Ex. No: 1 **Retrieving data using URL**

```
import java.net.*;
import java.util.*;
import java.io.*;
class url
{
        public static void main(String ar[]) throws Exception
        {
        URL u=new URL("file:///D:/Network/a.html");
        URLConnection con=u.openConnection();
        System.out.println("The File is       :"+u.getFile());
        System.out.println("The Path of the URL:"+u.getPath());
        System.out.println("The Port Number is :"+u.getPort());
        System.out.println("The Protocol used  :"+u.getProtocol());
        System.out.println("The Date is       :"+new Date(con.getDate()));
        System.out.println("The Content type is:"+con.getContentType());
        System.out.println("The last modified  :"
                                +new Date(con.getLastModified()));
        long d=con.getExpiration();
        System.out.println("The Expires Date is       :"+new Date(d));
        System.out.println("The length  is     :"+con.getContentLength());
        System.out.println("The Content in file is ");
        System.out.println("======================");
        int ch;
        InputStream in=con.getInputStream();
        while((ch=in.read())!=-1)
                System.out.print((char)ch);
        in.close();
        }
}
```

Output:

```
D:\Network>javac url.java

D:\Network>java url

The File is        :/D:/Network/a.html
The Path of the URL:/D:/Network/a.html
The Port Number is :-1
The Protocol used  :file
The Date is        :Wed Dec 31 16:00:00 PST 1969
The Content type is:text/html
The last modified  :Thu Feb 10 03:11:46 PST 2011
The Expires Date is        :Wed Dec 31 16:00:00 PST 1969
The length  is     :1720
The Content in file is
======================
<html>
<head>
<title> My HomePage </title>
</head>
<body bgcolor="Pink">
<marquee WIDTH=100% BEHAVIOR=ALTERNATE BGColor=yellow><h1>
HTML BASICS </h1></marquee>
</body>
</html>
```

Ex. No.: 2a Implementation of the Socket Programming Using TCP/IP

//------tcpserver1.java--------

```
import java.io.*;
import java.net.*;
public class tcpserver1
{
public static void main(String a[]) throws Exception
{
System.out.println("TCP SERVER");
System.out.println("Server is ready to connect…");
ServerSocket serversoc=new ServerSocket(9);
Socket clientsoc = serversoc.accept();
PrintWriter out = new PrintWriter(clientsoc.getOutputStream(), true);
BufferedReader in = new BufferedReader(new
InputStreamReader(clientsoc.getInputStream()));
String inputline;
BufferedReader stdin = new
                        BufferedReader(new InputStreamReader(System.in));
try
{
while (true)
{
inputline = stdin.readLine();
out.println(inputline);
System.out.println("Client Says : "+in.readLine());
}
}
```

```
catch(Exception e)
{
System.exit(0);
}
}
}
```

//-----tcpclient1.java-----

```
import java.io.*;
import java.net.*;
public class tcpclient1
{
public static void main(String[] args) throws IOException
{
System.out.println("TCP CLIENT");
System.out.println("Enter the host name to connect");
DataInputStream inp=new DataInputStream(System.in);
String str=inp.readLine();
Socket clientsoc = new Socket(str, 9);
PrintWriter out = new PrintWriter(clientsoc.getOutputStream(), true);
BufferedReader in = new BufferedReader(new
InputStreamReader(clientsoc.getInputStream()));
BufferedReader stdin = new
                                BufferedReader(new InputStreamReader(System.in));
String userinput;
try
{
while (true)
{
System.out.println("Sever Says : " + in.readLine());
```

```
userinput = stdin.readLine();
out.println(userinput);
}
}
catch(Exception e)
{
System.exit(0);
}
}
}
```

Output:

```
D:\Network>javac tcpserver1.java
D:\Network>java tcpserver1
TCP SERVER
Server is ready to connectà
hi
Client Says : hello server
ya client

D:\Network>javac tcpclient1.java
Note: tcpclient1.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
D:\Network>java tcpclient1
TCP CLIENT
Enter the host name to connect
mecselab-20.
Sever Says : hi
hello server
Sever Says : ya client
```

