1. Write a program to demonstrate bit-stuffing technique. Assume the flag pattern is 111110. Your program should stuff bits accordingly and de-stuff the shifted bit string.

CODE

```
import java.util.*;

public class BitStuffing {

  String strnew;

   public void addBits(){
      Scanner sc=new Scanner(System.in);
      System.out.println("Enter The String : ");
      String str=sc.nextLine();
      String match="1111";
      String newmatch="11110";
      if(str.contains(match)){
         strnew=str.replace(match,newmatch);
         System.out.println("The message after stuffing is " + strnew);
      }
   }

   public void deleteBits(){
     System.out.println("Destuffing Bits");
     System.out.println("The received message is " + strnew);
     String str = strnew;
     String match = "11110";
     String newMatch = "1111";
     String newString;
     if(str.contains(match)){
       newString = str.replace(match,newMatch);
       System.out.println("The message after destuffing " + newString);
     }
   }

    public static void main(String[] args) {
        BitStuffing bs=new BitStuffing();
        bs.addBits();
        bs.deleteBits();
    }
}
```

OUTPUT

```
Enter The String :
111111111111
The message after stuffing is 111101111011110
Destuffing Bits
The received message is 111101111011110
The message after destuffing 111111111111
```

2. Write a program to detect error using CRC technique. The message to transmit is M=1110001101, divisor polynomial D=$x^5+x^3+1$. Your program output should show the transmitting bit-stream along with CRC, error detected if any and the received message.

CODE

```
import java.io.*;
class CRC
{
   public static void main(String args[]) throws IOException
   {
      BufferedReader br=new BufferedReader(new
InputStreamReader(System.in));
      int[] data;
      int[] div;
      int[] divisor;
      int[] rem;
      int[] crc;
      int data_bits, divisor_bits, tot_length;

      System.out.println("Enter number of data bits : ");
      data_bits=Integer.parseInt(br.readLine());
      data=new int[data_bits];

      System.out.println("Enter data bits : ");
      for(int i=0; i<data_bits; i++)
         data[i]=Integer.parseInt(br.readLine());

      System.out.println("Enter number of bits in divisor : ");
      divisor_bits=Integer.parseInt(br.readLine());
      divisor=new int[divisor_bits];

      System.out.println("Enter Divisor bits : ");
      for(int i=0; i<divisor_bits; i++)
         divisor[i]=Integer.parseInt(br.readLine());


      System.out.print("Data bits are : ");
      for(int i=0; i< data_bits; i++)
         System.out.print(data[i]);
      System.out.println();

      System.out.print("divisor bits are : ");
      for(int i=0; i< divisor_bits; i++)
         System.out.print(divisor[i]);
      System.out.println();

     tot_length=data_bits+divisor_bits-1;

      div=new int[tot_length];
```

```
    rem=new int[tot_length];
    crc=new int[tot_length];
/*------------------ CRC GENERATION-----------------------*/
    for(int i=0;i<data.length;i++)
        div[i]=data[i];

    System.out.print("Dividend (after appending 0's) are : ");
    for(int i=0; i< div.length; i++)
        System.out.print(div[i]);
    System.out.println();

    for(int j=0; j<div.length; j++){
          rem[j] = div[j];
    }

    rem=divide(div, divisor, rem);

    for(int i=0;i<div.length;i++)          //append dividend and ramainder
    {
        crc[i]=(div[i]^rem[i]);
    }

    System.out.println();
    System.out.println("CRC code : ");
    for(int i=0;i<crc.length;i++)
        System.out.print(crc[i]);

/*-------------------ERROR DETECTION----------------------*/
    System.out.println();
    System.out.println("Enter CRC code of "+tot_length+" bits : ");
    for(int i=0; i<crc.length; i++)
        crc[i]=Integer.parseInt(br.readLine());


    System.out.print("crc bits are : ");
    for(int i=0; i< crc.length; i++)
        System.out.print(crc[i]);
    System.out.println();

    for(int j=0; j<crc.length; j++){
          rem[j] = crc[j];
    }

    rem=divide(crc, divisor, rem);

    for(int i=0; i< rem.length; i++)
    {
        if(rem[i]!=0)
        {
```

```
                System.out.println("Error");
                break;
            }
            if(i==rem.length-1)
                System.out.println("No Error");
        }

        System.out.println("THANK YOU.... :)");
    }

    static int[] divide(int div[],int divisor[], int rem[])
    {
        int cur=0;
        while(true)
        {
            for(int i=0;i<divisor.length;i++)
                rem[cur+i]=(rem[cur+i]^divisor[i]);

            while(rem[cur]==0 && cur!=rem.length-1)
                cur++;

            if((rem.length-cur)<divisor.length)
                break;
        }
        return rem;
    }
}
```

