" Text="-" />
&nbsp;&nbsp;&nbsp;&nbsp;
        <asp:Button ID="Button3" runat="server" Text="*" />
&nbsp;&nbsp;&nbsp;&nbsp;
        <asp:Button ID="Button4" runat="server" Text="/" />
        <br />
        <br />
        Result&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp; :&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;
        <asp:TextBox ID="TextBox3" runat="server"></asp:TextBox>
        <br />
        <br />
        <asp:Button ID="Button5" runat="server" Text="Clear" />
        <br />
        <br />
        <br />

    </div>
    </form>
</body>
</html>
```

>> Now Add a web reference of your created web service to your web application.
>> To add Web Reference Right Click on your web application and then select Add Web Reference.



>> Paste the copied link of your web Service. And then click on Arrow, and then click on Add Reference.


Add Web Reference

Navigate to a web service URL and click Add Reference to add all the available services.

URL: http://localhost:62033/CalculatorWebService/Service.asmx

Service

The following operations are supported. For a formal definition, please review the Service Description.

- Add
- Div
- Mul
- Sub

This web service is using http://tempuri.org/ as its default namespace.

Recommendation: Change the default namespace before the XML Web service is made public.

Each XML Web service needs a unique namespace in order for client applications to distinguish it from other services on the Web. http://tempuri.org/ is available for XML Web services that are under development, but published XML Web services should use a more permanent namespace.

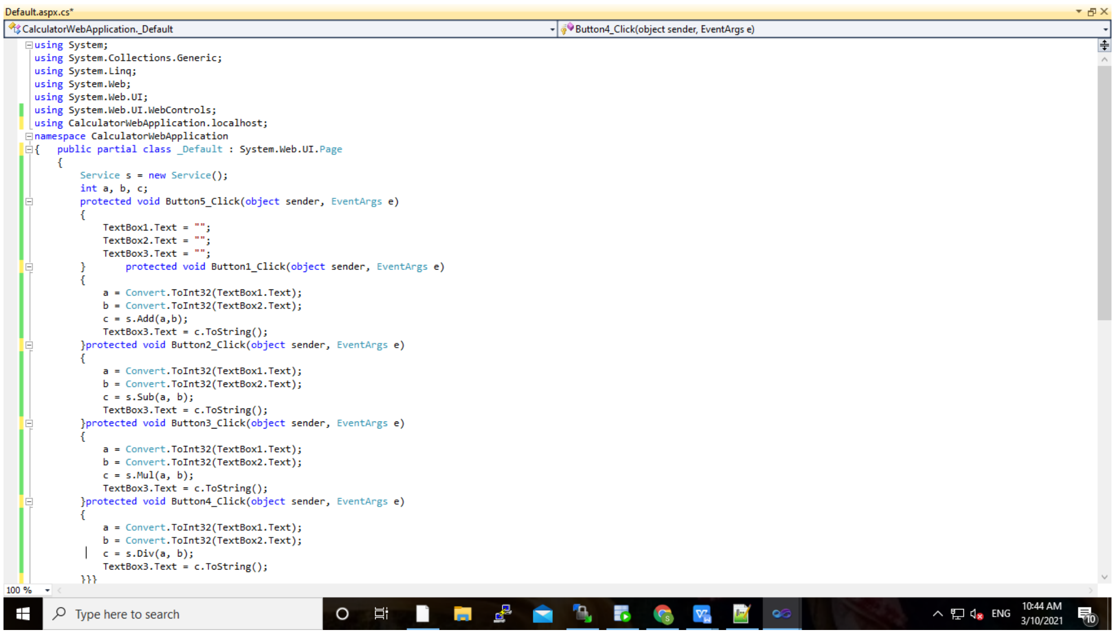
Web services found at this URL:

1 Service Found:

- Service

Web reference name: localhost

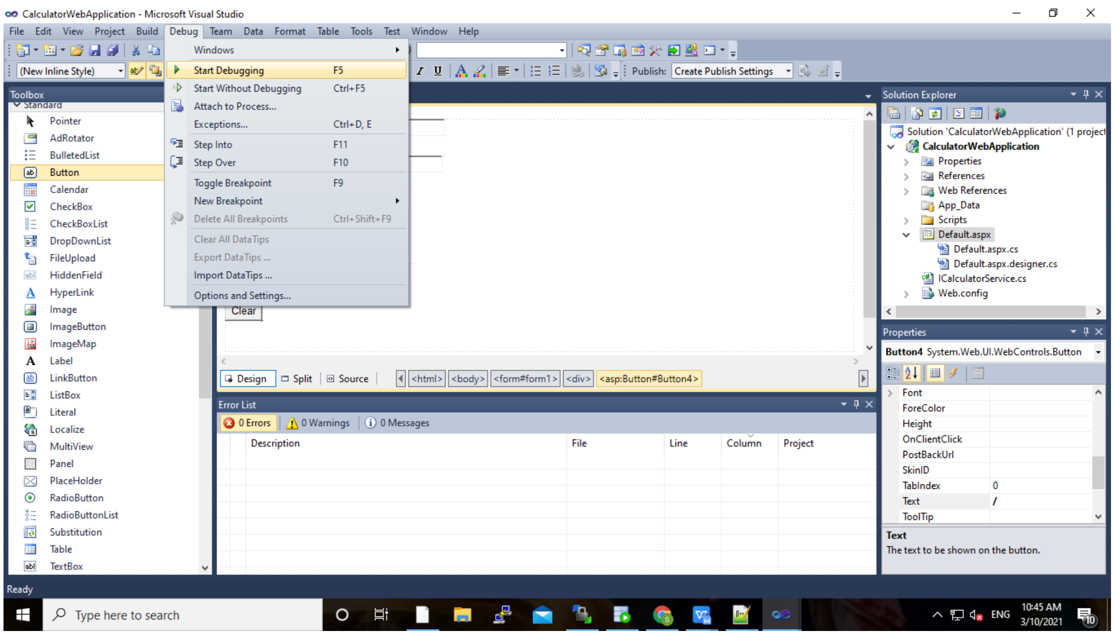
>> Edit the Default.aspx.cs file as follows.

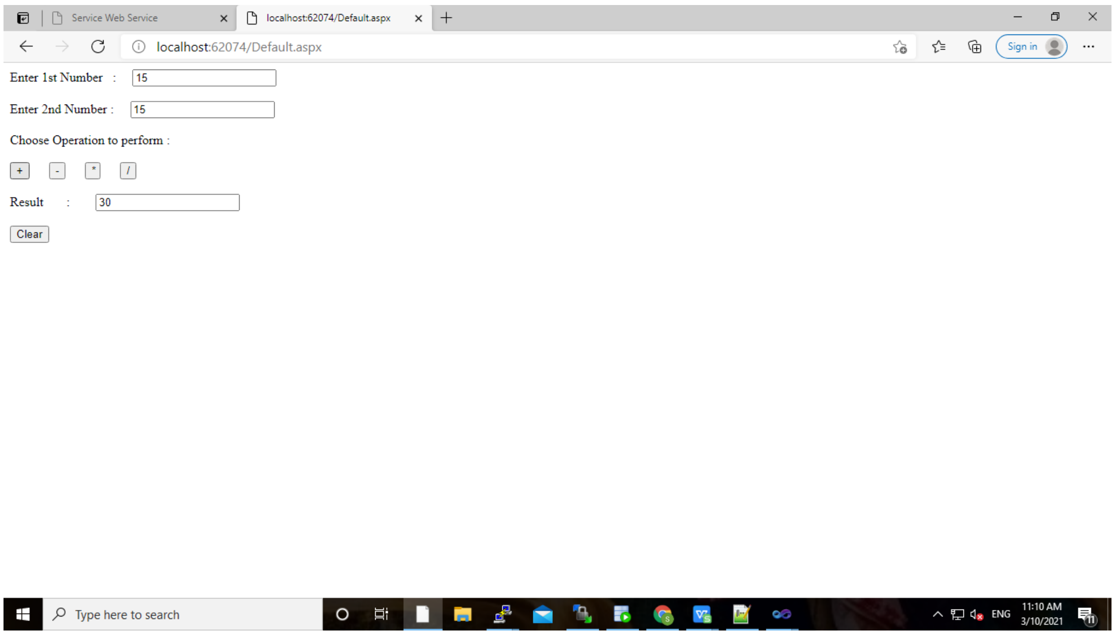

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using CalculatorWebApplication.localhost;
namespace CalculatorWebApplication
{   public partial class _Default : System.Web.UI.Page
    {
        Service s = new Service();
        int a, b, c;
        protected void Button5_Click(object sender, EventArgs e)
        {
            TextBox1.Text = "";
            TextBox2.Text = "";
            TextBox3.Text = "";
        }       protected void Button1_Click(object sender, EventArgs e)
        {
            a = Convert.ToInt32(TextBox1.Text);
            b = Convert.ToInt32(TextBox2.Text);
            c = s.Add(a,b);
            TextBox3.Text = c.ToString();
        }protected void Button2_Click(object sender, EventArgs e)
        {
            a = Convert.ToInt32(TextBox1.Text);
            b = Convert.ToInt32(TextBox2.Text);
            c = s.Sub(a, b);
            TextBox3.Text = c.ToString();
        }protected void Button3_Click(object sender, EventArgs e)
        {
            a = Convert.ToInt32(TextBox1.Text);
            b = Convert.ToInt32(TextBox2.Text);
            c = s.Mul(a, b);
            TextBox3.Text = c.ToString();
        }protected void Button4_Click(object sender, EventArgs e)
        {
            a = Convert.ToInt32(TextBox1.Text);
            b = Convert.ToInt32(TextBox2.Text);
            c = s.Div(a, b);
            TextBox3.Text = c.ToString();
        }}}
```