Ex. No.: 2b Implementation of Socket Programming

Using UDP

//------udps.java-------

```
import java.io.*;
import java.net.*;
import java.lang.*;
public class udps
{
  public static DatagramSocket ds;
  public static int clientport=789,serverport=790;
  public static void main(String args[])throws Exception
  {
   byte buffer[]=new byte[1024];
   ds=new DatagramSocket(serverport);
   String sl=new String();
   BufferedReader br=new
                              BufferedReader(new InputStreamReader(System.in));
   System.out.println("client is accepted");
   System.out.println("Enter end to enter into dos prompt");
   InetAddress in=InetAddress.getByName("localhost");
   System.out.println(in);
   while (true)
   {
        sl=br.readLine();
        if(sl==null||sl.equals("end"))
          break;
        buffer=sl.getBytes();
        ds.send(new DatagramPacket(buffer,sl.length(),in,clientport));
    }
  }
}
```

```
//------udpc.java-------
import java.io.*;
import java.net.*;
public class udpc
{
   public static DatagramSocket ds;
   public static byte buffer[]=new byte[1024];
   public static int clientport=789,serverport=790;
   public static void main(String args[])throws Exception
   {
     ds=new DatagramSocket(clientport);
     System.out.println("Client is waiting for the message from server");
     System.out.println("Enter ctrl+c to enter into the Dos Prompt");
     while(true)
     {
          DatagramPacket p=new DatagramPacket(buffer,buffer.length);
          ds.receive(p);
          String st=new String(p.getData(),0,p.getLength());
          if (st.equals("end"))
             break;
          System.out.println(st);
      }
    }
 }
```

Output:

D:\Network>javac udps.java

D:\Network>java udps

client is accepted

Enter end to enter into dos prompt

localhost/127.0.0.1

Hello client

How are you?

D:\Network>javac udpc.java

D:\Network>java udpc

Client is waiting for the message from server

Enter ctrl+c to enter into the Dos Prompt

Hello client

How are you?

Ex. No.: 3 Implementation of FTP

```
//------ftps1.java-----
import java.net.*;
import java.io.*;
class ftps1
{
public static void main(String args[])throws Exception
{
  String msg,str;
try
{
System.out.println("FILES IN SERVER");
System.out.println("1. url.java");
System.out.println("2. tcpserver1.java");
System.out.println("3. tcpclient1.java");
System.out.println("4. udps.java");
ServerSocket serversock=new ServerSocket(1092);
Socket clientsock=serversock.accept();
System.out.println("Connection Established");
DataInputStream dis=new DataInputStream(clientsock.getInputStream());
str=dis.readLine();
System.out.println("Client requests to download the file : "+str);
File f=new File(str);
FileInputStream fis=new FileInputStream(f);
DataInputStream disf=new DataInputStream(fis);
DataOutputStream dos=new DataOutputStream(clientsock.getOutputStream());
if(f.exists())
{
```

```
while((msg=disf.readLine())!=null)
{
    dos.writeChars(msg);
 }
 System.out.println("File Transfer Completed");
 }
 serversock.close();
 clientsock.close();
}
catch(Exception e)
{
  System.out.println("Unable to receive file name from client: "+e);
}
 }
}
```

//------ftpc1.java---------

```
import java.net.*;
import java.io.*;
class ftpc1
{
public static void main(String args[])throws Exception
{
Socket echosocket=null;
PrintStream ps;
String UserInput,f1;
int i=0;
try
{
```

```
echosocket = new Socket(InetAddress.getLocalHost(),1092);
ps=new PrintStream(echosocket.getOutputStream(),true);
DataInputStream dis=new DataInputStream(echosocket.getInputStream());
DataInputStream din=new DataInputStream(System.in);
System.out.println();
System.out.print("Enter name of the file to be downloaded : ");
String s=din.readLine();
System.out.println();
System.out.print("Enter  the file name to save in  : ");
f1=din.readLine();
ps.println(s.toString());
FileOutputStream fos=new FileOutputStream(f1);
DataOutputStream dos=new DataOutputStream(fos);
System.out.println(" Download in progress . . . ");
System.out.println("     FILE CONTENT ");
while((UserInput=dis.readLine())!=null)
{
dos.writeChars(UserInput);
System.out.println(UserInput);
}
System.out.println("Download completed.");
echosocket.close();
}
catch(Exception e)
{
System.out.println("Exception : "+e);
}
}
}
```