OUTPUT

```
Enter number of data bits :
10
Enter data bits :
1
1
1
0
0
0
1
1
0
1
Enter number of bits in divisor :
6
Enter Divisor bits :
1
0
1
0
0
```

1
Data bits are : 1110001101
divisor bits are : 101001
Dividend (after appending 0's) are : 111000110100000

CRC code :
111000110110110
Enter CRC code of 15 bits :
1
1
1
0
0
0
1
1
0
1
1
0
1
1
1
crc bits are : 111000110110111
Error
THANK YOU.... :)

3. Write a program of hamming code generation for error detection /correction. Message will be input by the user. Program should be able to calculate the number of redundant bits. The final output should show the transmitting stream of bits. The user will be asked introduce error in the transmitting message. Program should detect the position of the error and rectify it. Finally the corrected message should be shown.

CODE

```
import java.util.Scanner;

public class hamcode {
	public static void main(String args[]) {
		Scanner scan = new Scanner(System.in);
		System.out.println("Enter the number of bits for the Hamming data:");
		int n = scan.nextInt();
		int a[] = new int[n];

		for(int i=0 ; i < n ; i++) {
			System.out.println("Enter bit no. " + (n-i) + ":");
			a[n-i-1] = scan.nextInt();
		}

		System.out.println("You entered:");
		for(int i=0 ; i < n ; i++) {
			System.out.print(a[n-i-1]);
		}
		System.out.println();

		int b[] = generateCode(a);

		System.out.println("Generated code is:");
		for(int i=0 ; i < b.length ; i++) {
			System.out.print(b[b.length-i-1]);
		}
		System.out.println();

		// Difference in the sizes of original and new array will give us the number of parity bits added.
		System.out.println("Enter position of a bit to alter to check for error detection at the receiver end (0 for no error):");
		int error = scan.nextInt();
		if(error != 0) {
			b[error-1] = (b[error-1]+1)%2;
		}
		System.out.println("Sent code is:");
		for(int i=0 ; i < b.length ; i++) {
			System.out.print(b[b.length-i-1]);
		}
		System.out.println();
```

```
		receive(b, b.length - a.length);
	}

	static int[] generateCode(int a[]) {
		// We will return the array 'b'.
		int b[];

		// We find the number of parity bits required:
		int i=0, parity_count=0 ,j=0, k=0;
		while(i < a.length) {
			// 2^(parity bits) must equal the current position
			// Current position is (number of bits traversed + number
of parity bits + 1).
			// +1 is needed since array indices start from 0 whereas we
need to start from 1.

			if(Math.pow(2,parity_count) == i+parity_count + 1) {
				parity_count++;
			}
			else {
				i++;
			}
		}

		// Length of 'b' is length of original data (a) + number of parity
bits.
		b = new int[a.length + parity_count];

		// Initialize this array with '2' to indicate an 'unset' value in parity
bit locations:

		for(i=1 ; i <= b.length ; i++) {
			if(Math.pow(2, j) == i) {
			// Found a parity bit location.
			// Adjusting with (-1) to account for array indices starting
from 0 instead of 1.

				b[i-1] = 2;
				j++;
			}
			else {
				b[k+j] = a[k++];
			}
		}
		for(i=0 ; i < parity_count ; i++) {
			// Setting even parity bits at parity bit locations:

			b[((int) Math.pow(2, i))-1] = getParity(b, i);
		}
```

```
		return b;
	}

	static int getParity(int b[], int power) {
		int parity = 0;
		for(int i=0 ; i < b.length ; i++) {
			if(b[i] != 2) {
				// If 'i' doesn't contain an unset value,
				// We will save that index value in k, increase it by
1,
				// Then we convert it into binary:

				int k = i+1;
				String s = Integer.toBinaryString(k);

				//Nw if the bit at the 2^(power) location of the binary
value of index is 1
				//Then we need to check the value stored at that
location.
				//Checking if that value is 1 or 0, we will calculate