Default.aspx.cs
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using CalculatorWebApplication.localhost;
namespace CalculatorWebApplication
{   public partial class _Default : System.Web.UI.Page
    {
        Service s = new Service();
        int a, b, c;
        protected void Button5_Click(object sender, EventArgs e)
        {
```

```
        TextBox1.Text = "";
        TextBox2.Text = "";
        TextBox3.Text = "";
    }       protected void Button1_Click(object sender, EventArgs e)
    {
        a = Convert.ToInt32(TextBox1.Text);
        b = Convert.ToInt32(TextBox2.Text);
        c = s.Add(a,b);
        TextBox3.Text = c.ToString();
    }protected void Button2_Click(object sender, EventArgs e)
    {
        a = Convert.ToInt32(TextBox1.Text);
        b = Convert.ToInt32(TextBox2.Text);
        c = s.Sub(a, b);
        TextBox3.Text = c.ToString();
    }protected void Button3_Click(object sender, EventArgs e)
    {
        a = Convert.ToInt32(TextBox1.Text);
        b = Convert.ToInt32(TextBox2.Text);
        c = s.Mul(a, b);
        TextBox3.Text = c.ToString();
    }protected void Button4_Click(object sender, EventArgs e)
    {
        a = Convert.ToInt32(TextBox1.Text);
        b = Convert.ToInt32(TextBox2.Text);
        c = s.Div(a, b);
        TextBox3.Text = c.ToString();
    }}}
```

>> Now Debug your Web Application.

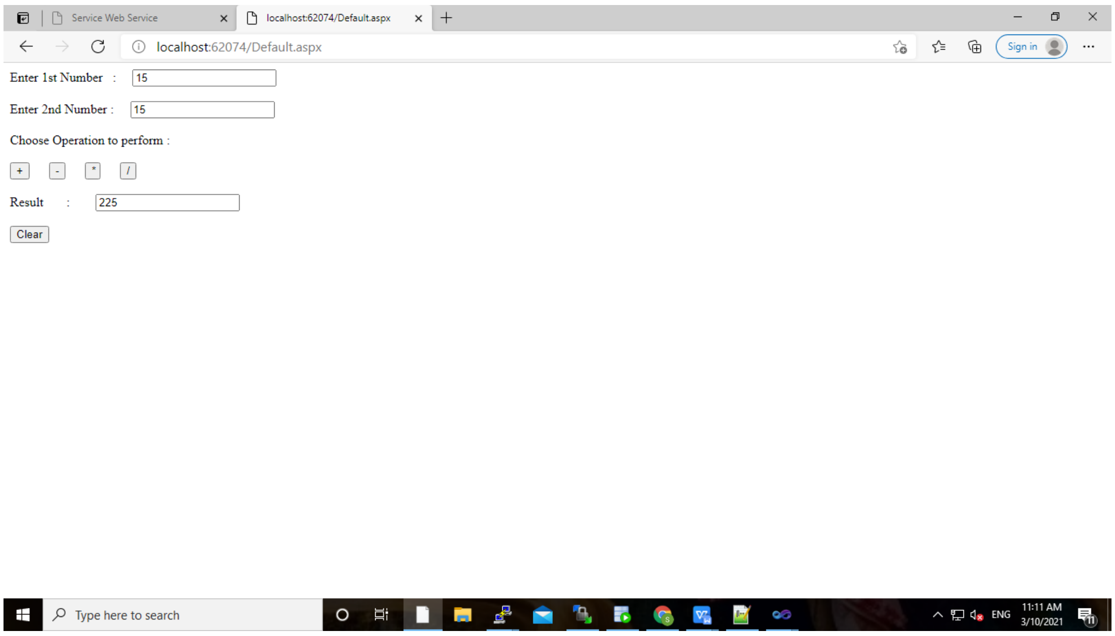



Enter 1st Number : 15

Enter 2nd Number : 15

Choose Operation to perform :

Result : 30


Enter 1st Number : 15

Enter 2nd Number : 15

Choose Operation to perform :

Result : 225

Practical 5

Write a program to execute any one mutual exclusion algorithm.

Mutual exclusion:
Mutual exclusion is a concurrency control property which is introduced to prevent race conditions. It is the requirement that a process cannot enter its critical section while another concurrent process is currently present or executing in its critical section i.e. only one process is allowed to execute the critical section at any given instance of time.

Distributed mutual exclusion algorithms must deal with unpredictable message delays and incomplete knowledge of the system state.
Three basic approaches for distributed mutual exclusion:
1. Token based approach 2. Non-token-based approach 3. Quorum based approach

Token-based approach: • A unique token is shared among the sites. • A site is allowed to enter its CS if it possesses the token. • Mutual exclusion is ensured because the token is unique. • Example: Suzuki-Kasami's Broadcast Algorithm

Non-token-based approach: • Two or more successive rounds of messages are exchanged among the sites to determine which site will enter the CS next. • Example: Lamport's algorithm, Ricart–Agrawala algorithm

Quorum based approach: • Each site requests permission to execute the CS from a subset of sites (called a quorum). • Any two quorums contain a common site. • This common site is responsible to make sure that only one request executes the CS at any time. • Example: Maekawa's Algorithm

Requirements of Mutual Exclusion Algorithms:
1. No Deadlock: Two or more site should not endlessly wait for any message that will never arrive.
32

2. No Starvation: Every site who wants to execute critical section should get an opportunity to execute it in finite time. Any site should not wait indefinitely to execute critical section while other site is repeatedly executing critical section 3. Fairness: Each site should get a fair chance to execute critical section. Any request to execute critical section must be executed in the order they are made i.e Critical section execution requests should be executed in the order of their arrival in the system. 4. Fault Tolerance: In case of failure, it should be able to recognize it by itself in order to continue functioning without any disruption.

Q. 5 A) Write a program to execute any one mutual exclusion algorithm. (The Server which has Token can only send messages to the server).

CODE

TokenServer.java

```
import java.net.*;
public class TokenServer
{
public static void main(String agrs[])throws Exception
{
     while(true)
     {
         Server sr=new Server();
         sr.recPort(8000);
         sr.recData();
     }
}
}
class Server
{
boolean hasToken=false;
    boolean sendData=false;
    int recport;
    void recPort(int recport)
    {
     this.recport=recport;
}
void recData()throws Exception
{
     byte buff[]=new byte[256];
     DatagramSocket ds;
       DatagramPacket dp;
       String str;
     ds=new DatagramSocket(recport);
     dp=new DatagramPacket(buff,buff.length);
     ds.receive(dp);
                        ds.close();
str=new String(dp.getData(),0,dp.getLength());
System.out.println("The Message From :- "+str);
}
}
```

TokenClient1.java

```
import java.io.*;
import java.net.*;
public class TokenClient1
{
public static void main(String arg[]) throws Exception
{
InetAddress lclhost;
BufferedReader br;
String str="";
TokenClient12 tkcl,tkser;
boolean hasToken;
boolean setSendData;
while(true)
{
lclhost=InetAddress.getLocalHost();
tkcl = new TokenClient12(lclhost);
tkser = new TokenClient12(lclhost);
tkcl.setSendPort(9004);
tkcl.setRecPort(8002);
lclhost=InetAddress.getLocalHost();
tkser.setSendPort(9000);
if(tkcl.hasToken == true)
{
System.out.println("Do you want to enter the Data -->YES/NO");
br=new BufferedReader(new InputStreamReader(System.in));
str=br.readLine();
if(str.equalsIgnoreCase("yes"))
{
System.out.println("ready to send");
tkser.setSendData = true;
tkser.sendData();
tkser.setSendData = false;
}
else if(str.equalsIgnoreCase("no"))
{
tkcl.sendData();
tkcl.recData();
}
}
else
{
System.out.println("ENTERING RECEIVING MODE");
tkcl.recData();
}
}
}
}
class TokenClient12
{
InetAddress lclhost;
int sendport,recport;
boolean hasToken = true;
boolean setSendData = false;
TokenClient12 tkcl,tkser;
TokenClient12(InetAddress lclhost)
{
this.lclhost = lclhost;
}
void setSendPort(int sendport)
{
this.sendport = sendport;
}
void setRecPort(int recport)
{
this.recport = recport;
}
```