Output:

```
D:\Network>javac ftps1.java
Note: ftps1.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
D:\Network>java ftps1
FILES IN SERVER
1. url.java
2. tcpserver1.java
3. tcpclient1.java
4. udps.java
Connection Established
Client requests to download the file : url.java
File Transfer Completed

D:\Network>javac ftpc1.java
Note: ftpc1.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.

D:\Network>java ftpc1

Enter name of the file to be downloaded : udps.java

Enter  the file name to save in  : 2.java
 Download in progress . . .
    FILE CONTENT
import java.io.*;
import java.net.*;
import java.lang.*;
public class udps
Download completed.
```

Ex. No.: 4a **Implementation of ECHO Command**

//-----echos.java------

```
import java.io.*;
import java.net.*;
class echos
{
	public static void main(String a[])throws IOException
	{
		try
		{
			ServerSocket ss=new ServerSocket(1100);
			Socket s=ss.accept();
			PrintStream ps=new PrintStream(s.getOutputStream());
			BufferedReader br=new
				BufferedReader(new InputStreamReader(System.in));
			String str,data;
			do
			{
				System.out.println("SERVER TO LOCALHOST");
				str=br.readLine();
				ps.println(str);
			}
			while(!(str.equalsIgnoreCase("end")));
		}
		catch(IOException e)
		{
			System.out.println("error"+e);
		}
	}
}
```

```
//------echoc.java-----
import java.io.*;
import java.net.*;
class echoc
{
	public static void main(String a[])throws IOException
	{
		try
		{
			Socket ss=new Socket(InetAddress.getLocalHost(),1100);
			BufferedReader br= new
		BufferedReader(new InputStreamReader(ss.getInputStream()));
			String str,data;
			do
			{
				str=br.readLine();
				System.out.println("ECHO FROM SERVER::"+str);
			}
			while(!(str.equalsIgnoreCase("end")));
		}
		catch(IOException e)
		{
			System.out.println("error"+e);
		}
	}

}
```

Output:

D:\Network>javac echos.java

D:\Network>java echos

SERVER TO LOCALHOST

hello client

SERVER TO LOCALHOST

Ya are you busy??????

SERVER TO LOCALHOST

D:\Network>javac echoc.java

D:\Network>java echoc

ECHO FROM SERVER::hello client

ECHO FROM SERVER::Ya are you busy??????

Ex. No: 4b **Implementation of Ping**

//-----pings .java-------

```
import java.io.*;
import java.net.*;
class pings
{
        public static void  main(String args[])
       {
         BufferedReader in;
         try
         {
                Runtime r=Runtime.getRuntime();
                Process p=r.exec("Ping 10.0.6.40");
                        if(p==null)
                System.out.println("Could not connect");
                in=new BufferedReader
                        (new InputStreamReader(p.getInputStream()));
                String line;
                while((line=in.readLine())!=null)
                {
                        if(line.startsWith("reply"))
                        System.out.println("this is reply");
                        else if(line.startsWith("request"))
                        System.out.println("there is no reply");
                        else if(line.startsWith("destinator"))
                        System.out.println("destination host unreachable");
                        else
                        System.out.println(line);
                }
                System.out.println(in.readLine());
```

```
                in.close();
        }
        catch(IOException e)
        {
              System.out.println(e.toString());
         }
      }
}
```

Output:

```
D:\Network>javac pings.java
D:\Network>java pings
Pinging 10.0.6.40 with 32 bytes of data:
Reply from 10.0.6.40: bytes=32 time<1ms TTL=128

Reply from 10.0.6.40: bytes=32 time<1ms TTL=128

Reply from 10.0.6.40: bytes=32 time<1ms TTL=128

Reply from 10.0.6.40: bytes=32 time<1ms TTL=128

Ping statistics for 10.0.6.40:

    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),

Approximate round trip times in milli-seconds:

    Minimum = 0ms, Maximum = 0ms, Average = 0ms

null
```

