RAMNIRANJAN JHUNJHUNWALA COLLEGE

GHATKOPAR (W), MUMBAI - 400 086

DEPARTMENT OF INFORMATION TECHNOLOGY

2020 - 2021

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

Distributed System

Name: Deokar Gauri Rajendra

Roll No.: 07

Hindi Vidya Prachar Samiti's

RAMNIRANJAN JHUNJHUNWALA COLLEGE

(AUTONOMOUS)

Opposite Ghatkopar Railway Station, Ghatkopar West, Mumbai-400086

CERTIFICATE

This is to certify that Mr. / Miss/Mrs. **Gauri Deokar** with Roll No.**07** has

successfully completed the necessary course of experiments in the subject of

DISTRIBUTED SYSTEM during the academic year **2020 – 2021**

complying with the requirements of **RAMNIRANJAN JHUNJHUNWALA**

COLLEGE OF ARTS, SCIENCE AND COMMERCE, for the course of

M.Sc. (IT) semester – I.

Internal Examiner **Date**

HOD Of IT Department **College Seal** **External Examiner**

INDEX:

Practical No : 1

1(A) 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.

CODE :

ServerPrimeNumber.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 || x==2 || x ==3)
                        {
```

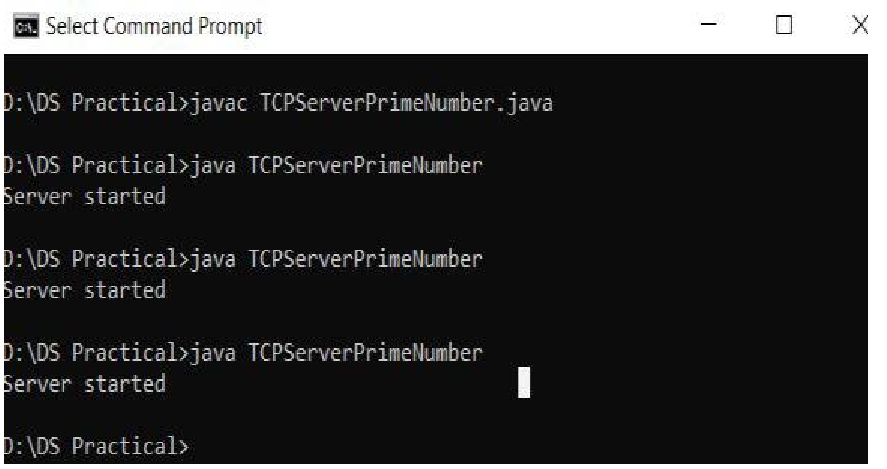
```
                    otc.writeUTF(x+"is Prime Number");
                    System.exit(0);
                }
                for(int i = 2; i <=y; i++)
                {
                    if( x % i != 0)
                    {
                        otc.writeUTF(x + "is Prime Number");
                    }
                    else
                    {
                        otc.writeUTF(x + "is not Prime Number");
                    }
                }
            }
            catch(Exception e)
            {
                System.out.println(e.toString());
            }
        }
}
```

ClientPrimeNumber.java

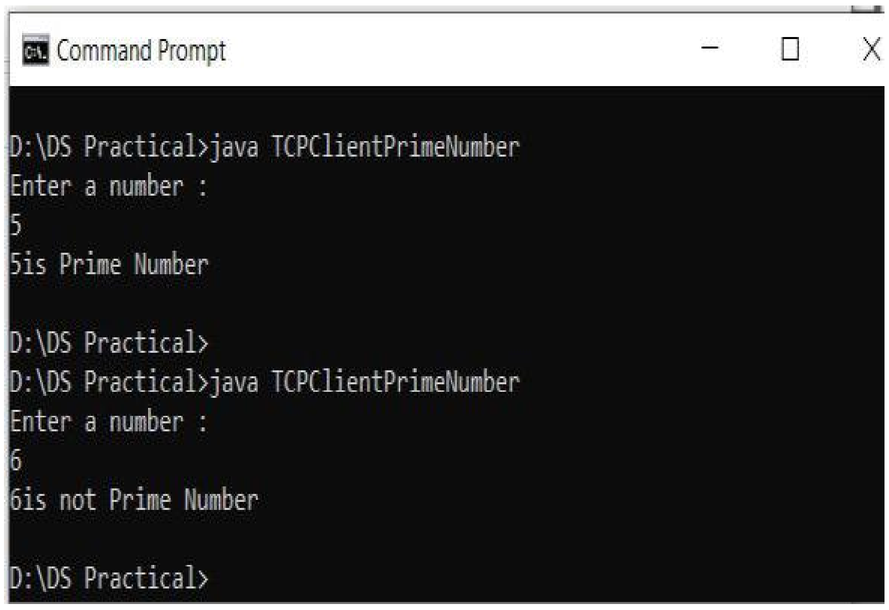
```
import java.net.*;
import java.io.*;

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

            }
```

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

OUTPUT :



```
D:\DS Practical>javac TCPServerPrimeNumber.java

D:\DS Practical>java TCPServerPrimeNumber
Server started

D:\DS Practical>java TCPServerPrimeNumber
Server started

D:\DS Practical>java TCPServerPrimeNumber
Server started

D:\DS Practical>
```



```
D:\DS Practical>java TCPClientPrimeNumber
Enter a number :
5
5is Prime Number

D:\DS Practical>
D:\DS Practical>java TCPClientPrimeNumber
Enter a number :
6
6is not Prime Number

D:\DS Practical>
```

1(B) A Client Server TCP based chat application

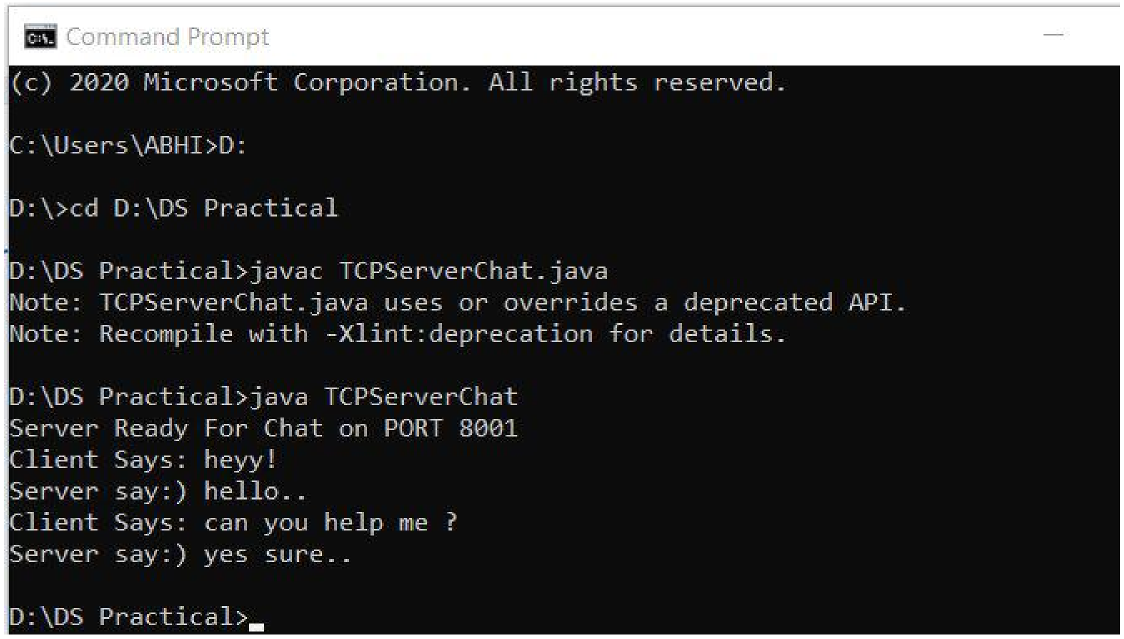
ServerChat.java

```
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());
        }
    }
}
```

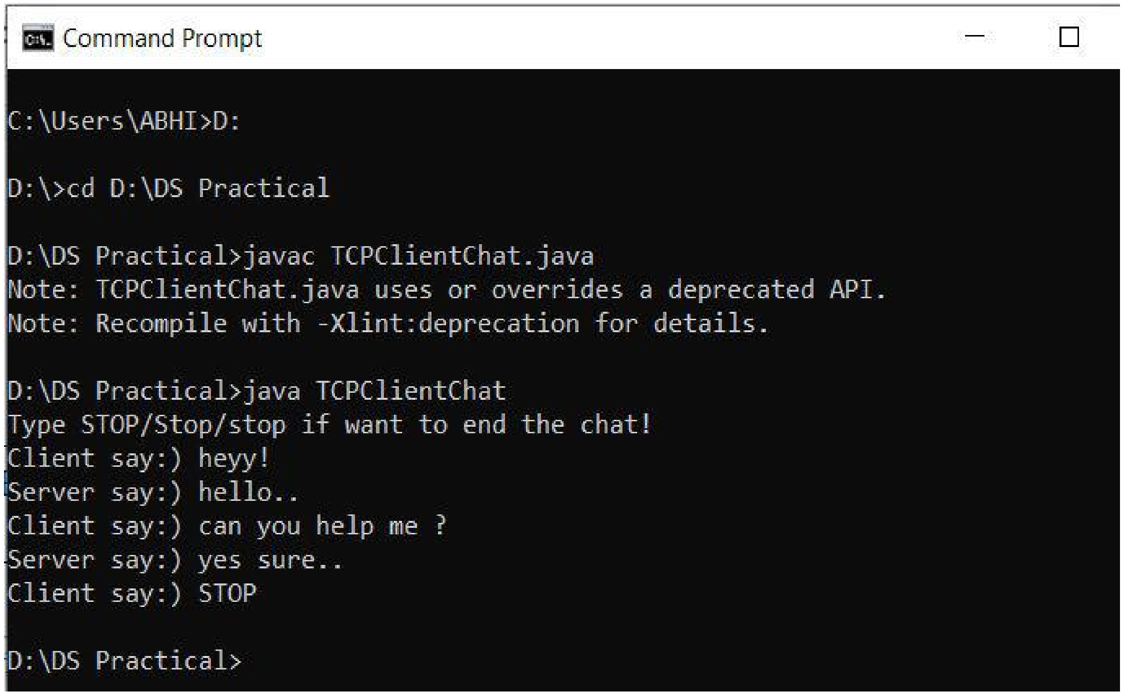
ClientChat.java

```
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) 2020 Microsoft Corporation. All rights reserved.