```
void sendData() throws Exception
{
BufferedReader br;
String str="Token";
DatagramSocket ds;
DatagramPacket dp;
if(setSendData == true)
{
System.out.println("Enter the Data");
br=new BufferedReader(new InputStreamReader(System.in));
str = "ClientOne....." + br.readLine();
System.out.println("now sending.....");
System.out.println("Data Sent");
}
ds = new DatagramSocket(sendport);
dp = new DatagramPacket(str.getBytes(),str.length(),lclhost,sendport-1000);
ds.send(dp);
ds.close();
setSendData = false;
hasToken = false;
}
void recData()throws Exception
{
String msgstr;
byte buffer[] = new byte[256];
DatagramSocket ds;
DatagramPacket dp;
ds = new DatagramSocket(recport);
dp = new DatagramPacket(buffer,buffer.length);
ds.receive(dp);
ds.close();
msgstr = new String(dp.getData(),0,dp.getLength());
System.out.println("The data is "+msgstr);
if(msgstr.equals("Token"))
{
hasToken = true;
}
}
}
```

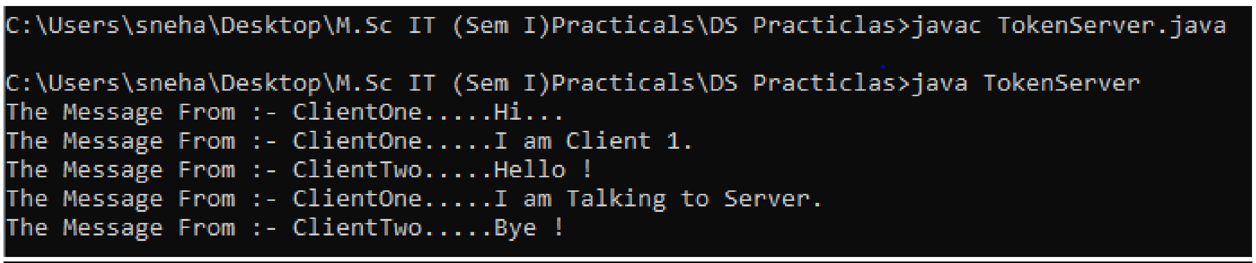
TokenClient2.java

```
import java.io.*;
import java.net.*;
public class TokenClient2
{
static boolean setSendData ;
static boolean hasToken ;
public static void main(String arg[]) throws Exception
{
InetAddress lclhost;
BufferedReader br;
String str1;
TokenClient21 tkcl;
TokenClient21 ser;
```

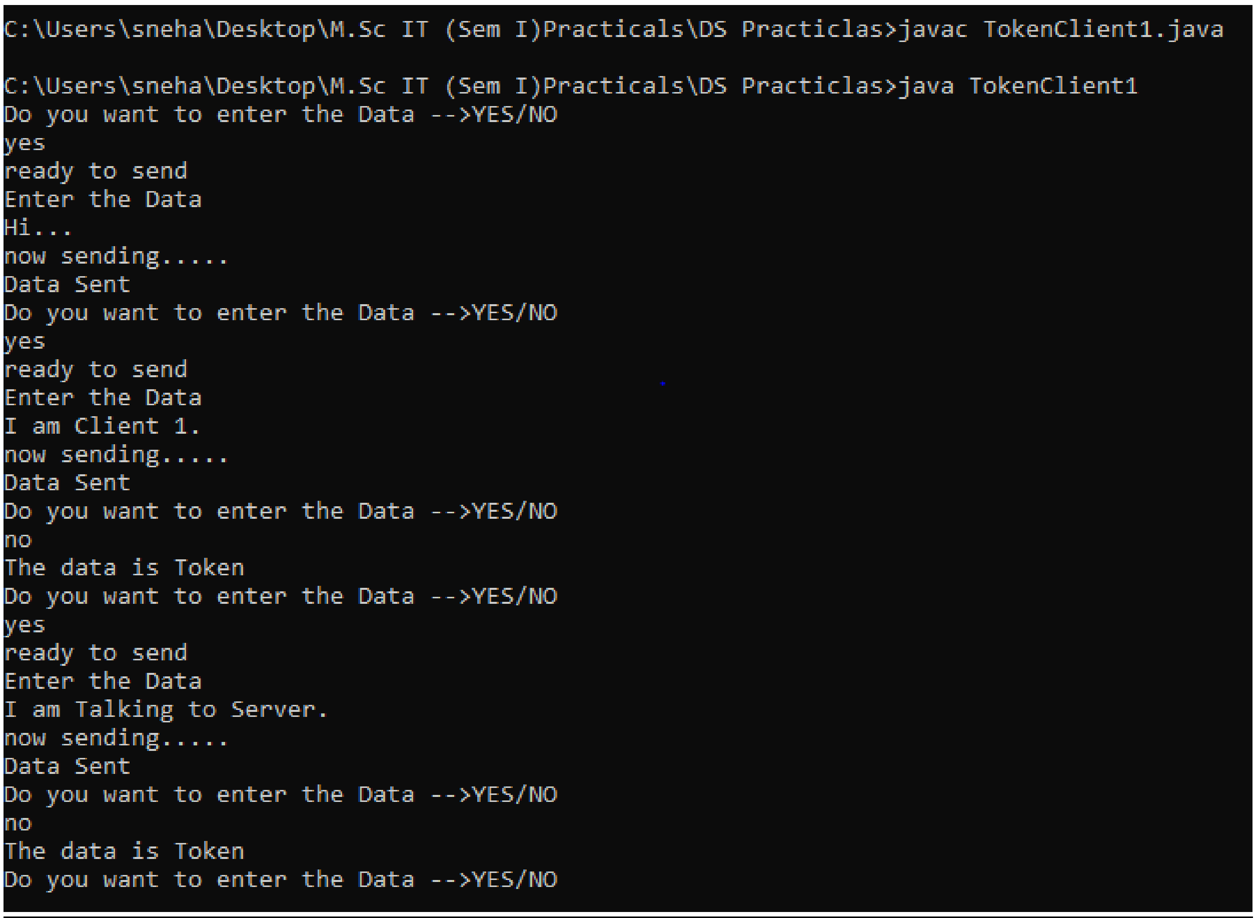
```
while(true)
{
lclhost=InetAddress.getLocalHost();
tkcl = new TokenClient21(lclhost);
tkcl.setRecPort(8004);
tkcl.setSendPort(9002);
lclhost=InetAddress.getLocalHost();
ser = new TokenClient21(lclhost);
ser.setSendPort(9000);
if(hasToken == true)
{
System.out.println("Do you want to enter the Data -->YES/NO");
br=new BufferedReader(new InputStreamReader(System.in));
str1=br.readLine();
if(str1.equalsIgnoreCase("yes"))
{
System.out.println("Ready to send");
ser.setSendData = true;
ser.sendData();
ser.setSendData = false;
}
else if(str1.equalsIgnoreCase("no"))
{
tkcl.sendData();
tkcl.recData();
}
}
else
{
System.out.println("ENTERING RECEIVING MODE");
tkcl.recData();
hasToken=true;
}
}
}
}
class TokenClient21
{
InetAddress lclhost;
int sendport,recport;
boolean setSendData = false;
boolean hasToken = false;
TokenClient21 tkcl;
TokenClient21 ser;
TokenClient21(InetAddress lclhost)
{
this.lclhost = lclhost;
}
void setSendPort(int sendport)
{
this.sendport = sendport;
}
void setRecPort(int recport)
{
this.recport = recport;
}
void sendData() throws Exception
{
```