Ex.No: 4c Implementation of talk

//------talkc.java--------

```
import java.io.*;
import java.net.*;
class talkc
{
public static void main(String args[])throws IOException
 {
   try
    {
          int client=3000,server=3002;
          DatagramSocket ds=new DatagramSocket(client);
          DatagramPacket dp;
          byte outbuff[]=new byte[1024];
          byte inbuff[]=new byte[1024];
          String str,data;
          DataInputStream dis=new DataInputStream(System.in);
          do
          {
          System.out.println("To Server:");
          str=dis.readLine();
          outbuff=str.getBytes();
          InetAddress ia=InetAddress.getLocalHost();
          dp=new DatagramPacket(outbuff,str.length(),ia,server);
          ds.send(dp);
          System.out.println("From Server:");
          dp=new DatagramPacket(inbuff,inbuff.length);
          ds.receive(dp);
          data=new String(dp.getData(),0,dp.getLength());
          System.out.println(data);
```

```
        }while(!(str.equalsIgnoreCase("end")  || data.equalsIgnoreCase("end") ));
    }
   catch(IOException e)
    {
      System.out.println("Error"+e);
    }
 }
}
```

//------talks.java-----

```
import java.io.*;
import java.net.*;
class talks
{
public static void main(String args[])throws IOException
 {
   try
    {
          int client=3000,server=3002;
          DatagramSocket ds=new DatagramSocket(server);
          DatagramPacket dp;
          String str,data;
          byte inbuff[]=new byte[1024];
          byte outbuff[]=new byte[1024];
          do
          {
          dp=new DatagramPacket(inbuff,inbuff.length);
          ds.receive(dp);
          str=new String(dp.getData(),0,dp.getLength());
          System.out.println("From Client:"+str);
          System.out.println("To Client:");
```

```
        DataInputStream dis=new DataInputStream(System.in);
        data=dis.readLine();
        outbuff=data.getBytes();
        InetAddress ia=dp.getAddress();
        int port=dp.getPort();
        dp=new DatagramPacket(outbuff,data.length(),ia,port);
        ds.send(dp);
        }while(!(str.equalsIgnoreCase("end") || data.equalsIgnoreCase("end")  ));
   }
  catch(IOException e)
   {
     System.out.println("Error"+e);
   }
 }
}
```

Output:

```
D:\NETWORK LAB>javac talkc.java
Note: talkc.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.

D:\NETWORK LAB>java talkc
To Server:
Hi
From Server:
Hello
To Server:
Hello
From Server:
People in India!!!!
```

To Server:
People in India!!!!

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

D:\NETWORK LAB>java talks
From Client:Hi
To Client:
Hello
From Client:Hello
To Client:
People in India!!!!
From Client:People in India!!!!

Ex.No. 5 Implementation of Remote Command Execution

//-------remotec.java---------

```
import java.io.*;
import java.net.*;
public class remotec
{
        public static void main(String a[])throws Exception
        {
                Socket s=new Socket("localhost",22222);
                DataOutputStream dos=new
                                        DataOutputStream(s.getOutputStream());
                DataInputStream dis=new DataInputStream(System.in);
                String str;
                System.out.println("Enter the command to be executed");
                str=dis.readLine();
                dos.writeBytes(str+"\n");
                System.out.println(str+" has been executed on the server");
                s.close();
        }
}
```

//---------remotes.java--------

```
import java.io.*;
import java.net.*;
import java.lang.*;
public class remotes
{
        public static void main(String a[])throws Exception
        {
                ServerSocket ss=new ServerSocket(22222);
```

```
            Socket s=ss.accept();
            DataInputStream dis=new DataInputStream(s.getInputStream());
            String str=dis.readLine();
            Runtime r=Runtime.getRuntime();
        Process p=r.exec(str);
            System.out.println(str+" has been executed successfully");
    }
}
```

Output:

```
D:\NETWORK LAB>javac remotes.java
Note: remotes.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.

D:\NETWORK LAB>java remotes
mspaint has been executed successfully

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

D:\NETWORK LAB>java remotec
Enter the command to be executed
mspaint
mspaint has been executed on the server
```