C:\Users\ABHI>D:

D:\>cd D:\DS Practical

D:\DS Practical>javac TCPServerChat.java
Note: TCPServerChat.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.

D:\DS Practical>java TCPServerChat
Server Ready For Chat on PORT 8001
Client Says: heyy!
Server say:) hello..
Client Says: can you help me ?
Server say:) yes sure..

D:\DS Practical>_
```



```

C:\Users\ABHI>D:

D:\>cd D:\DS Practical

D:\DS Practical>javac TCPClientChat.java
Note: TCPClientChat.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.

D:\DS Practical>java TCPClientChat
Type STOP/Stop/stop if want to end the chat!
Client say:) heyy!
Server say:) hello..
Client say:) can you help me ?
Server say:) yes sure..
Client say:) STOP

D:\DS Practical>
```

Practical No : 2

Aim: Write a program to show the object communication using RM**I.**

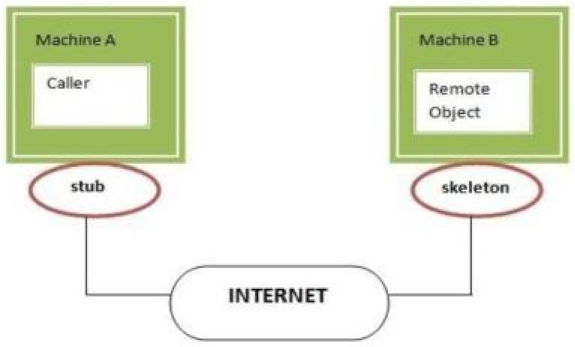
The RMI (Remote Method Invocation) is an API that provides a mechanism to create distributed application in java. The RMI allows an object to invoke methods on an object running in another JVM. The RMI provides remote communication between the applications using two objects *stub* and *skeleton*. A **remote 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.



2(A) : A RMI based application program to display current date and time.

CODE:

`InterfaceDate.java`

```
import java.rmi.*;

public interface InterfaceDate 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 InterfaceDate {
   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);
      }
   }
}
```

ClientDate.java

```
import java.rmi.*;
import java.rmi.registry.LocateRegistry;
import java.rmi.registry.Registry;
```

```
public class ClientDate {
    public static void main(String args[]) throws RemoteException {
        ClientDate c = new ClientDate();
        c.connectRemote();
    }

    private void connectRemote() throws RemoteException

    {
        try {
            String s1;
            Registry reg = LocateRegistry.getRegistry("localhost", 5678);
            InterfaceDate h1 = (InterfaceDate) reg.lookup("DS");
            s1 = h1.display();
            System.out.println(s1);
        } catch (NotBoundException | RemoteException e) {
            System.out.println("exception" + e);
        }
    }
}
```

2(B) : A RMI based application program that converts digits to words, e.g. 123 will be converted to one two three

CODE :

InterfaceConverter.java

```
import java.rmi.*;

public interface InterfaceConverter extends Remote {
    public String convertDigit(String no) throws RemoteException;;
}
```

ClientConverter.java

```
import java.rmi.*;
import java.io.*;
import java.rmi.registry.LocateRegistry;
import java.rmi.registry.Registry;

public class ClientConverter {
    public static void main(String args[]) throws RemoteException, IOException {
        ClientConverter c = new ClientConverter();
        c.connectRemote();
    }

    private void connectRemote() throws RemoteException, IOException {
        try {
            Registry reg = LocateRegistry.getRegistry("localhost", 5678);
            InterfaceConverter h1 = (InterfaceConverter) reg.lookup("Wrd");
            BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
            System.out.println("Enter a number : \t");
            String no = br.readLine();
            String ans = h1.convertDigit(no);
            System.out.println("The word representation of the entered digit is : " + ans);
        } catch (NotBoundException | RemoteException e) {
            System.out.println("exception" + e);
        }
    }
}
```

ServerConverter.java

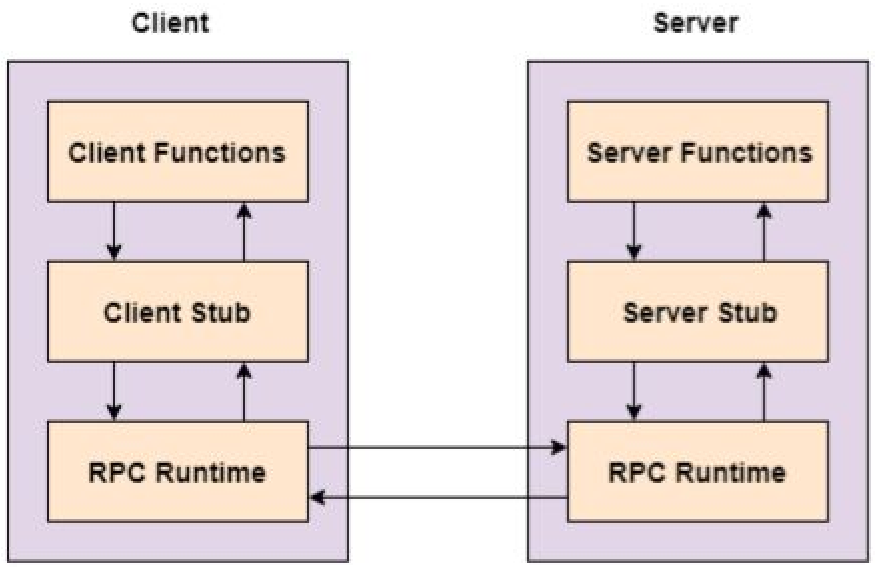
```
import java.rmi.*;
import java.rmi.registry.LocateRegistry;
import java.rmi.registry.Registry;
import java.rmi.server.*;
```

```
public class ServerConverter extends UnicastRemoteObject implements InterfaceConverter {
    public ServerConverter() throws RemoteException {
    }

    public String convertDigit(String no) throws RemoteException {
        String str = "";
        for (int i = 0; i < no.length(); i++) {
            int p = no.charAt(i);
            switch (p) {
                case 48:
                    str += "zero ";
                    break;
                case 49:
                    str += "one ";
                    break;
                case 50:
                    str += "two ";
                    break;
                case 51:
                    str += "three ";
                    break;
                case 52:
                    str += "four ";
                    break;
                case 53:
                    str += "five ";
                    break;
                case 54:
                    str += "six ";
                    break;
                case 55:
                    str += "seven ";
                    break;
                case 56:
                    str += "eight ";
                    break;
                case 57:
                    str += "nine ";
                    break;

            }
        }
        return str;
    }

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

Practical No : 3

3(A) Show the implementation of Remote Procedure Call

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.

CODE :

<u>**RPCServer.java**</u>

```
import java.io.*;

import java.net.*;

import java.util.StringTokenizer;


final class RCPServer

{

        DatagramSocket ds;

        DatagramPacket dp;

        String str,methodName,result;

        int val1,val2;

        RCPServer()

        {

                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"))
{
```

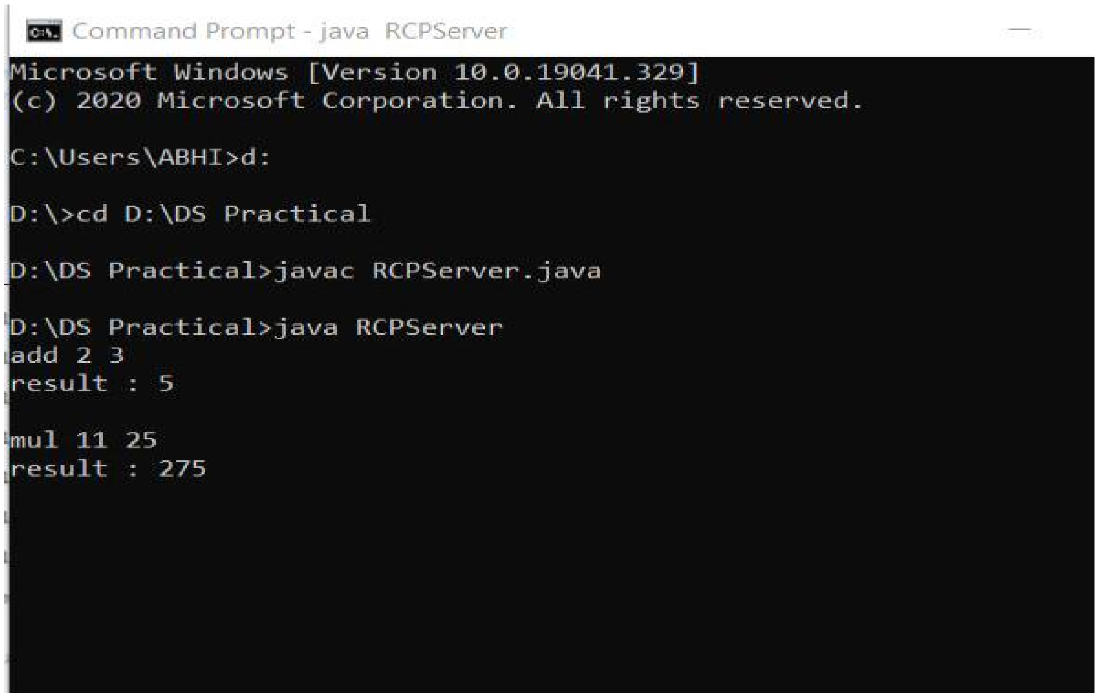
```
                        result= "" + add(val1,val2);
                    }
                    else if(methodName.equalsIgnoreCase("sub"))
                    {
                        result= "" + sub(val1,val2);
                    }
                    else if(methodName.equalsIgnoreCase("mul"))
                    {
                        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 val3, int val4)
                {
                        return val3-val4;
                }
                public int mul(int val5, int val6)
                {
                        return val5*val6;
                }
                public int div(int val7, int val8)
                {
                        return val7/val8;
                }
                public static void main(String[] args)
                {
                        new RCPServer();
                }
}
```