```
BufferedReader br;
String str = "Token";
DatagramSocket ds;
DatagramPacket dp;
if(setSendData == true)
{
System.out.println("Enter the Data");
br=new BufferedReader(new InputStreamReader(System.in));
str = "ClientTwo....." + br.readLine();
System.out.println("now sending.....");
System.out.println("Data Sent");
}
ds = new DatagramSocket(sendport);
dp = new DatagramPacket(str.getBytes(),str.length(),lclhost,sendport-1000);
ds.send(dp);
ds.close();
setSendData = false;
hasToken = false;
}
void recData()throws Exception
{
String msgstr;
byte buffer[] = new byte[256];
DatagramSocket ds;
DatagramPacket dp;
ds = new DatagramSocket(recport);
dp = new DatagramPacket(buffer,buffer.length);
ds.receive(dp);
ds.close();
msgstr = new String(dp.getData(),0,dp.getLength());
System.out.println("The data is "+msgstr);
if(msgstr.equals("Token"))
{
hasToken = true;
}
}
}
```

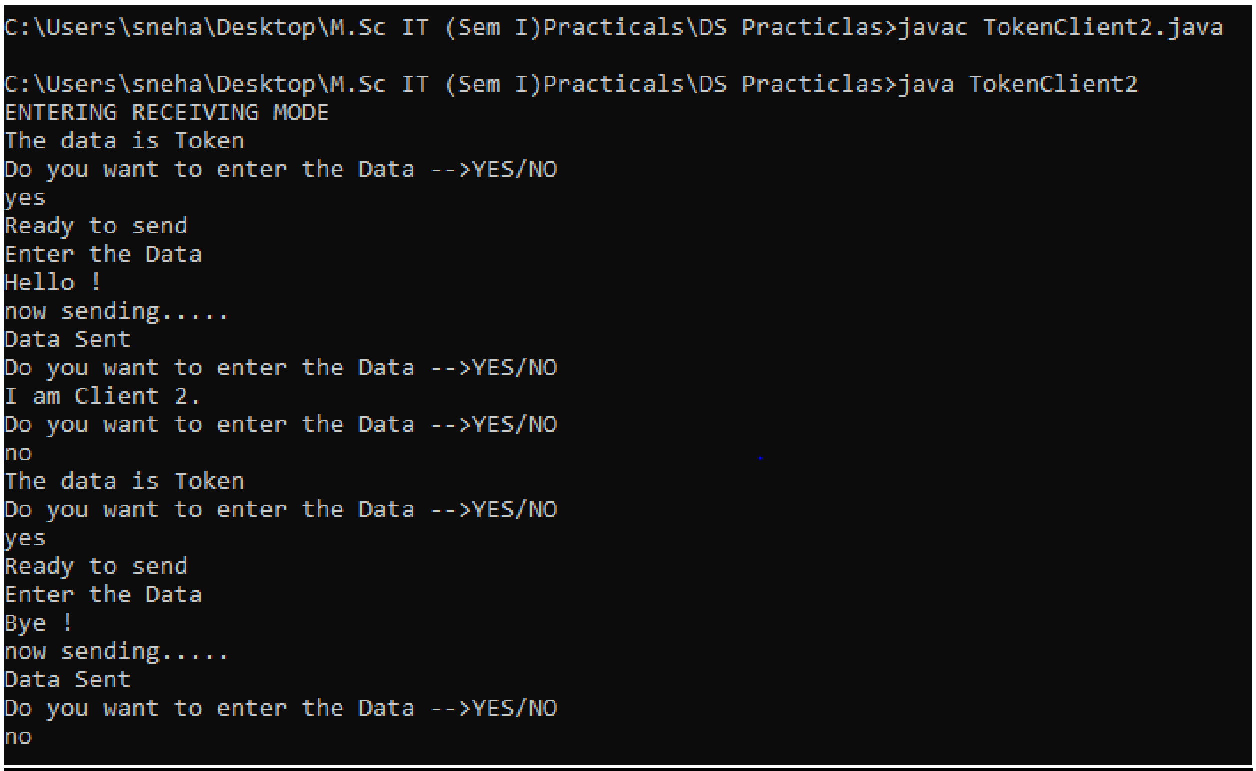
OUTPUT:



```
C:\Users\sneha\Desktop\M.Sc IT (Sem I)Practicals\DS Practiclas>javac TokenServer.java

C:\Users\sneha\Desktop\M.Sc IT (Sem I)Practicals\DS Practiclas>java TokenServer
The Message From :- ClientOne.....Hi...
The Message From :- ClientOne.....I am Client 1.
The Message From :- ClientTwo.....Hello !
The Message From :- ClientOne.....I am Talking to Server.
The Message From :- ClientTwo.....Bye !
```



```
C:\Users\sneha\Desktop\M.Sc IT (Sem I)Practicals\DS Practiclas>javac TokenClient1.java

C:\Users\sneha\Desktop\M.Sc IT (Sem I)Practicals\DS Practiclas>java TokenClient1
Do you want to enter the Data -->YES/NO
yes
ready to send
Enter the Data
Hi...
now sending.....
Data Sent
Do you want to enter the Data -->YES/NO
yes
ready to send
Enter the Data
I am Client 1.
now sending.....
Data Sent
Do you want to enter the Data -->YES/NO
no
The data is Token
Do you want to enter the Data -->YES/NO
yes
ready to send
Enter the Data
I am Talking to Server.
now sending.....
Data Sent
Do you want to enter the Data -->YES/NO
no
The data is Token
Do you want to enter the Data -->YES/NO
```



```
C:\Users\sneha\Desktop\M.Sc IT (Sem I)Practicals\DS Practiclas>javac TokenClient2.java

C:\Users\sneha\Desktop\M.Sc IT (Sem I)Practicals\DS Practiclas>java TokenClient2
ENTERING RECEIVING MODE
The data is Token
Do you want to enter the Data -->YES/NO
yes
Ready to send
Enter the Data
Hello !
now sending.....
Data Sent
Do you want to enter the Data -->YES/NO
I am Client 2.
Do you want to enter the Data -->YES/NO
no
The data is Token
Do you want to enter the Data -->YES/NO
yes
Ready to send
Enter the Data
Bye !
now sending.....
Data Sent
Do you want to enter the Data -->YES/NO
no
```

Practical 6

Write a program to implement any one election algorithm.

Election algorithms
Many distributed algorithms require the election of a special coordinator process; one that has a special role, or initiates something, or monitors something. Often it doesn't matter who the special process is, but one and only one must be elected, and it can't be known in advance who it will be. On a low-level you can think about the monitor in a token ring as an example.
The assumptions of these algorithms are that every process can be uniquely identified (by IP address, for example), and that each process can find out the id of the other processes. What the processes don't know is which processes are up and which are down at any given point in time.
Bully algorithm
Garcia-Molina, 1982. The process with the highest identity always becomes the coordinator.
When a process P sees that the coordinator is no longer responding to requests it initiates an election by sending ELECTION messages to all processes whose id is higher than its own. If no one responds to the messages then P is the new coordinator. If one of the higher-ups responds, it takes over and P doesn't have to worry anymore.
When a process receives an ELECTION message it sends a response back saying OK. It then holds its own election (unless it is already holding one). Eventually there is only one process that has not given up and that is the new coordinator. It is also the one with the highest number currently running. When the election is done the new coordinator sends a COORDINATOR message to everyone informing them of the change.
If a process which was down comes back up, it immediately holds an election. If this process had previously been the coordinator it will take this role back from whoever is doing it currently (hence the name of the algorithm).
Ring algorithm
Assumes that processes are logically ordered in some fashion, and that each process knows the order and who is coordinator. No token is involved. When a process notices that the coordinator is not responding it sends an ELECTION message with its own id to its downstream neighbor. If that neighbor doesn't respond it sends it to its neighbor's neighbor, etc. Each station that receives the ELECTION message adds its own id to the list. When the message circulates back to the originator it selects the highest id in the list and sends a COORDINATOR message announcing the new coordinator. This message circulates once and is removed by the originator.
If two elections are held simultaneously (say because two different processes notice simultaneously that the coordinator is dead) then each comes up with the same list and elects the same coordinator. Some time is wasted, but nothing is really hurt by this.

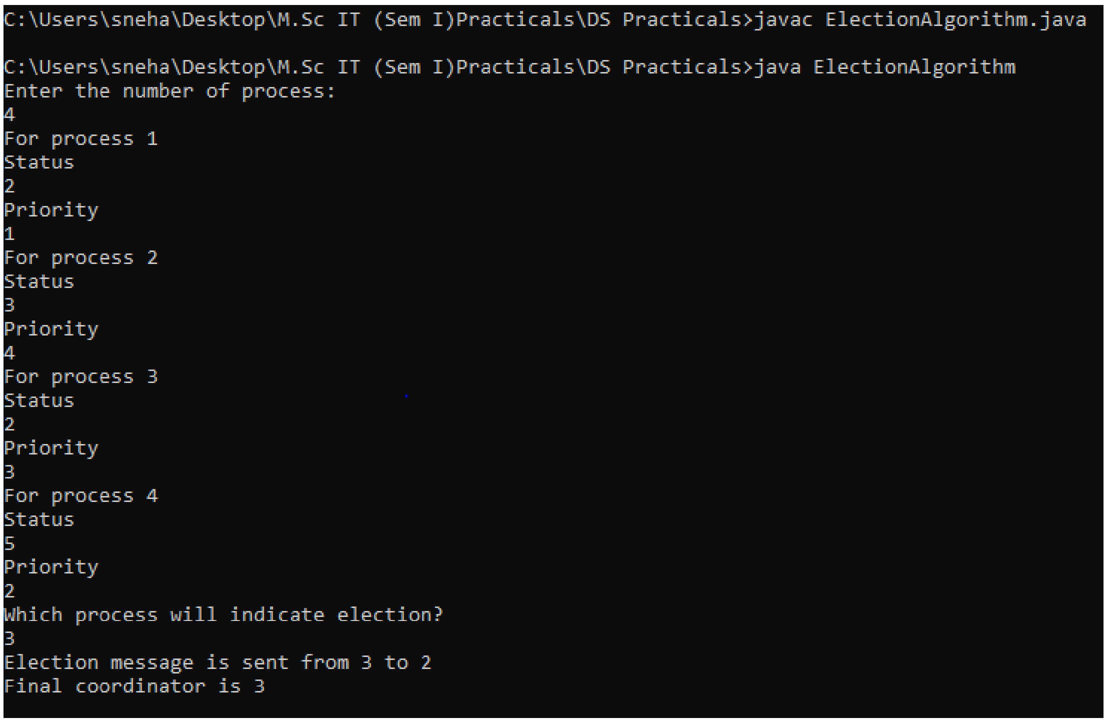
Q. 6. A) Write a program to implement any one one election algorithm.

CODE:

```
import java.io.*;
import java.util.Scanner;
class ElectionAlgorithm
{
	static int n;
	static int pro[]= new int[100];
```

```
        static int sta[]= new int[100];
        static int co;

        public static void main(String args[])throws IOException
        {
         System.out.println("Enter the number of process:");
         Scanner in = new Scanner(System.in);
         n = in.nextInt();
         int i;
         for(i=0;i<n;i++)
         {
                System.out.println("For process "+(i+1)+"");
                System.out.println("Status");
                sta[i] = in.nextInt();
                System.out.println("Priority");
                pro[i] = in.nextInt();
         }
         System.out.println("Which process will indicate election?");
         int ele = in.nextInt();
         elect(ele);
         System.out.println("Final coordinator is "+co);
        }
        static void elect(int ele)
        {
         ele = ele-1;
         co = ele+1;
         for(int i=0;i<n;i++)
         {
                if(pro[ele]<pro[i])
                {
                        System.out.println("Election message is sent from "+(ele+1)+" to "+(i+1));
                        if(sta[i]==1)
                                elect(i+1);
                }
         }
        }
}
```

OUTPUT:



```
C:\Users\sneha\Desktop\M.Sc IT (Sem I)Practicals\DS Practicals>javac ElectionAlgorithm.java

C:\Users\sneha\Desktop\M.Sc IT (Sem I)Practicals\DS Practicals>java ElectionAlgorithm
Enter the number of process:
4
For process 1
Status
2
Priority
1
For process 2
Status
3
Priority
4
For process 3
Status
2
Priority
3
For process 4
Status
5
Priority
2
Which process will indicate election?
3
Election message is sent from 3 to 2
Final coordinator is 3
```

Practical 7

Show the implementation of any one clock synchronization algorithm.

Clock synchronization algorithm:
Clock synchronization is a method of synchronizing clock values of any two nodes in a distributed system with the use of external reference clock or internal clock value of the node. During the synchronization, many factors effect on a network. As discussed above, these factors need to be considered before correcting actual clock value. Based on the approach, clock synchronization algorithms are divided as Centralized Clock Synchronization and Distributed Clock Synchronization Algorithm.
Distributed algorithm:

• There is particular time server.
• The processor periodically reach an agreement on the clock value by averaging the time of neighbor clock and its local clock.
• This can be used if no UTC receiver exist (no external synchronization is needed). Only internal synchronization is performed.
• Processes can run different machine and no global clock to judge which event happened first.

Q. 7. A) Show the implementation of any one clock synchronization algorithm.

CODE

SCServer.java

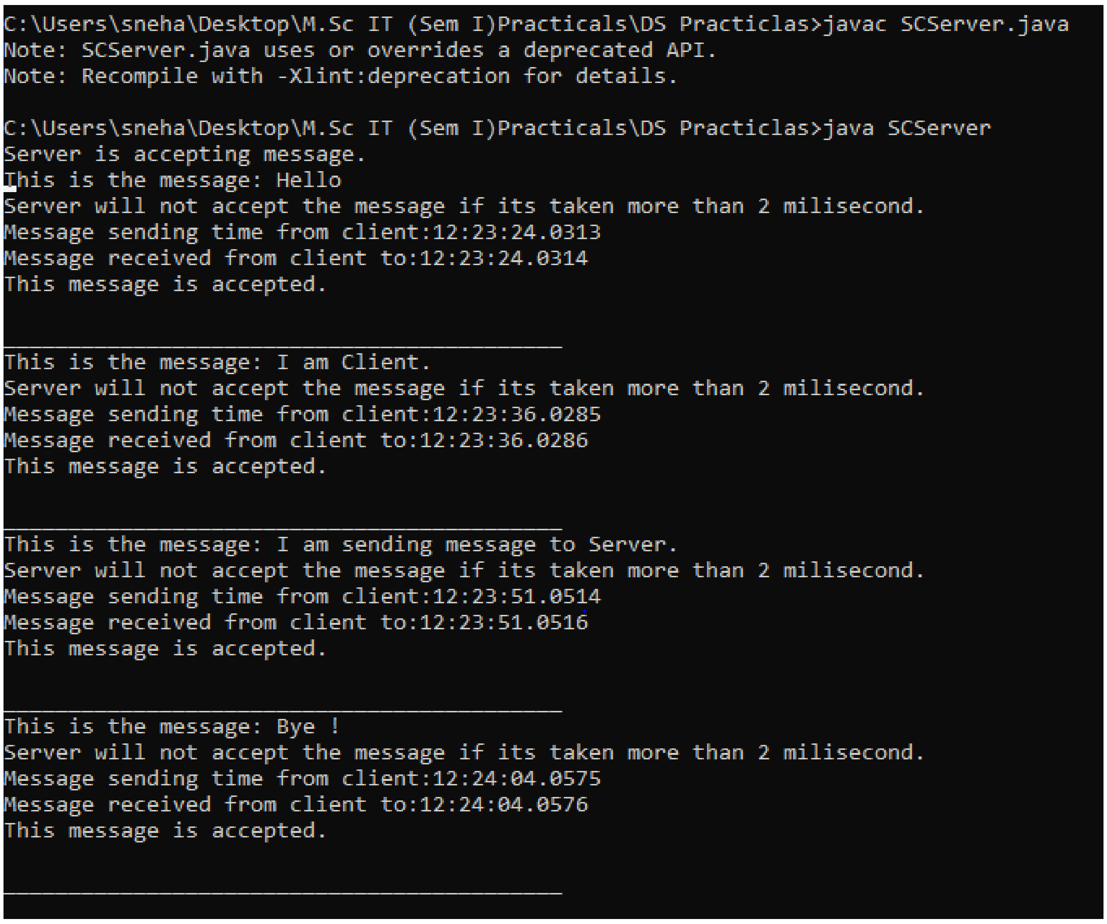
```
import java.net.*;
import java.io.*;
import java.text.DateFormat;
import java.text.SimpleDateFormat;
import java.util.Date;

public class SCServer {

    public static void main(String[] args) {

        final int PORT = 8001;

        try {

            ServerSocket serverSocket = new ServerSocket(PORT);
            System.out.println("Server is accepting message.");

            Socket socket = serverSocket.accept();

            DataInputStream inputStream = new DataInputStream(socket.getInputStream());

            String receieve;

            DateFormat dateFormat = new SimpleDateFormat("hh:mm:ss.SSSS");

            while ((receieve = inputStream.readLine()) != null) {
                String[] message = receieve.split(",");
                if (message[0].toLowerCase().equals("stop"))
                    break;

                Date date = new Date();

                Long clientTime = Long.parseLong(message[1]);
                Long timeMilli = date.getTime();

                Long requiredTime = timeMilli - clientTime;

                System.out.println("This is the message: " + message[0]);

                System.out.println("Server will not accept the message if its taken more than 2 
milisecond.");