Ex.No.: 6 Implementation of ARP

```
//------arprot.java----------
import java.io.*;
import java.util.*;
public class arprot
{
Hashtable h=new Hashtable();
StringTokenizer st;
String x;
int cnt=0;
public void getARP() throws IOException
{
Process p=Runtime.getRuntime().exec("arp -a");
BufferedReader br=new
                BufferedReader(new InputStreamReader(p.getInputStream()));
while( (x=br.readLine()) != null )
{
System.out.println(x);
cnt++;
if(cnt>3)
{
st=new StringTokenizer(x," ");
if(st.hasMoreTokens())
h.put(st.nextToken(),st.nextToken());
}
}
}
public void display()
{
System.out.println("no of items in hash is:"+h.size());
```

```
Enumeration er=h.keys();
while (er.hasMoreElements())
System.out.println(er.nextElement());
}
public void search(String str)
{
if (h.containsKey(str))
{
System.out.println("IP address:"+str);
System.out.println("MAC address:"+h.get(str));
}
else
System.out.println("No entry in ARP cache");
}
public static void main(String as[])throws IOException
{
arprot ar=new arprot();
ar.getARP();
ar.display();
BufferedReader da=new BufferedReader(new InputStreamReader(System.in));
System.out.println("Enter IP address:\n ----------");
String data = da.readLine();
ar.search(data);
}
}
```

Output:

```
D:\NETWORK LAB>javac arprot.java
Note: arprot.java uses unchecked or unsafe operations.
Note: Recompile with -Xlint:unchecked for details.
```

```
D:\NETWORK LAB>java arprot

Interface: 10.0.6.40 --- 0x2
  Internet Address      Physical Address      Type
  10.0.6.1              00-23-ea-cb-72-c6     dynamic
  10.0.6.23             00-16-e6-93-88-06     dynamic
  10.0.6.32             00-16-e6-94-a4-94     dynamic
  10.0.6.36             00-16-e6-94-a6-db     dynamic
no of items in hash is:4
10.0.6.32
10.0.6.1
10.0.6.36
10.0.6.23
Enter IP address:
 -------------------------
10.0.6.36
IP address:10.0.6.36
MAC address:00-16-e6-94-a6-db
```

Ex. No.: 7 **IMPLEMENTATION OF RARP**

```
// RARP.java
import java.io.*;
import java.util.*;
public class RARP
{
        Hashtable h=new Hashtable();
        StringTokenizer st;
        String x;
        int cnt=0;
        public void getRARP() throws IOException
        {
            Process p=Runtime.getRuntime().exec("arp -a");
            BufferedReader br=new
            BufferedReader(new InputStreamReader(p.getInputStream()));
            while( (x=br.readLine()) != null )
            {
                cnt++;
                if(cnt>3)
                {
                st=new StringTokenizer(x," ");
                if(st.hasMoreTokens())
                        h.put(st.nextToken(),st.nextToken());
                }
            }}
        public void display()
        {
            System.out.println("no of items in hash is:"+h.size());
            Enumeration er=h.keys();
            while (er.hasMoreElements())
```

```
            System.out.println(er.nextElement());
        }
        public void search(String str)
        {
            Enumeration er=h.keys();
            System.out.println("no of items in hash is:"+h.size());
            if (h.containsValue(str))
            {
                System.out.println("MAC address:"+str);
                System.out.println("IP address:"+er.nextElement());
            }
            else
                System.out.println("No entry in RARP cache");
        }
        public static void main(String as[])throws IOException
        {
            RARP r=new RARP();
            r.getRARP();
            r.display();
            BufferedReader da=new BufferedReader(new
InputStreamReader(System.in));
            System.out.println("Enter MAC address:\n ------------------------");
            String data = da.readLine();
            r.search(data);
        }

}
```

Output:

```
D:\NETWORK LAB>javac rarp.java
Note: rarp.java uses unchecked or unsafe operations.
Note: Recompile with -Xlint:unchecked for details.

D:\NETWORK LAB>java RARP
no of items in hash is:3
10.0.6.32
10.0.6.1
10.0.6.50
Enter MAC address:
 -------------------------
00-16-e6-94-a4-94
no of items in hash is:3
MAC address:00-16-e6-94-a4-94
IP address:10.0.6.32
```