RPCClient.java

```
import java.io.*;

import java.net.*;


class RCPClient

{

	RCPClient()

	{

		try

		{

			InetAddress ia = InetAddress.getLocalHost();

			DatagramSocket ds =new DatagramSocket();

			DatagramSocket ds1 = new DatagramSocket(1300);

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

			System.out.println("Enter method name and parameters like add
num1 and 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);
```

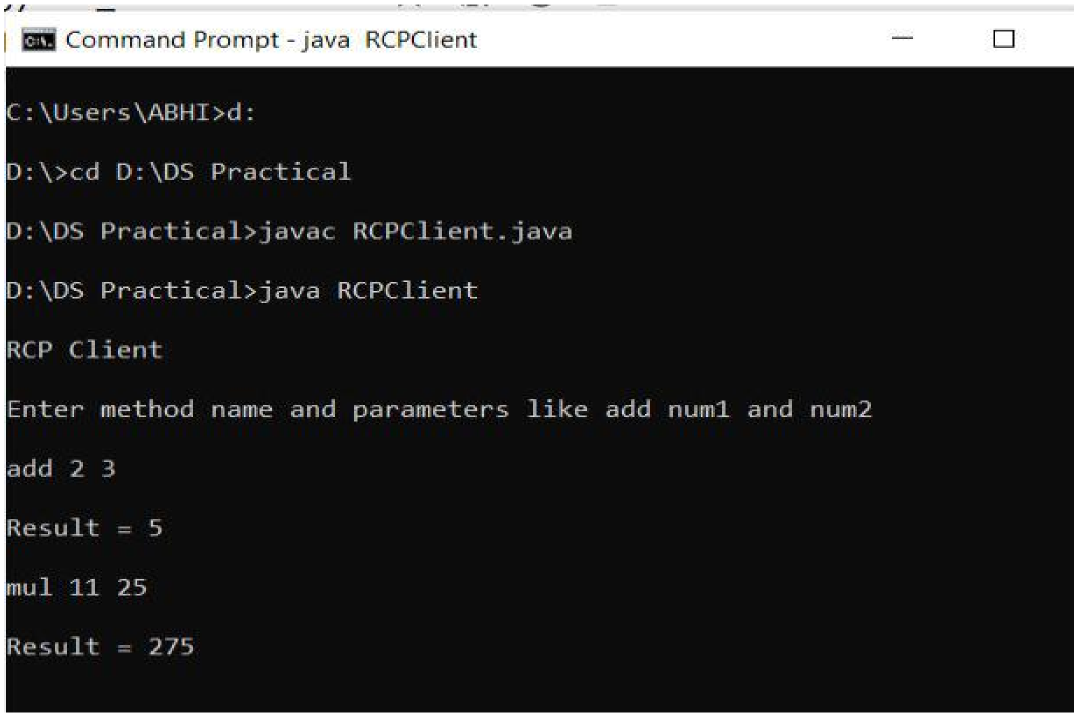
```
                    ds.send(dp);

                    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(IOException e)

            {

                e.printStackTrace();

            }

        }

        public static void main(String[] args)

        {

            new RCPClient();

        }

}
```

Output



Command Prompt - java RCPServer

```
Microsoft Windows [Version 10.0.19041.329]
(c) 2020 Microsoft Corporation. All rights reserved.

C:\Users\ABHI>d:

D:\>cd D:\DS Practical

D:\DS Practical>javac RCPServer.java

D:\DS Practical>java RCPServer
add 2 3
result : 5

mul 11 25
result : 275
```



Command Prompt - java RCPClient

```
C:\Users\ABHI>d:

D:\>cd D:\DS Practical

D:\DS Practical>javac RCPClient.java

D:\DS Practical>java RCPClient

RCP Client

Enter method name and parameters like add num1 and num2

add 2 3

Result = 5

mul 11 25

Result = 275
```

3(B) A program that finds the square, square root, cube and cube root of the entered number using Remote Procedure Call

RCPServer.java

```
import java.io.*;

import java.net.*;

import java.util.StringTokenizer;


final class RCPServer

{

        DatagramSocket ds;

        DatagramPacket dp;

        String str,methodName,result;

        int val;

        RCPServer()

        {

                try

                {

                        ds=new DatagramSocket(1200);

                        byte b[]=new byte[4096];

                        while(true)

                        {

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

                                ds.receive(dp);
```

```
str=new String(dp.getData(),0,dp.getLength());
if(str.equalsIgnoreCase("q"))
{
        System.exit(1);
}
else
{
        StringTokenizer st = new StringTokenizer(str," ");
        int i=0;

        while(st.hasMoreTokens())
        {
                String token = st.nextToken();
                methodName=token;
                val = Integer.parseInt(st.nextToken());

        }
}
System.out.println(str);

InetAddress ia = InetAddress.getLocalHost();
if(methodName.equalsIgnoreCase("square"))
{
```

```
                    result= "" + square(val);
                }
                else if(methodName.equalsIgnoreCase("squareroot"))
                {
                    result= "" + squareroot(val);
                }
                else if(methodName.equalsIgnoreCase("cube"))
                {
                    result= "" + cube(val);
                }
                else if(methodName.equalsIgnoreCase("cuberoot"))
                {
                    result= "" + cuberoot(val);
                }
                byte b1[]=result.getBytes();
                DatagramSocket ds1 = new DatagramSocket();
                DatagramPacket dp1 = new
DatagramPacket(b1,b1.length,InetAddress.getLocalHost(), 1300);
                System.out.println("result : "+result+"\n"); ds1.send(dp1);


            }
        }
        catch(Exception e)
        {
```

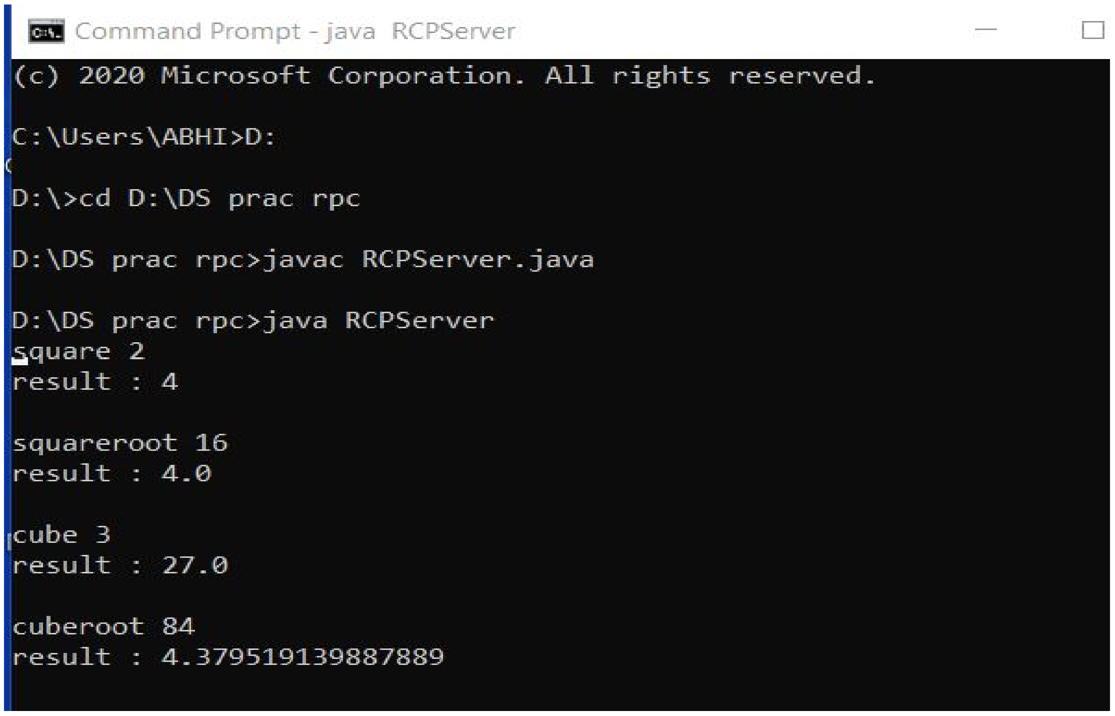
```
                        e.printStackTrace();
        }


    }
    public int square(int val)
            {
                        return val*val;
            }
            public double squareroot(int val)
            {
                        float temp,sqrt;
                        sqrt=val/2;
                        temp=0;

                        while(sqrt != temp)
                        {
                                    temp=sqrt;
                                    sqrt=(val/temp + temp)/2;
                        }
                        return sqrt;
            }
            public double cube(int val)
            {
                        return Math.pow(val ,3);

            }

            public double cuberoot(int val)

            {

                        return Math.pow(val ,1.0/3.0);

            }

            public static void main(String[] args)

            {

                        new RCPServer();

            }

    }
```