                String formattedClientTime = dateFormat.format(clientTime);

                if (clientTime.equals(timeMilli)) {
                    String strDate = dateFormat.format(timeMilli);
                    System.out.println("Message sending time and receving time is same:" + strDate);
                } else if (requiredTime > 2) {
                    System.out.println("Message sending time from client:" + formattedClientTime);
                    String strDate = dateFormat.format(timeMilli);
                    System.out.println("Message received from client to:" + strDate);
                    System.out.println("This message is rejected.\n");
                   System.out.println("_______________________________________________");
                } else {
                    System.out.println("Message sending time from client:" + formattedClientTime);
                    String strDate = dateFormat.format(timeMilli);
                    System.out.println("Message received from client to:" + strDate);
                    System.out.println("This message is accepted.\n");
                   System.out.println("_______________________________________________");
                }
            }
            inputStream.close();

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

    }
}
```

SCClient.java

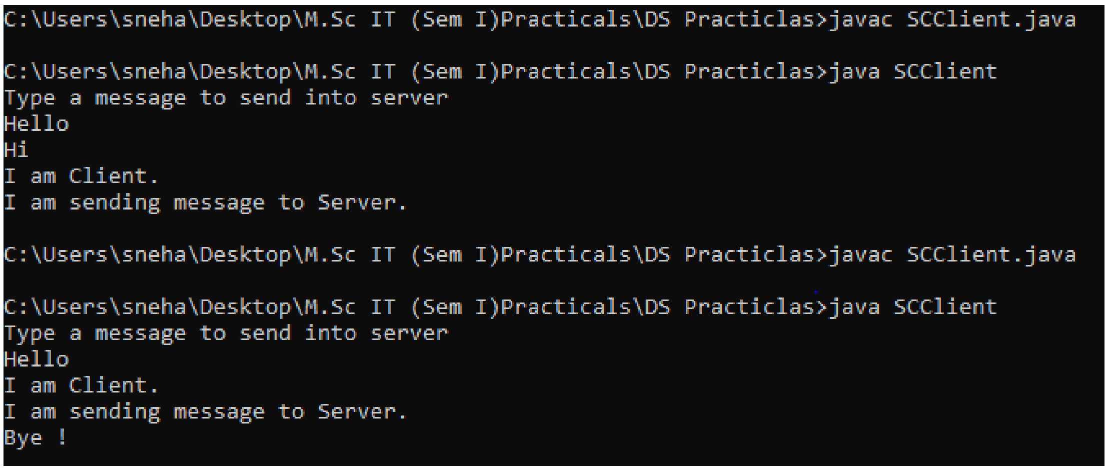
```
import java.net.*;
import java.io.*;
import java.util.Date;

public class SCClient {
    public static void main(String[] args) {
        final int PORT = 8001;
        final String HOST = "LocalHost";
        try {
            Socket socket = new Socket(HOST, PORT);
            BufferedReader inputBuffer = new BufferedReader(new InputStreamReader(System.in));
```

```
                DataOutputStream outputStream = new DataOutputStream(socket.getOutputStream());
                String send;
                System.out.println("Type a message to send into server");
                while ((send = inputBuffer.readLine()) != null) {
                    Date date = new Date();
                    long timeMilli = date.getTime();
                    String t = String.valueOf(timeMilli);
                    outputStream.writeBytes(send + "," + t + "\n");
                    if (send.toLowerCase().equals("stop"))
                        break;
                }
                inputBuffer.close();
                outputStream.close();
                socket.close();
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
}
```

OUTPUT:



```
C:\Users\sneha\Desktop\M.Sc IT (Sem I)Practicals\DS Practiclas>javac SCServer.java
Note: SCServer.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.

C:\Users\sneha\Desktop\M.Sc IT (Sem I)Practicals\DS Practiclas>java SCServer
Server is accepting message.
This is the message: Hello
Server will not accept the message if its taken more than 2 milisecond.
Message sending time from client:12:23:24.0313
Message received from client to:12:23:24.0314
This message is accepted.

___________________________________________
This is the message: I am Client.
Server will not accept the message if its taken more than 2 milisecond.
Message sending time from client:12:23:36.0285
Message received from client to:12:23:36.0286
This message is accepted.

___________________________________________
This is the message: I am sending message to Server.
Server will not accept the message if its taken more than 2 milisecond.
Message sending time from client:12:23:51.0514
Message received from client to:12:23:51.0516
This message is accepted.

___________________________________________
This is the message: Bye !
Server will not accept the message if its taken more than 2 milisecond.
Message sending time from client:12:24:04.0575
Message received from client to:12:24:04.0576
This message is accepted.

___________________________________________
```



```
C:\Users\sneha\Desktop\M.Sc IT (Sem I)Practicals\DS Practiclas>javac SCClient.java

C:\Users\sneha\Desktop\M.Sc IT (Sem I)Practicals\DS Practiclas>java SCClient
Type a message to send into server
Hello
Hi
I am Client.
I am sending message to Server.

C:\Users\sneha\Desktop\M.Sc IT (Sem I)Practicals\DS Practiclas>javac SCClient.java

C:\Users\sneha\Desktop\M.Sc IT (Sem I)Practicals\DS Practiclas>java SCClient
Type a message to send into server
Hello
I am Client.
I am sending message to Server.
Bye !
```

Practical 8
Write a program to implement two phase commit protocol

Two Phase Commit Protocol:

In transaction processing, databases and computer networking, the two phase commit protocol (2PC) is a type of atomic commitment protocol (ACP).
Basically, it is used to make sure the transactions are in sync when you have 2 or more DB's.
Assume I've two DBs (A & B) using 2PC in two different locations. Before A and B are ready to commit a transaction, both DBs will report back to the transaction manger saying they are ready to commit. So, when the transaction manager is acknowledge, it will send a signal back to A and B telling them to go ahead.
The steps performed in the two phases are as follows....

Phase 1: Prepare Phase
After each slave has locally completed its transaction, it sends a "DONE" message to the controlling site. When the controlling site has received DONE message from all slaves, it sends a "Prepare" message to the slaves.
The slaves vote on whether they still want to commit or not. If a slave wants to commit, it sends a "Ready" message.
A slave that does not want to commit sends a Not Ready" message. This may happen when the slave has conflicting concurrent transactions or there is a timeout
Phase 2: Commit/Abort Phase
After the controlling site has received "Ready" message from all the slaves-- The controlling site sends a "Global Commit" message to the slaves.
- The slaves apply the transaction and send a "Commit ACK" message to the controlling site.
- When the controlling site receives "Commit ACK" message from all the slaves, it considers the transaction as committed.

After the controlling site has received the first "Not Ready" message from any slave –
-The controlling site sends a "Global Abort" message to the slaves.
- The slaves abort the transaction and send a "Abort ACK message to the controlling site.
- Wen the controlling site receives "Abort ACK" message from all the slaves, it considers the transaction as aborted.

Two-Phase Commit Protocol follows following Steps:
Step1:
The Coordinator sends a VOTE_REQUEST message to all participants.
Step 2:

When a participant receives a VOTE_REQUEST message, it returns either a VOTE_COMMIT message to the coordinator telling the coordinator that is prepared to commit or a VOTE_ABORT message.

Step3:

The coordinator collects all votes from the participants. If all participants had voted to commit the transaction then it sends GLOBAL_COMMIT message to all participants. Even if one participant had voted to abort the transaction then it send GLOBAL_ABORT message.

Step 4:

Each participant that voted for a commit waits for the final reaction by the coordinator. If a participant receives a GLOBAL_COMMIT message it locally commits the transaction else it abort the local transaction.

Q. 8. A) Write a program to implement two phase commit protocol

CODE:

TPCServer.java

```
import java.io.*;
import java.net.*;

public class TPCServer{
   public static void main(String a[]) throws Exception{
            BufferedReader br;
            InetAddress lclhost;
            lclhost=InetAddress.getLocalHost();
            Server ser=new Server(lclhost);
            System.out.println("Server in sending mode....");
            //Sending data to client1
            ser.setSendPort(9000);    //recport=8000
            ser.setRecPort(8001);    //sendport=9001
            System.out.println("Send request data to client1....");
            br=new BufferedReader(new InputStreamReader(System.in));
            String s=br.readLine();
            System.out.println("Data is " +s);
            ser.sendData();
            System.out.println("Waiting for response from client1.....");
            ser.recData();
            //Sending data to client 2
            ser.setSendPort(9002);    //recport=8002
            ser.setRecPort(8003);    //senport=9003
            System.out.println("Send request data to client2...");
            br=new BufferedReader(new InputStreamReader(System.in));
            String s1=br.readLine();
            System.out.println("Data is " +s1);
            ser.sendData();
            System.out.println("Waiting for response from client2...");
```