Ex. No.: 8 **IMPLEMENTATION OF RMI / RPC**

// RMI_Client.java

```
import java.io.*;
import java.rmi.*;
import java.rmi.Naming.*;
public class RMI_Client
{
        public static void main(String args[])throws IOException
        {
          int val1,val2;
          try
          {
            BufferedReader in = new
                BufferedReader(new InputStreamReader(System.in));
           RMI_Intf ref=(RMI_Intf)Naming.lookup("impl");
           boolean flag=false;
           do
           {
                System.out.println("1. ADDITION");
                System.out.println("2. SUBTRACTION");
                System.out.println("3. MULTIPLICATION");
                System.out.println("4. DIVISION");
                System.out.println("5. QUIT");
                System.out.println("Enter your Choice [1 / 2 / 3 / 4 / 5]: ");
                int ch=Integer.parseInt(in.readLine());
                 String s;
                switch(ch)
                {
```

```
        case 1:
                System.out.println("Enter Values : ");
                 val1=Integer.parseInt(in.readLine());
                val2=Integer.parseInt(in.readLine());
                  System.out.println("Sum : "+ref.add(val1,val2));
        break;
        case 2:
                System.out.println("Enter Values : ");
                val1=Integer.parseInt(in.readLine());
                val2=Integer.parseInt(in.readLine());
                System.out.println("Difference : "+ref.sub(val1,val2));
        break;
        case 3:
                System.out.println("Enter Values : ");
                val1=Integer.parseInt(in.readLine());
                val2=Integer.parseInt(in.readLine());
                System.out.println("Product : "+ref.mul(val1,val2));
        break;

        case 4:
                System.out.println("Enter Values : ");
                val1=Integer.parseInt(in.readLine());
                val2=Integer.parseInt(in.readLine());
                System.out.println("Quotient : "+ref.div(val1,val2));
        break;
        case 5:
                flag=true;
        break;
        }
   }while(!flag);
}
```

```
        catch(Exception e)
        {
            System.out.println("Exception : "+e);
        }
    }
}
```

// RMI_Impl.java

```
import java.rmi.*;
import java.rmi.server.UnicastRemoteObject;
public class RMI_Impl extends UnicastRemoteObject implements RMI_Intf
{       public RMI_Impl()throws RemoteException
        {
                        super();
        }
        public int add(int a, int b)throws RemoteException
        {
                return(a+b);
        }
        public int sub(int a, int b)throws RemoteException
        {
                return(a-b);
        }
        public int mul(int a, int b)throws RemoteException
        {
                return(a*b);
        }
        public int div(int a, int b)throws RemoteException
        {
                return(a/b);
        }}
```

// RMI_Intf.java

```
import java.rmi.*;
public interface RMI_Intf extends Remote
{
  public int add(int a,int b) throws RemoteException;
  public int sub(int a,int b)throws RemoteException;
  public int mul(int a,int b)throws RemoteException;
  public int div(int a, int b)throws RemoteException;
}
```

// RMI_Server.java

```
import java.net.*;
import java.rmi.Naming;
public class RMI_Server
{
        public static void main(String args[])
        {
                try
                {
                   RMI_Impl impl=new RMI_Impl();
                   Naming.rebind("impl",impl);
                }
                catch(Exception e)
                {
                   System.out.println(“Exception : “+e);
                }
        }
}
```

Output:

```
D:\NETWORK LAB>javac RMI_Intf.java
D:\NETWORK LAB>javac RMI_Impl.java
D:\NETWORK LAB>javac RMI_server.java
D:\NETWORK LAB>start rmiregistry
D:\NETWORK LAB>java RMI_Server

D:\NETWORK LAB>javac RMI_Client.java
D:\NETWORK LAB>java RMI_Client
1. ADDITION
2. SUBTRACTION
3. MULTIPLICATION
4. DIVISION
5. QUIT
Enter your Choice [1 / 2 / 3 / 4 / 5]:
1
Enter Values :
259
260
Sum : 519
1. ADDITION
2. SUBTRACTION
3. MULTIPLICATION
4. DIVISION
5. QUIT
Enter your Choice [1 / 2 / 3 / 4 / 5]:
2
Enter Values :
4
3
Difference : 1
```