RCPClient.java

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

class RCPClient
{
	RCPClient()
	{
		try
		{
			InetAddress ia = InetAddress.getLocalHost();
			DatagramSocket ds =new DatagramSocket();
			DatagramSocket ds1 = new DatagramSocket(1300);
			System.out.println("\nRCP Client\n");
			System.out.println("Enter method name : 1. square ,2. squareroot
,3.cube ,4.cuberoot\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);
				if(str.equalsIgnoreCase("q"))
				{
					System.exit(1);
				}
				else
				{
				dp = new DatagramPacket(b,b.length);
				ds1.receive(dp);
				String s = new String(dp.getData(),0,dp.getLength());
				System.out.println("\nResult = " + s + "\n");
				}
			}
		}
```

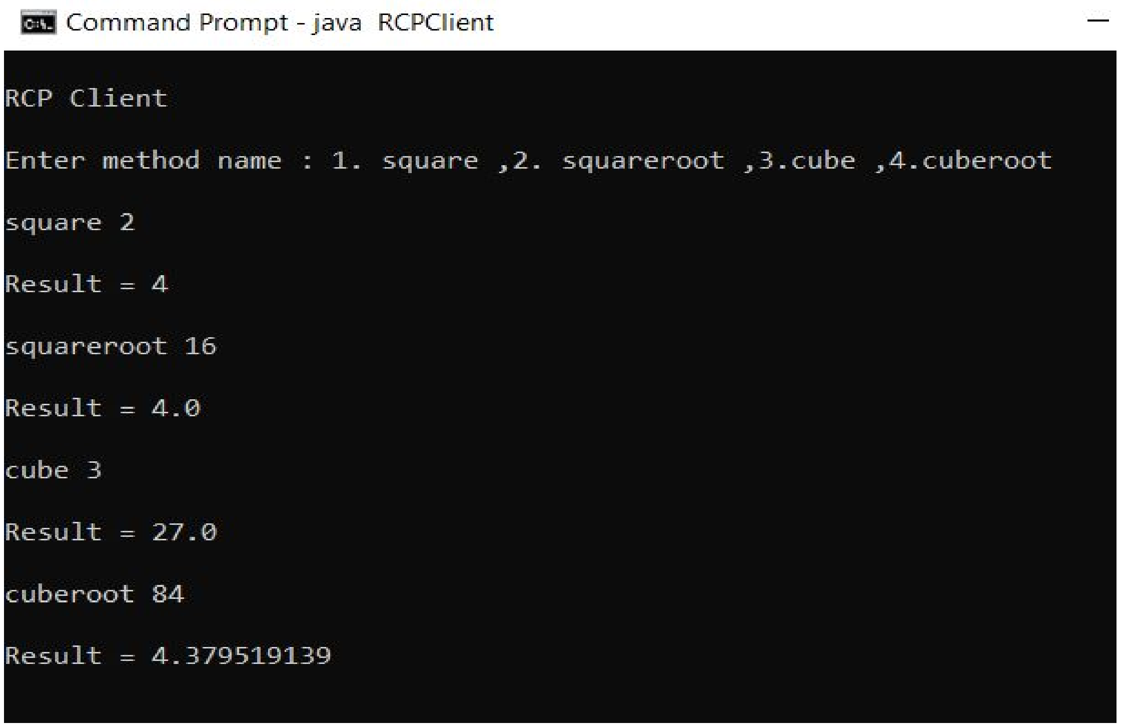
```
                catch(IOException e)
                {
                        e.printStackTrace();
                }
        }
        public static void main(String[] args)
        {
                new RCPClient();
        }
}
```

Output



Command Prompt - java RCPServer

```
(c) 2020 Microsoft Corporation. All rights reserved.

C:\Users\ABHI>D:

D:\>cd D:\DS prac rpc

D:\DS prac rpc>javac RCPServer.java

D:\DS prac rpc>java RCPServer
square 2
result : 4

squareroot 16
result : 4.0

cube 3
result : 27.0

cuberoot 84
result : 4.379519139887889
```



Command Prompt - java RCPClient

```
RCP Client

Enter method name : 1. square ,2. squareroot ,3.cube ,4.cuberoot

square 2

Result = 4

squareroot 16

Result = 4.0

cube 3

Result = 27.0

cuberoot 84

Result = 4.379519139
```

Practical No : 4

Aim: 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.

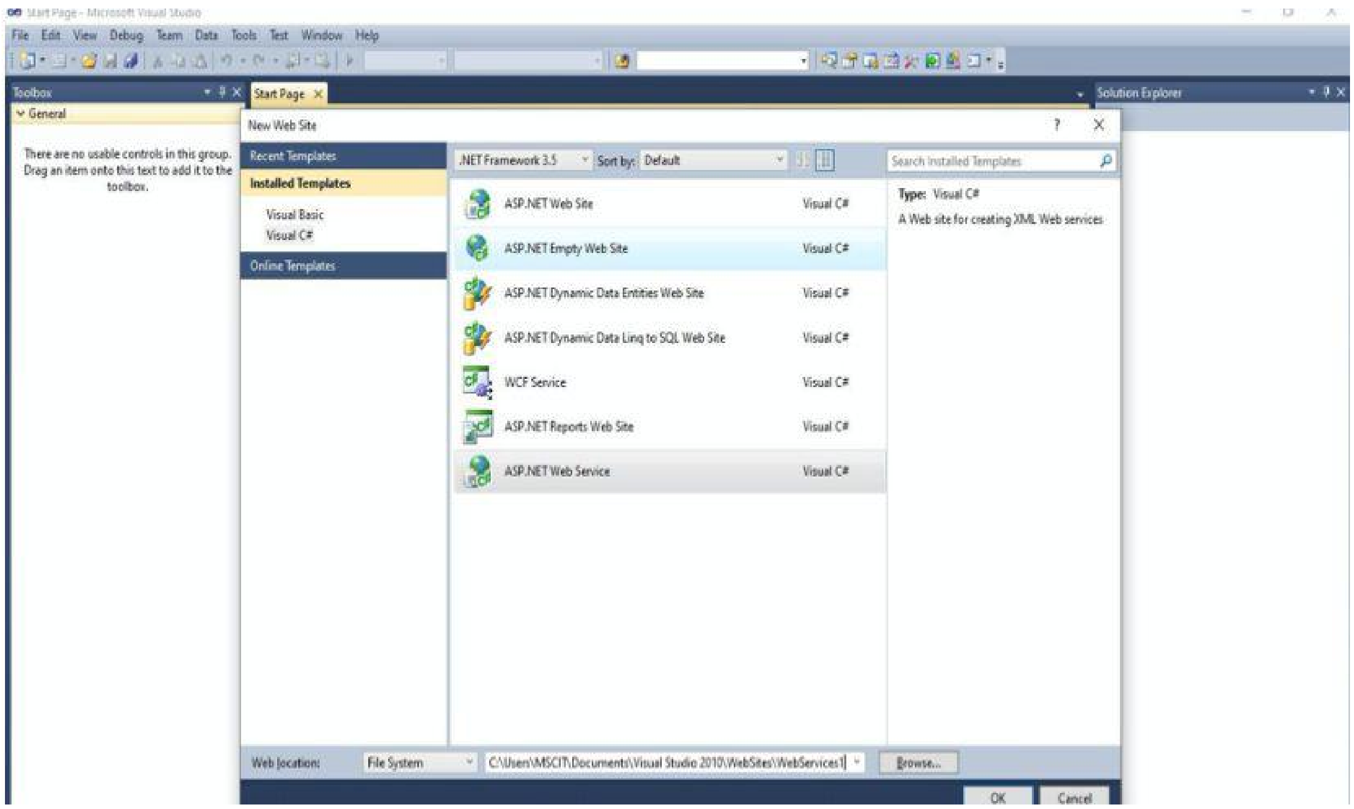
A web service can perform almost any kind of task
1. Web Portal- A web portal might obtain top news headlines from an associated press web service. 2. Weather Reporting- It can use as Weather reporting web service to display weather information in your personal website. 3. Stock Quote- It can display 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.

Step 1- Create the ASP.NET Web Service Source File

Step 1- Create the ASP.NET Web Service Source File

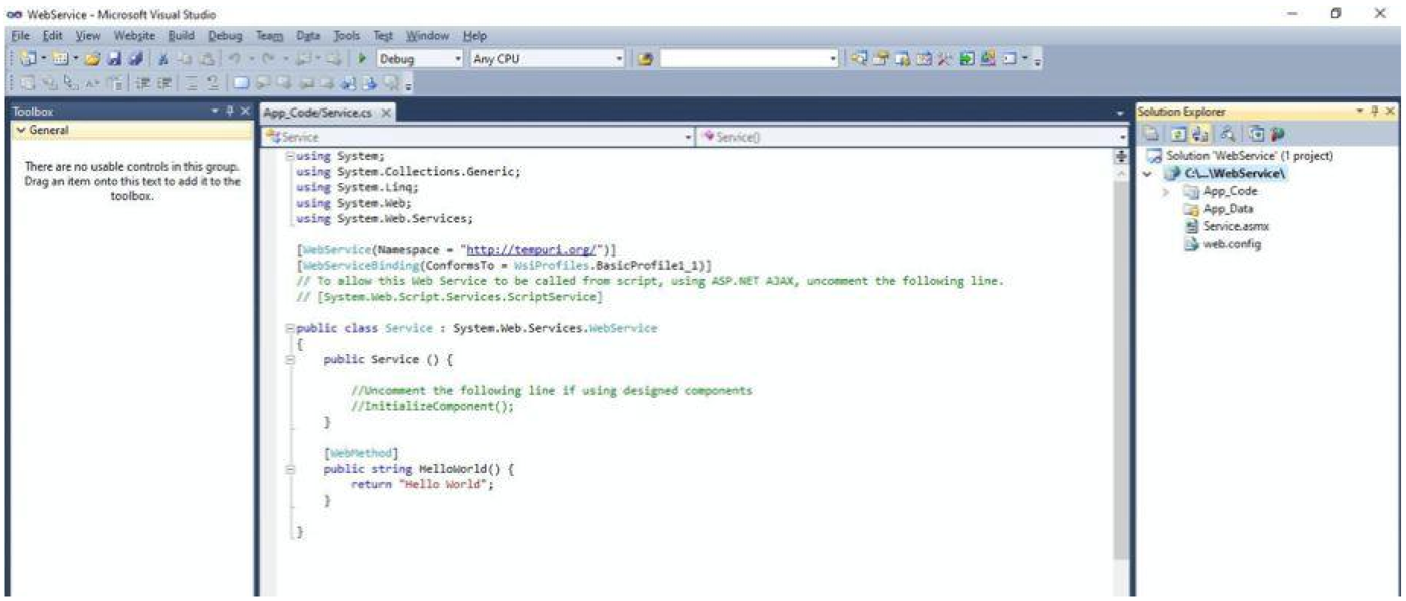
Open Visual Studio and create a new web site.->Select File ->Select WebSie -> Then select ASP.NET Web Service -> Ok


Start Page - Microsoft Visual Studio
File Edit View Debug Team Data Tools Test Window Help
Toolbox
General
There are no usable controls in this group. Drag an item onto this text to add it to the toolbox.
Start Page
Solution Explorer
New Web Site
Recent Templates
Installed Templates
Visual Basic
Visual C#
Online Templates
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#
Browse...
OK
Cancel

.NET Framework 3.5 Sort by: Default

Type: Visual C#
A Web site for creating XML Web services

Web location: File System C:\Users\MSCIT\Documents\Visual Studio 2010\WebSites\WebServices1


WebService - Microsoft Visual Studio
File Edit View Website Build Debug Team Data Tools Test Window Help
Debug
Any CPU
Toolbox
General
There are no usable controls in this group. Drag an item onto this text to add it to the toolbox.
App_Code/Service.cs
Service
Service()
Solution Explorer
Solution 'WebService' (1 project)
C:\...\WebService\
App_Code
App_Data
Service.asmx
web.config