```
                ser.recData();
                //Sending the final result to client 1
                ser.setSendPort(9000);
                  ser.sendData();
                  //Sending the final result to client 2
                  ser.setSendPort(9002);
                  ser.sendData();
      }
}
class Server{
InetAddress lclhost;
int sendPort,recPort;
int ssend=0;
int scounter=0;
Server(InetAddress lclhost)
{
            this.lclhost=lclhost;
}
public void setSendPort(int sendPort)
{
            this.sendPort=sendPort;
}
public void setRecPort(int recPort)
{
              this.recPort=recPort;
}
public void sendData() throws Exception
{
DatagramSocket ds;
DatagramPacket dp;
String data="";
if(scounter<2 && ssend<2)
{
                  data="VOTE_REQUEST";
}
if(scounter<2 && ssend>1)
{
  data="GLOBAL_ABORT";
  data=data+"TRANSACTION ABORTED";
}
if(scounter==2 && ssend>1){
data="GLOBAL_COMMIT";
data=data+"TRANSACTION COMMITTED";
}
ds=new DatagramSocket(sendPort);
dp=new DatagramPacket(data.getBytes(),data.length(), lclhost, sendPort-1000);
```

```
ds.send(dp);
ds.close();
ssend++;
}
public void recData() throws Exception
{
byte buf[]=new byte[256];
DatagramPacket dp=null;
DatagramSocket ds = null;
String msgStr="";
try{
ds=new DatagramSocket(recPort);
dp=new DatagramPacket(buf,buf.length);
ds.receive(dp);
ds.close();
}
catch(Exception e){
e.printStackTrace();
}
msgStr=new String(dp.getData(),0,dp.getLength());
System.out.println("String="+msgStr);
if(msgStr.equalsIgnoreCase("VOTE_COMMIT"))
{
scounter++;
}
System.out.println("Counter value ="+scounter + "n Send value = "+ssend);
}
}
```

<u>TPCClient1.java</u>

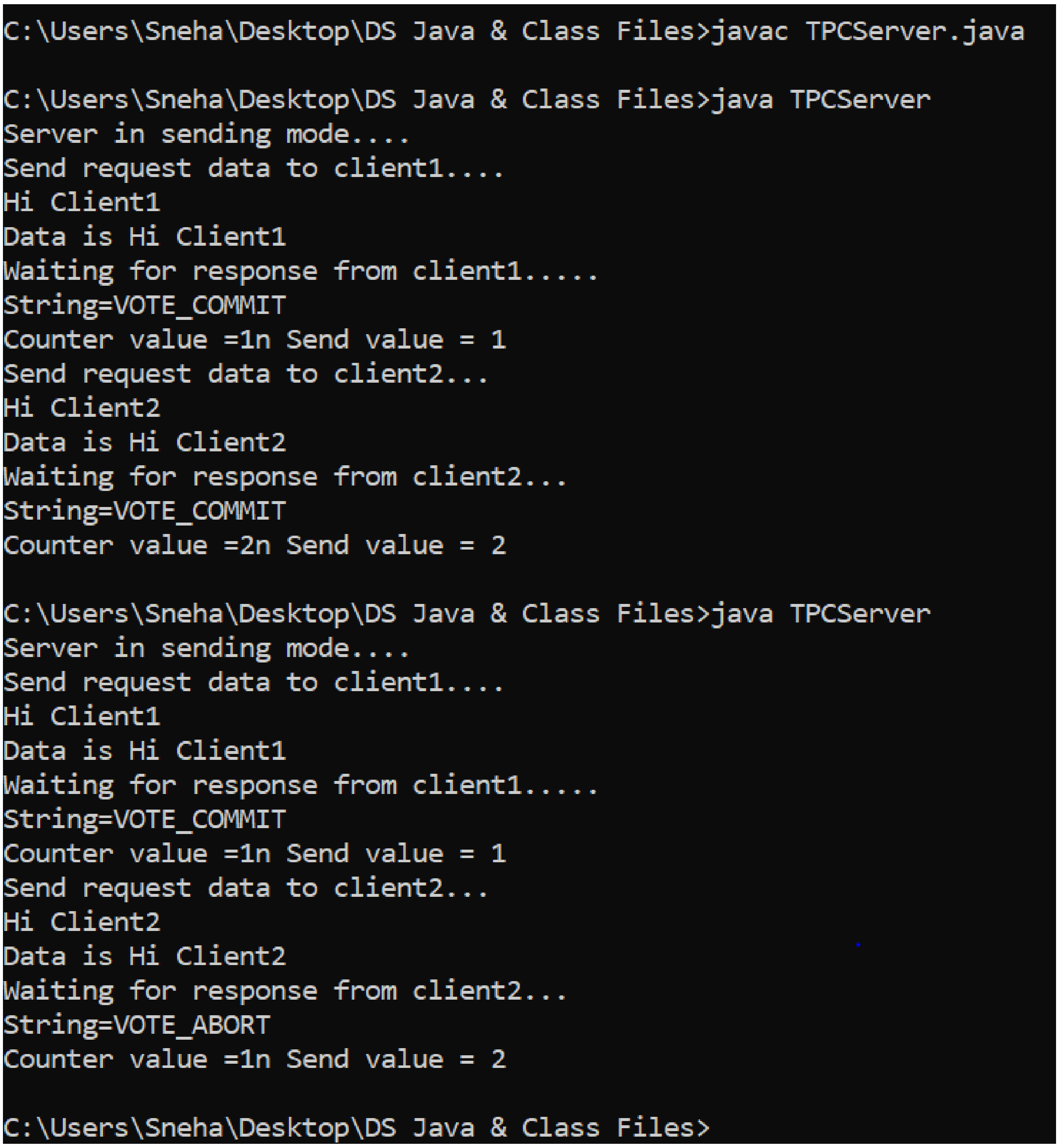
```
import java.io.*;
import java.net.*;
public class TPCClient1
{
   public static void main(String a[])throws Exception
   {
     InetAddress lclhost;
     lclhost=InetAddress.getLocalHost();
     Client clnt=new Client(lclhost);
     clnt.setSendPort(9001);
     clnt.setRecPort(8000);
     clnt.recData();
     clnt.sendData();
     clnt.recData();
   }
}
class Client{
   InetAddress lclhost;
     int sendPort,recPort;
     Client(InetAddress lclhost)
     {
     this.lclhost=lclhost;
     }
     public void setSendPort(int sendPort)
     {
     this.sendPort=sendPort;
     }

     public void setRecPort(int recPort)
     {
     this.recPort=recPort;
     }
     public void sendData()throws Exception
     {
     BufferedReader br;
     DatagramSocket ds;
     DatagramPacket dp;
     String data="";
System.out.println("Enter the Response 'VOTE_COMMIT'||'VOTE_ABORT'");
     br=new BufferedReader(new InputStreamReader(System.in));
     data = br.readLine();
     System.out.println("Data is : "+data);
     ds=new DatagramSocket(sendPort);
     dp=new
DatagramPacket(data.getBytes(),data.length(),lclhost,sendPort-1000);
     ds.send(dp);
     ds.close();
     }

     public void recData()throws Exception
     {
     byte buf[]=new byte[256];
     DatagramPacket dp;
     DatagramSocket ds;
     ds=new DatagramSocket(recPort);
     dp=new DatagramPacket(buf,buf.length);
     ds.receive(dp);
     ds.close();
     String msgStr=new
     String(dp.getData(),0,dp.getLength());
     System.out.println("Client1 data"+msgStr);
     }
}
```

TPCClient2.java

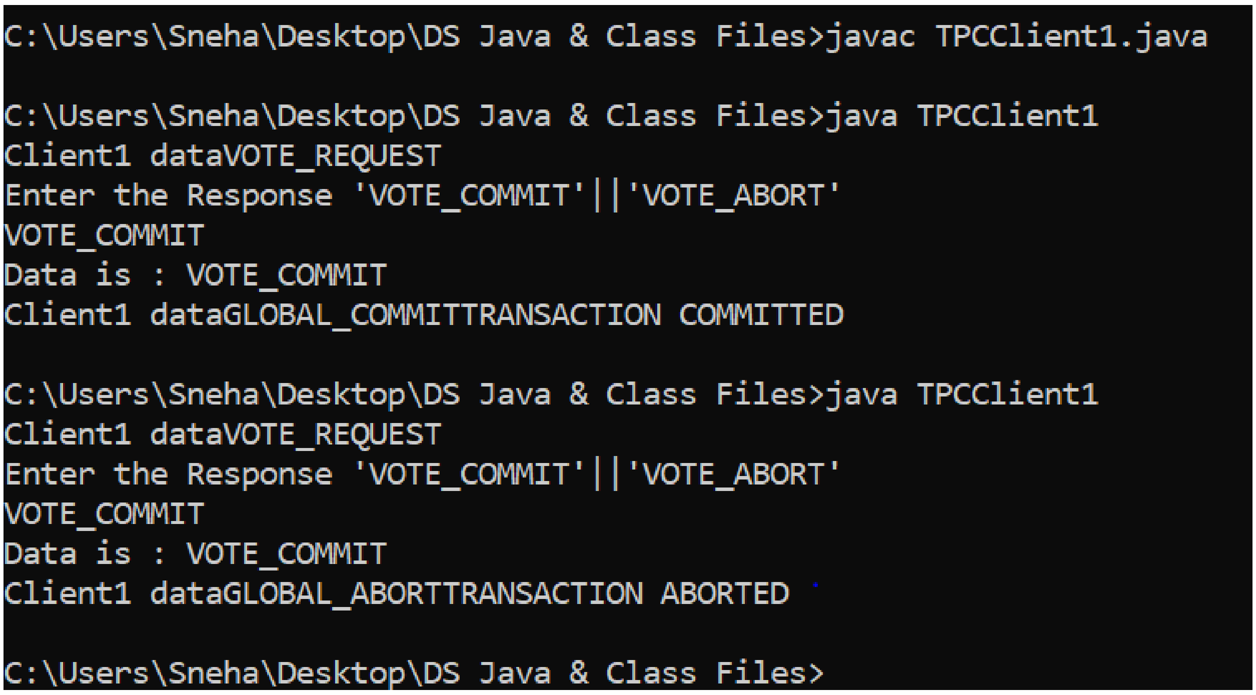
```
import java.io.*;
import java.net.*;
public class TPCClient2{
public static void main(String a[])throws Exception{
          InetAddress lclhost;
          lclhost = InetAddress.getLocalHost();
          Client client = new Client(lclhost);