1. ADDITION
2. SUBTRACTION
3. MULTIPLICATION
4. DIVISION
5. QUIT
Enter your Choice [1 / 2 / 3 / 4 / 5]:
3
Enter Values :
7
9
Product : 63
1. ADDITION
2. SUBTRACTION
3. MULTIPLICATION
4. DIVISION
5. QUIT
Enter your Choice [1 / 2 / 3 / 4 / 5]:
4
Enter Values :
9
3
Quotient : 3
1. ADDITION
2. SUBTRACTION
3. MULTIPLICATION
4. DIVISION
5. QUIT
Enter your Choice [1 / 2 / 3 / 4 / 5]: 4

EX.NO.: 9 Implementation of Shortest path Routing Algorithm

//------CODE-------

```
import java.io.*;
class shortestpath
{
int n,infy,s,t,i,n1;
int weight[][]=new int[10][10];
int pd,k,dc;
int prem[]=new int[10];
int distance[]=new int[10];
int current,smalldist,newdist;
int j,n3;
shortestpath()
{
infy=1000;
}
void input()
{
try
{
BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
System.out.println("enter the no:of nodes:");
String n1=br.readLine();
n=Integer.parseInt(n1);
System.out.println("Enter the weight");
for(i=0;i<n;i++)
{
for(j=0;j<n;j++)
{
```

```
String n2=br.readLine();
n3=Integer.parseInt(n2);
weight[i][j]=n3;
}
}
System.out.println("Enter the source:");
String s1=br.readLine();
s=Integer.parseInt(s1);
System.out.println("Enter the destination:");
String t1=br.readLine();
t=Integer.parseInt(t1);
}
catch(IOException e)
{
System.out.println("Error:"+e);
}
spath(s,t);
}
void spath(int s,int t)
{
for(i=0;i<n;i++)
{
prem[i]=0;
distance[i]=infy;
}
distance[s]=0;
prem[s]=1;
current=s;
while(current!=t)
{
smalldist=infy;
```

```
dc=distance[current];
for(i=0;i<n;i++)
{
if(prem[i]==0)
{
newdist=dc+weight[current][i];
if(newdist< distance[i])
{
distance[i]=newdist;
}
if(distance[i]<smalldist)
{
smalldist=distance[i];
k=i;
}
}
}
current=k;
prem[current]=1;
}
pd=distance[t];
System.out.println("The shortest pathn is:"+pd);
}
public static void main(String a[])throws IOException
{
shortestpath sp=new shortestpath();
sp.input();
}
}
```

//-------OUTPUT----------

```
D:\networks>java shortestpath
enter the no:of nodes:
5
Enter the weight
1000
10
20
30
200
1000
1000
1000
50
1000
1000
1000
1000
1000
1000
1000
1000
1000
1000
60
1000
1000
70
1000
1000
Enter the source:
```

0

Enter the destination:

4

The shortest pathn is:90

EX.NO.: 10 Implementation of Sliding window protocol

//-----CODE-----

```
import java.io.*;
import java.lang.*;
class sliding
{
public static void main(String arg[])throws IOException
{
int a[]=new int[10];
int i,n,k,z;
double m;
System.out.println("Simulation of slidingwindow protocol");
System.out.println("Enter the range of sliding window");
BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
n=Integer.parseInt(br.readLine());
m=Math.pow(2,n);
for(i=0;i<m;i++)
a[i]=i;
i=0;
System.out.println("SENDER \t\t RECIEVER \n");
while(i<m)
{
z=i+1;
for(k=0;k<m;k++)
{
if(k==z)
{
System.out.println("[");
}
```

```
if(k==(z+n))
{
System.out.println("]");
}
System.out.println(a[k]);
}
if(i<m-1)
System.out.println("\n Ack for"+a[i+1]+"\n");
i++;
}
System.out.println("\n\n There are no frames to be sent");
}
}
```

//----------OUTPUT----------

```
D:\networks>javac sliding.java

D:\networks>java sliding
Simulation of slidingwindow protocol
Enter the range of sliding window
2
SENDER          RECIEVER

0
[
1
2
]
3
                        Ack for1
```

0

1

[

2

3

Ack for2

0

1

2

[

3

Ack for3

0

1

2

3

There are no frames to be sent