```
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)]
// To allow this Web Service to be called from script, using ASP.NET AJAX, uncomment the following line.
// [System.Web.Script.Services.ScriptService]

public class Service : System.Web.Services.WebService
{
    public Service () {

        //Uncomment the following line if using designed components
        //InitializeComponent();
    }

    [WebMethod]
    public string HelloWorld() {
        return "Hello World";
    }

}
```

Now, Write the Arithmetic Services.

Solution:

Service.cs

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)]

// To allow this Web Service to be called from script, using ASP.NET AJAX, uncomment the following line.

// [System.Web.Script.Services.ScriptService]

public class Service : System.Web.Services.WebService {

public Service () {

//Uncomment the following line if using designed components

//InitializeComponent(); }

```
[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 the webservice and copy the path.

Then create empty website and then right click on the website -> Now click on Add Web References-> Then Paste the copied path.

Write Arithmetic program;

Default.aspx

```
<%@ Page Language="C#" AutoEventWireup="true" Inherits="_Default" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML
"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>

CodeFile="Default.aspx.cs"
```

```
1.0 Transitional//EN"

<form id="form1" runat="server"> <div>

<table border="2" cellpadding="2" cellspacing="2"> <tr>

<td align="right"> <asp:Label

Number"></asp:Label> </td>

ID="Label1"

runat="server"

Text="Enter 1st

<td align="left">

<asp:TextBox ID="txtFno" runat="server"></asp:TextBox>

</td> </tr>

<tr>

<td align="right">

<asp:Label Number"></asp:Label>

</td>

<td align="left">

ID="Label2"

runat="server"

Text="Enter 2nd

<asp:TextBox ID="txtSno" runat="server"></asp:TextBox> </td>

</tr> <tr>

<td align="right">

<asp:Label ID="Label3" runat="server" Text="Result"></asp:Label>

</td>
```

```
<td align="left">

<asp:Label ID="lblResult" runat="server"></asp:Label> </td>

</tr> <tr>

<td align="center"> <asp:Button

OnClick="btnAdd_Click" /> </td>

<td align="center"> <asp:Button

OnClick="btnSub_Click" /> </td>

</tr> <tr>

<td align="center"> <asp:Button

OnClick="BtnMul_Click" />

ID="btnAdd"

ID="btnSub"

ID="BtnMul"

runat="server"

runat="server"

runat="server"

Text="Add(+)"

Text="Sub(-)"

Text="Mul(*)"

</td>

<td align="center">

<asp:Button OnClick="btnDiv_Click" />

</td> </tr>

</table> </div>
```

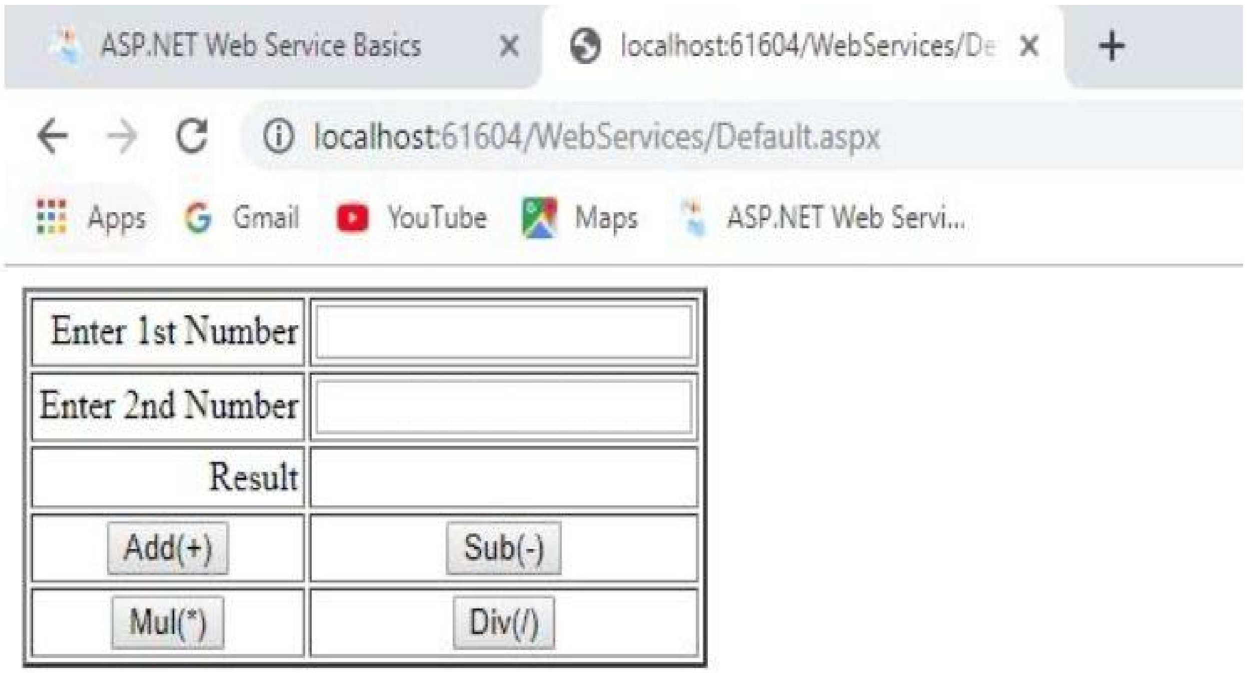
```
</form> </body>

</html>
```

Default.aspx.cs

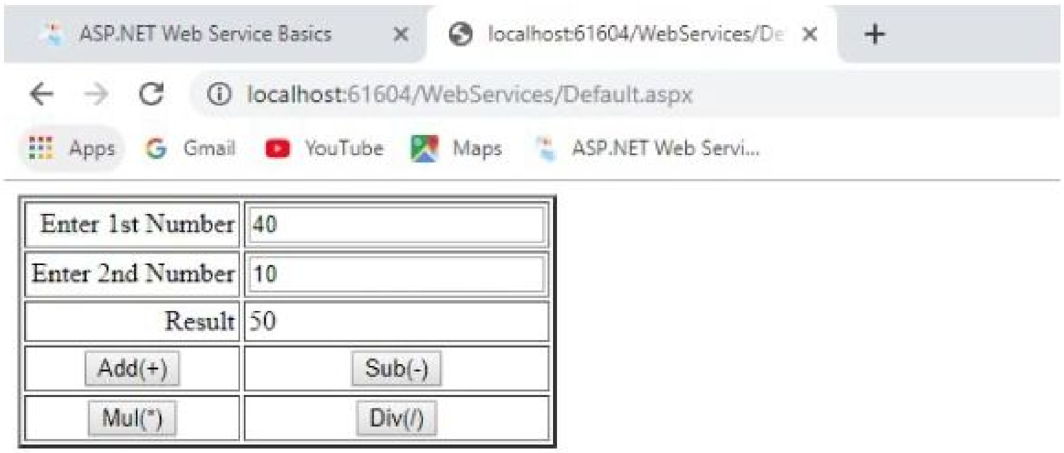
```
using System;

using System.Collections.Generic; using System.Linq; using System.Web; using
System.Web.UI;

using System.Web.UI.WebControls;

ID="btnDiv"

runat="server"

Text="Div(/)"

public partial class _Default : System.Web.UI.Page {

Service obj = new Service();

int a, b, c;

protected void Page_Load(object sender, EventArgs e) {

}

protected void btnAdd_Click(object sender, EventArgs e) {

a = Convert.ToInt32(txtFno.Text); b = Convert.ToInt32(txtSno.Text); obj.Add(a, b);

lblResult.Text = c.ToString(); }

c =

protected void btnSub_Click(object sender, EventArgs e)

{

a = Convert.ToInt32(txtFno.Text);

b = Convert.ToInt32(txtSno.Text); c = obj.Sub(a, b);

lblResult.Text = c.ToString(); }

protected void BtnMul_Click(object sender, EventArgs e) {
```

```
a = Convert.ToInt32(txtFno.Text);

b = Convert.ToInt32(txtSno.Text); c = obj.Mul(a, b);

lblResult.Text = c.ToString(); }

protected void btnDiv_Click(object sender, EventArgs e) {

a = Convert.ToInt32(txtFno.Text); b = Convert.ToInt32(txtSno.Text); obj.Div(a, b);

lblResult.Text = c.ToString(); }

}
```