          //Sending data to client2
          client.setSendPort(9003); //recport = 8002
          client.setRecPort(8002); //senport = 9003
          client.recData();
          client.sendData();
          client.recData();
          }
}

class Client
{
          InetAddress lclhost;
          int sendPort, recPort;
          Client(InetAddress lclhost)
          {
                    this.lclhost   = lclhost;
          }

          public void setSendPort(int sendPort)
          {
                    this.sendPort = sendPort;
          }
          public void setRecPort(int recPort)
          {
                    this.recPort = recPort;
          }

          public void sendData()throws Exception
          {
                    BufferedReader br;
                    DatagramSocket ds;
                    DatagramPacket dp;
```

```
                                String data = "";
System.out.println("Enter the Response 'VOTE_COMMIT'||'VOTE_ABORT'");
br = new BufferedReader(new InputStreamReader(System.in));
                                data = br.readLine();
                                System.out.println("Data is : " +data);
                                ds = new DatagramSocket(sendPort);
dp = new DatagramPacket(data.getBytes(), data.length(), lclhost, sendPort-1000);
                                ds.send(dp);
                                ds.close();
              }
              public void recData()throws Exception{
                                byte buf[] = new byte[256];
                                DatagramPacket dp;
                                DatagramSocket ds;
            ds = new DatagramSocket(recPort);
            dp = new DatagramPacket(buf, buf.length);
            ds.receive(dp);
            ds.close();
            String msgStr = new String(dp.getData(), 0, dp.getLength());
            System.out.println(msgStr);
          }
};
```

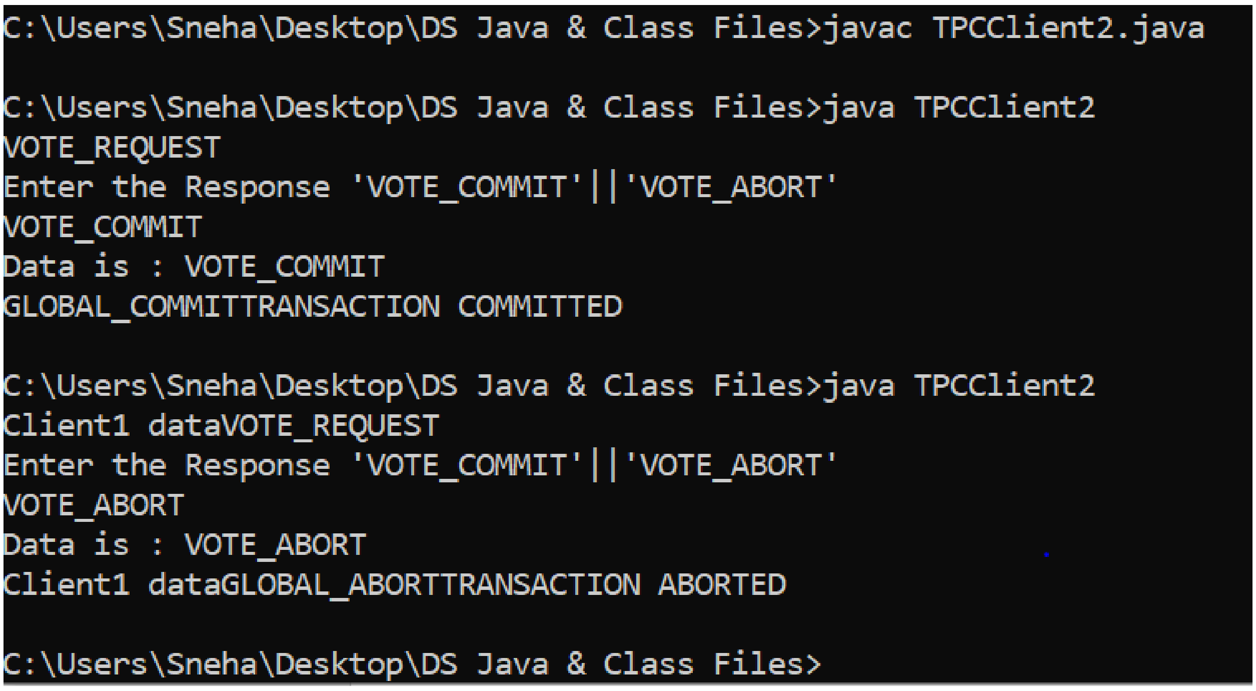
OUTPUT:



```
C:\Users\Sneha\Desktop\DS Java & Class Files>javac TPCServer.java

C:\Users\Sneha\Desktop\DS Java & Class Files>java TPCServer
Server in sending mode....
Send request data to client1....
Hi Client1
Data is Hi Client1
Waiting for response from client1.....
String=VOTE_COMMIT
Counter value =1n Send value = 1
Send request data to client2...
Hi Client2
Data is Hi Client2
Waiting for response from client2...
String=VOTE_COMMIT
Counter value =2n Send value = 2

C:\Users\Sneha\Desktop\DS Java & Class Files>java TPCServer
Server in sending mode....
Send request data to client1....
Hi Client1
Data is Hi Client1
Waiting for response from client1.....
String=VOTE_COMMIT
Counter value =1n Send value = 1
Send request data to client2...
Hi Client2
Data is Hi Client2
Waiting for response from client2...
String=VOTE_ABORT
Counter value =1n Send value = 2

C:\Users\Sneha\Desktop\DS Java & Class Files>
```



```
C:\Users\Sneha\Desktop\DS Java & Class Files>javac TPCClient1.java

C:\Users\Sneha\Desktop\DS Java & Class Files>java TPCClient1
Client1 dataVOTE_REQUEST
Enter the Response 'VOTE_COMMIT'||'VOTE_ABORT'
VOTE_COMMIT
Data is : VOTE_COMMIT
Client1 dataGLOBAL_COMMITTRANSACTION COMMITTED

C:\Users\Sneha\Desktop\DS Java & Class Files>java TPCClient1
Client1 dataVOTE_REQUEST
Enter the Response 'VOTE_COMMIT'||'VOTE_ABORT'
VOTE_COMMIT
Data is : VOTE_COMMIT
Client1 dataGLOBAL_ABORTTRANSACTION ABORTED

C:\Users\Sneha\Desktop\DS Java & Class Files>
```



```
C:\Users\Sneha\Desktop\DS Java & Class Files>javac TPCClient2.java

C:\Users\Sneha\Desktop\DS Java & Class Files>java TPCClient2
VOTE_REQUEST
Enter the Response 'VOTE_COMMIT'||'VOTE_ABORT'
VOTE_COMMIT
Data is : VOTE_COMMIT
GLOBAL_COMMITTRANSACTION COMMITTED

C:\Users\Sneha\Desktop\DS Java & Class Files>java TPCClient2
Client1 dataVOTE_REQUEST
Enter the Response 'VOTE_COMMIT'||'VOTE_ABORT'
VOTE_ABORT
Data is : VOTE_ABORT
Client1 dataGLOBAL_ABORTTRANSACTION ABORTED

C:\Users\Sneha\Desktop\DS Java & Class Files>
```