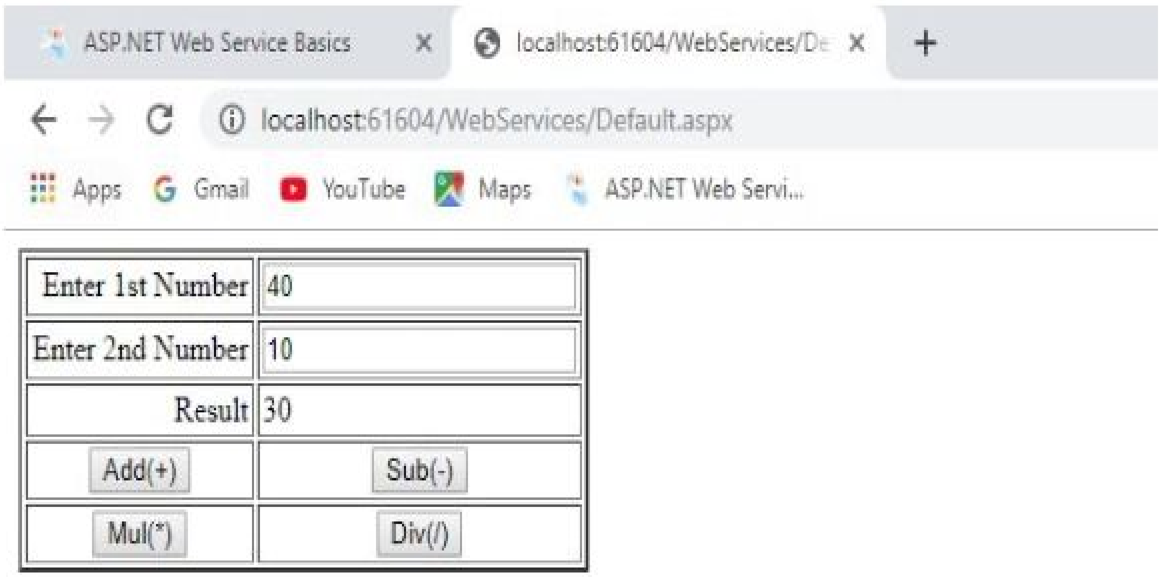
Output :


ASP.NET Web Service Basics
localhost:61604/WebServices/De
localhost:61604/WebServices/Default.aspx
Apps
Gmail
YouTube
Maps
ASP.NET Web Servi...
Add(+)
Sub(-)
Mul(*)
Div(/)

Enter 1st Number	
Enter 2nd Number	
Result	
Add(+)	Sub(-)
Mul(*)	Div(/)


ASP.NET Web Service Basics
localhost:61604/WebServices/De
localhost:61604/WebServices/Default.aspx
Apps
Gmail
YouTube
Maps
ASP.NET Web Servi...
Add(+)
Sub(-)
Mul(*)
Div(/)

Enter 1st Number	40
Enter 2nd Number	10
Result	50


ASP.NET Web Service Basics
localhost:61604/WebServices/De
localhost:61604/WebServices/Default.aspx
Apps
Gmail
YouTube
Maps
ASP.NET Web Servi...
Add(+)
Sub(-)
Mul(*)
Div(/)

Enter 1st Number	40
Enter 2nd Number	10
Result	30

Practical No : 5

Aim: Write a program to execute any one mutual exclusion algorithm.

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.
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.

5(A): Write a program to execute any one mutual exclusion algorithm.

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 is " + str);
   }
}
```

TokenClient1.java

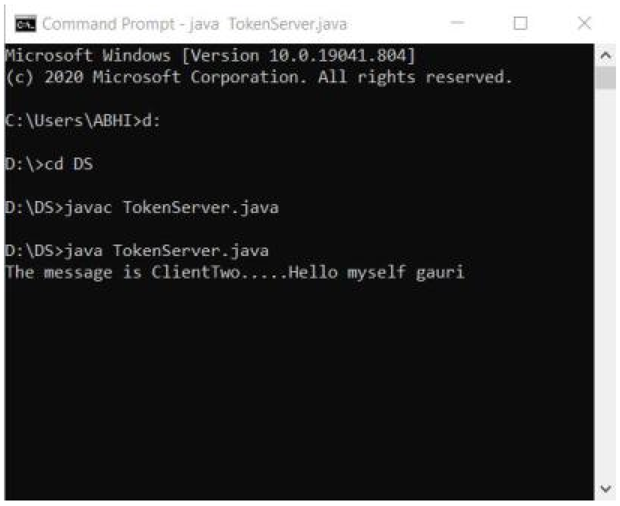
```
import java.io.*;
import java.net.*;
```

```
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;
        }
    }
}
```

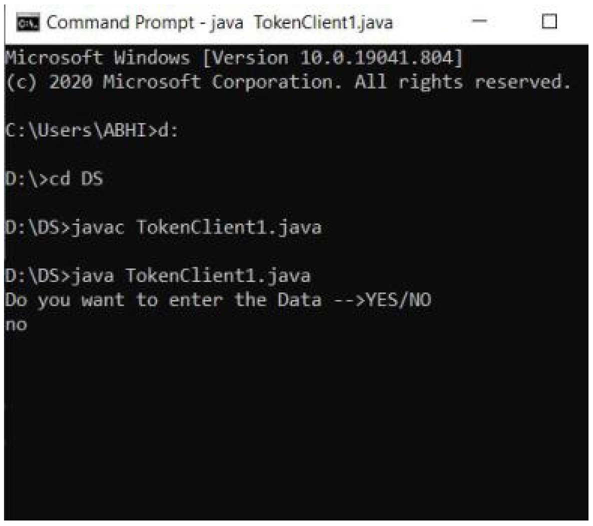
TokenClient2.java

```
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;
        }
    }
}
```

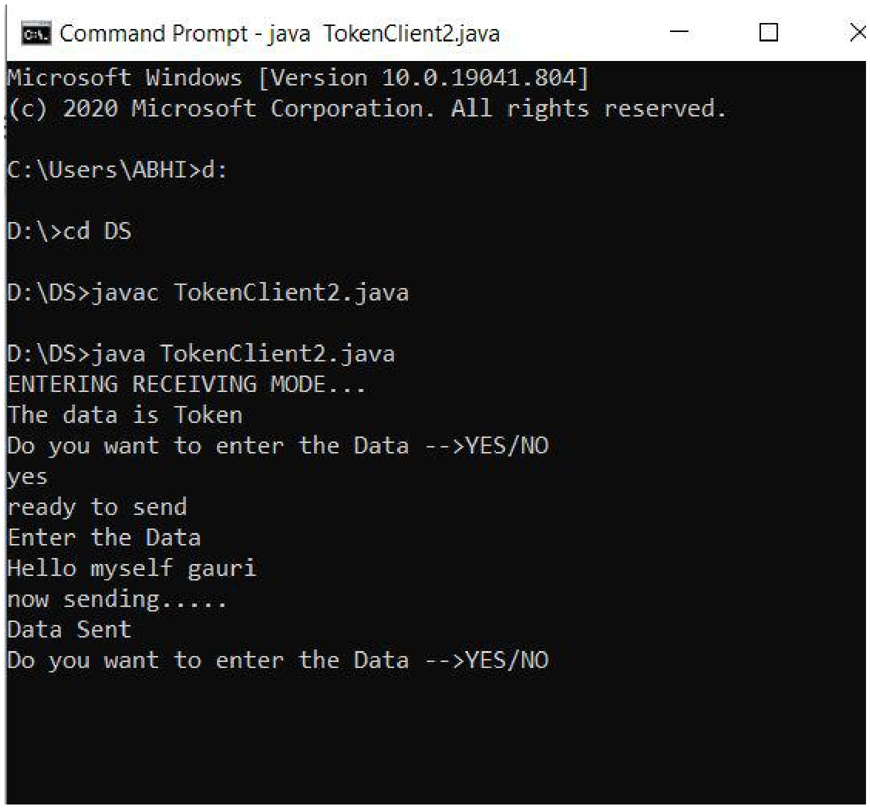
Output :



```
Command Prompt - java  TokenServer.java
Microsoft Windows [Version 10.0.19041.804]
(c) 2020 Microsoft Corporation. All rights reserved.

C:\Users\ABHI>d:

D:\>cd DS

D:\DS>javac TokenServer.java

D:\DS>java TokenServer.java
The message is ClientTwo.....Hello myself gauri
```



```
Command Prompt - java  TokenClient1.java
Microsoft Windows [Version 10.0.19041.804]
(c) 2020 Microsoft Corporation. All rights reserved.

C:\Users\ABHI>d:

D:\>cd DS

D:\DS>javac TokenClient1.java

D:\DS>java TokenClient1.java
Do you want to enter the Data -->YES/NO
no
```



```
Command Prompt - java  TokenClient2.java
Microsoft Windows [Version 10.0.19041.804]
(c) 2020 Microsoft Corporation. All rights reserved.

C:\Users\ABHI>d:

D:\>cd DS

D:\DS>javac TokenClient2.java

D:\DS>java TokenClient2.java
ENTERING RECEIVING MODE...
The data is Token
Do you want to enter the Data -->YES/NO
yes
ready to send
Enter the Data
Hello myself gauri
now sending.....
Data Sent
Do you want to enter the Data -->YES/NO
```

Practical No : 6

1(A): Write a program to implement any one election algorithm.

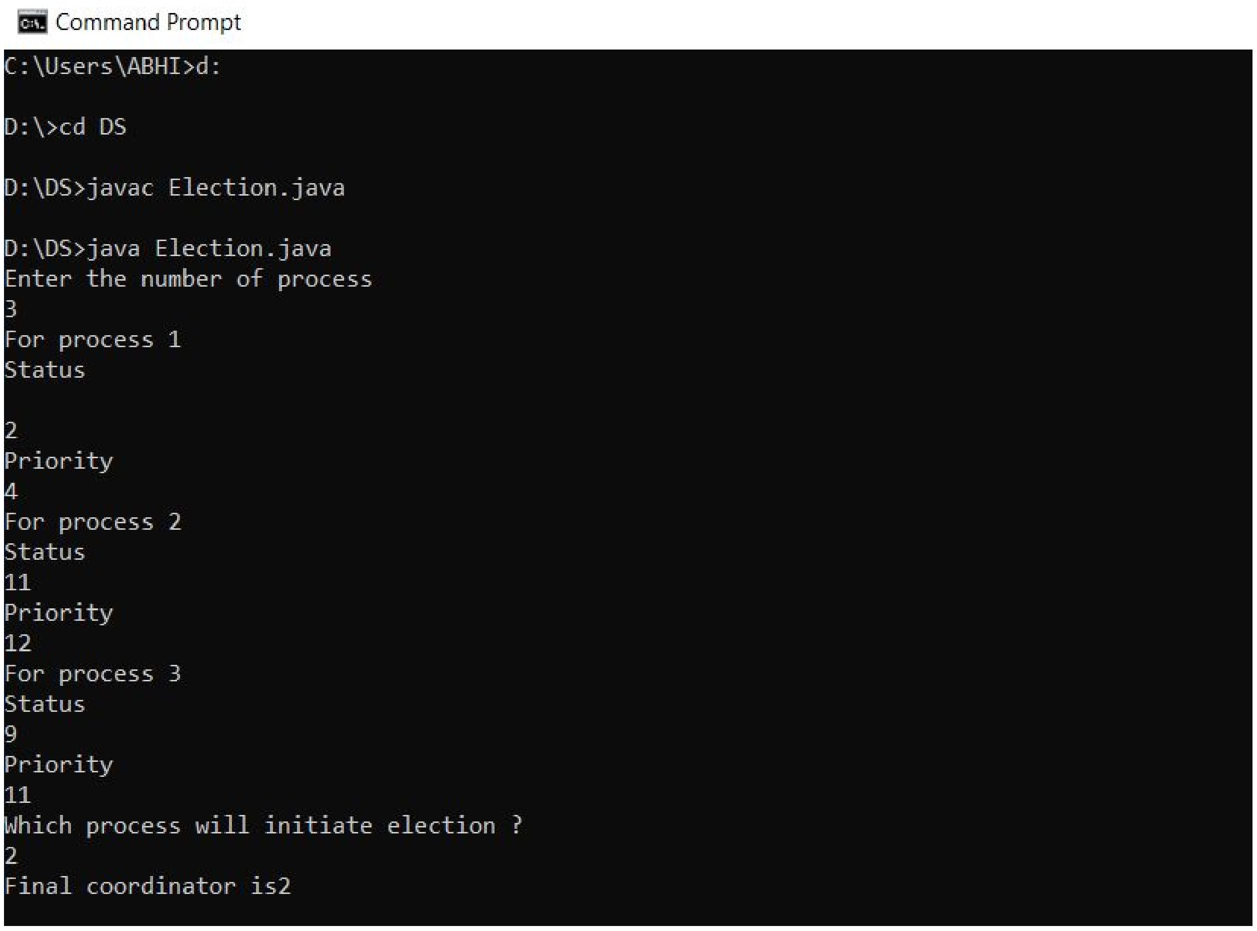
CODE :

Election.java

```
import java.io.*;
import java.util.Scanner;

class Election {
   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 initiate 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\ABHI>d:

D:\>cd DS

D:\DS>javac Election.java

D:\DS>java Election.java
Enter the number of process
3
For process 1
Status

2
Priority
4
For process 2
Status
11
Priority
12
For process 3
Status
9
Priority
11
Which process will initiate election ?
2
Final coordinator is2
```

Practical No : 7

1(A): Show the implementation of any one clock synchronization algorithm.

Code:

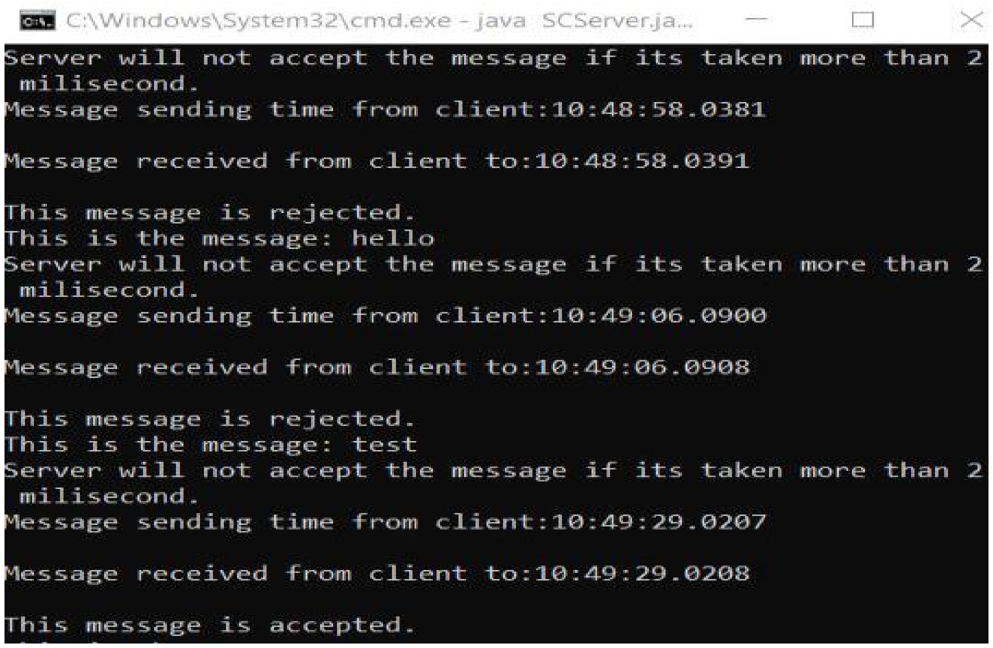
SCServer.java

```
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 at PORT " + PORT);

            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 +
"\n");
                    String strDate = dateFormat.format(timeMilli);
                    System.out.println("Message received from client to:" + strDate + "\n");
                    System.out.println("This message is rejected.");
                } else {
                    System.out.println("Message sending time from client:" + formattedClientTime +
"\n");
                    String strDate = dateFormat.format(timeMilli);
                    System.out.println("Message received from client to:" + strDate + "\n");
                    System.out.println("This message is accepted.");
                }
            }
            inputStream.close();

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

    }
}
```

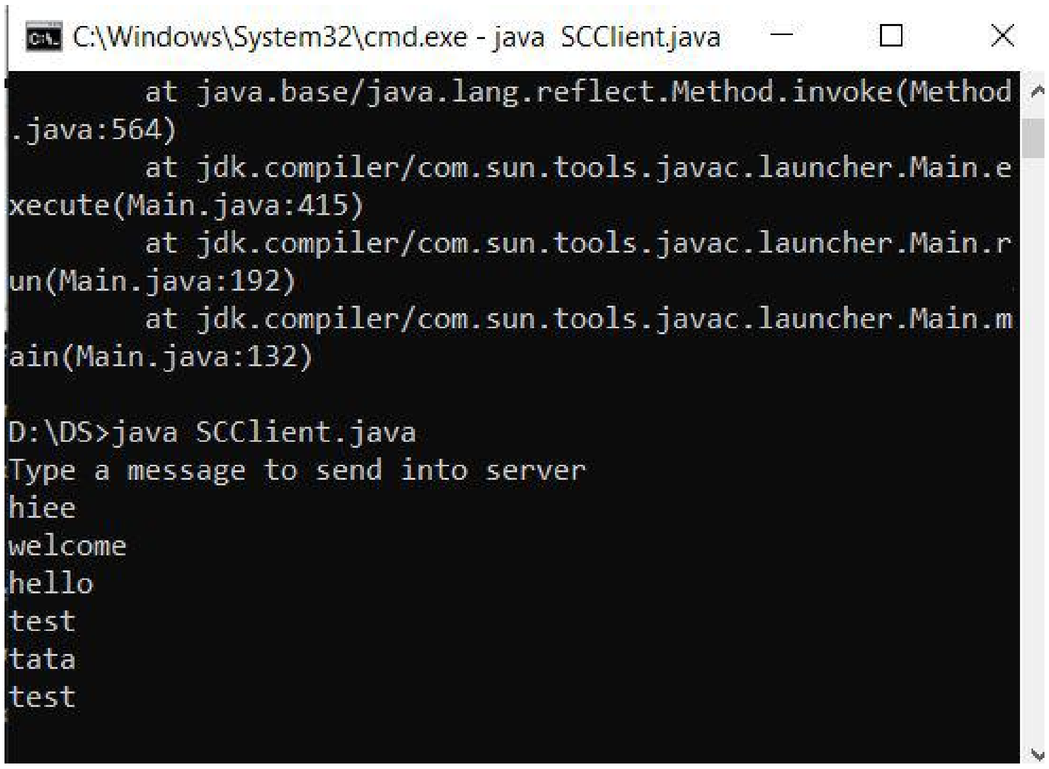
SCClient.java

```
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 :



```
Server will not accept the message if its taken more than 2
 milisecond.
Message sending time from client:10:48:58.0381

Message received from client to:10:48:58.0391

This message is rejected.
This is the message: hello
Server will not accept the message if its taken more than 2
 milisecond.
Message sending time from client:10:49:06.0900

Message received from client to:10:49:06.0908

This message is rejected.
This is the message: test
Server will not accept the message if its taken more than 2
 milisecond.
Message sending time from client:10:49:29.0207

Message received from client to:10:49:29.0208

This message is accepted.
```



```
        at java.base/java.lang.reflect.Method.invoke(Method
.java:564)
        at jdk.compiler/com.sun.tools.javac.launcher.Main.e
xecute(Main.java:415)
        at jdk.compiler/com.sun.tools.javac.launcher.Main.r
un(Main.java:192)
        at jdk.compiler/com.sun.tools.javac.launcher.Main.m
ain(Main.java:132)

D:\DS>java SCClient.java
Type a message to send into server
hiee
welcome
hello
test
tata
test
```

Practical No : 8

1(A): Write a program to implement two phase commit protocol.

Code :

Server.java

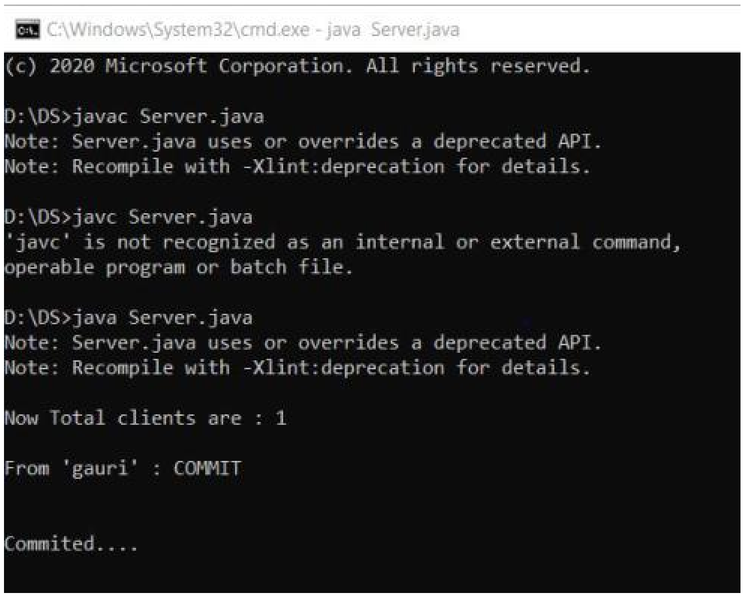
```
import java.io.*;
import java.net.*;
import java.util.*;

public class Server {
   boolean closed = false, inputFromAll = false;
   List<clientThread> t;
   List<String> data;

   Server() {
      t = new ArrayList<clientThread>();
      data = new ArrayList<String>();
   }

   public static void main(String args[]) {
      Socket clientSocket = null;
      ServerSocket serverSocket = null;
      int port_number = 1111;
      Server ser = new Server();
      try {
         serverSocket = new ServerSocket(port_number);
      } catch (IOException e) {
         System.out.println(e);
      }
      while (!ser.closed) {
         try {
            clientSocket = serverSocket.accept();
            clientThread th = new clientThread(ser, clientSocket);
            (ser.t).add(th);
            System.out.println("\nNow Total clients are : " + (ser.t).size());
            (ser.data).add("NOT_SENT");
            th.start();
         } catch (IOException e) {
         }
      }
      try {
         serverSocket.close();
      } catch (Exception e1) {
      }
   }
}

class clientThread extends Thread {
   DataInputStream is = null;
   String line;
   String destClient = "";
   String name;
   PrintStream os = null;
   Socket clientSocket = null;
   String clientIdentity;
   Server ser;

   public clientThread(Server ser, Socket clientSocket) {
      this.clientSocket = clientSocket;
      this.ser = ser;
   }

   public void run() {
      try {

         is = new DataInputStream(clientSocket.getInputStream());
         os = new PrintStream(clientSocket.getOutputStream());
         os.println("Enter your name.");
         name = is.readLine();
         clientIdentity = name;
         os.println("Welcome " + name + " to this 2 Phase Application.\nYou will receive a vote
Request now...");
         os.println("VOTE_REQUEST\nPlease enter COMMIT or ABORT to proceed : ");
         for (int i = 0; i < (ser.t).size(); i++) {
            if ((ser.t).get(i) != this) {
                ((ser.t).get(i)).os.println("---A new user " + name + " entered the Appilcation---");
            }
         }
         while (true) {
            line = is.readLine();
            if (line.equalsIgnoreCase("ABORT")) {
                System.out.println("\nFrom '" + clientIdentity
                        + "' : ABORT\n\nSince aborted we will not wait for inputs from other clients.");
                System.out.println("\nAborted....");
                for (int i = 0; i < (ser.t).size(); i++) {
                    ((ser.t).get(i)).os.println("GLOBAL_ABORT");
                    ((ser.t).get(i)).os.close();
                    ((ser.t).get(i)).is.close();
                }
                break;
            }
            if (line.equalsIgnoreCase("COMMIT")) {
                System.out.println("\nFrom '" + clientIdentity + "' : COMMIT");
                if ((ser.t).contains(this)) {
                    (ser.data).set((ser.t).indexOf(this), "COMMIT");
                    for (int j = 0; j < (ser.data).size(); j++) {
                        if (!(((ser.data).get(j)).equalsIgnoreCase("NOT_SENT"))) {
                            ser.inputFromAll = true;
                            continue;
                        }

                        else {
                            ser.inputFromAll = false;
                            System.out.println("\nWaiting for inputs from other clients.");
                            break;
                        }
                    }
                    if (ser.inputFromAll) {
                        System.out.println("\n\nCommited....");
                        for (int i = 0; i < (ser.t).size(); i++) {
                            ((ser.t).get(i)).os.println("GLOBAL_COMMIT");
                            ((ser.t).get(i)).os.close();
                            ((ser.t).get(i)).is.close();
                        }
                        break;
                    }
                }
            }
         }
         ser.closed = true;
         clientSocket.close();
      } catch (IOException e) {
      }
      ;
   }
}

}
```

Client.java

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

public class Client implements Runnable {
    static Socket clientSocket = null;
    static PrintStream os = null;
    static DataInputStream is = null;
    static BufferedReader inputLine = null;
    static boolean closed = false;

    public static void main(String[] args) {
        int port_number = 1111;
        String host = "localhost";
        try {
            clientSocket = new Socket(host, port_number);
            inputLine = new BufferedReader(new InputStreamReader(System.in));
            os = new PrintStream(clientSocket.getOutputStream());
            is = new DataInputStream(clientSocket.getInputStream());
        } catch (Exception e) {
            System.out.println("Exception occurred : " + e.getMessage());
        }
        if (clientSocket != null && os != null && is != null) {
            try {
                new Thread(new Client()).start();
                while (!closed) {
                    os.println(inputLine.readLine());
                }
                os.close();
                is.close();
                clientSocket.close();
            } catch (IOException e) {
                System.err.println("IOException: " + e);
            }
        }
    }

    public void run() {
        String responseLine;
        try {

            while ((responseLine = is.readLine()) != null) {
                System.out.println("\n" + responseLine);
                if (responseLine.equalsIgnoreCase("GLOBAL_COMMIT") == true
                        || responseLine.equalsIgnoreCase("GLOBAL_ABORT") == true) {
                    break;
                }
            }
            closed = true;
        } catch (IOException e) {
            System.err.println("IOException: " + e);
        }
    }
}
```

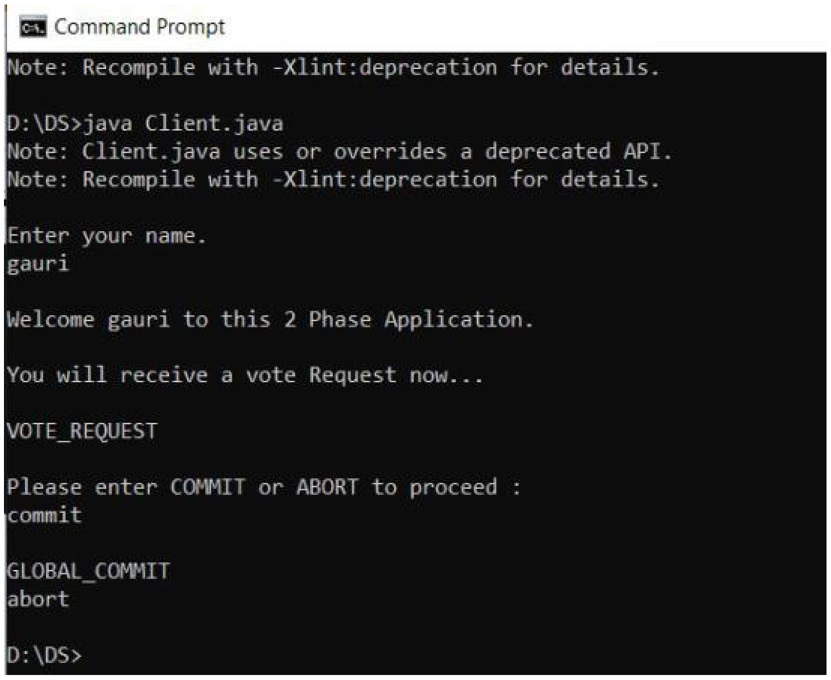
Output :

Server.java



```
C:\Windows\System32\cmd.exe - java  Server.java
(c) 2020 Microsoft Corporation. All rights reserved.

D:\DS>javac Server.java
Note: Server.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.

D:\DS>javc Server.java
'javc' is not recognized as an internal or external command,
operable program or batch file.

D:\DS>java Server.java
Note: Server.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.

Now Total clients are : 1

From 'gauri' : COMMIT


Commited....
```

Client.java



```
Command Prompt
Note: Recompile with -Xlint:deprecation for details.

D:\DS>java Client.java
Note: Client.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.

Enter your name.
gauri

Welcome gauri to this 2 Phase Application.

You will receive a vote Request now...

VOTE_REQUEST

Please enter COMMIT or ABORT to proceed :
commit

GLOBAL_COMMIT
abort

D:\DS>
```