the parity value.

				int x = ((Integer.parseInt(s))/((int) Math.pow(10,
power)))%10;
				if(x == 1) {
					if(b[i] == 1) {
						parity = (parity+1)%2;
					}
				}
			}
		}
		return parity;
	}

	static void receive(int a[], int parity_count) {
		// This is the receiver code. It receives a Hamming code in array
'a'.
		// We also require the number of parity bits added to the original
data.
		// Now it must detect the error and correct it, if any.

		int power;
		// We shall use the value stored in 'power' to find the correct bits
to check for parity.

		int parity[] = new int[parity_count];
		// 'parity' array will store the values of the parity checks.

		String syndrome = new String();
```

```
		// 'syndrome' string will be used to store the integer value of error
location.

		for(power=0 ; power < parity_count ; power++) {
		// We need to check the parities, the same no of times as the no
of parity bits added.

			for(int i=0 ; i < a.length ; i++) {
				// Extracting the bit from 2^(power):

				int k = i+1;
				String s = Integer.toBinaryString(k);
				int bit = ((Integer.parseInt(s))/((int) Math.pow(10,
power)))%10;
				if(bit == 1) {
					if(a[i] == 1) {
						parity[power] = (parity[power]+1)%2;
					}
				}
			}
			syndrome = parity[power] + syndrome;
		}
		// This gives us the parity check equation values.
		// Using these values, we will now check if there is a single bit
error and then correct it.

		int error_location = Integer.parseInt(syndrome, 2);
		if(error_location != 0) {
			System.out.println("Error is at location " + error_location +
".");
			a[error_location-1] = (a[error_location-1]+1)%2;
			System.out.println("Corrected code is:");
			for(int i=0 ; i < a.length ; i++) {
				System.out.print(a[a.length-i-1]);
			}
			System.out.println();
		}
		else {
			System.out.println("There is no error in the received data.");
		}

		// Finally, we shall extract the original data from the received
(and corrected) code:
		System.out.println("Original data sent was:");
		power = parity_count-1;
		for(int i=a.length ; i > 0 ; i--) {
			if(Math.pow(2, power) != i) {
				System.out.print(a[i-1]);
			}
```

```
                        else {
                                power--;
                        }
                }
                System.out.println();
        }

}
```

OUTPUT

```
Enter the number of bits for the Hamming data:
5
Enter bit no. 5:
1
Enter bit no. 4:
0
Enter bit no. 3:
1
Enter bit no. 2:
1
Enter bit no. 1:
0
You entered:
10110
Generated code is:
110110010
Enter position of a bit to alter to check for error detection at the receiver end (0 for no error):
3
Sent code is:
110110110
Error is at location 3.
Corrected code is:
110110010
Original data sent was:
10110
```

4. Write a program of error detection using Checksum technique. The message will be input by the user. The number of bits in the input message should be in multiple of 4. Program should dynamically divide the message taking 4 bits in a slice i.e. if number of input bits is 20, number of slices will be 5. The output of the program should show the transmitting message slices.

CODE

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

class Checksum1{
  public static void main(String[] args){

    //sender
    System.out.println("Enter the number of bits(multiple of 4 only):");
    Scanner scan = new Scanner(System.in);
    int n = scan.nextInt();
    int[] array = new int[n];
    System.out.println("Enter the bits to be sent:");
    for(int i = 0; i < n; i++){
      array[i] = scan.nextInt();
    }
    System.out.println("The input is:");
    for(int i = 0; i < n; i++){
      System.out.print(array[i] + " ");
    }
    int x = n / 4;
    int[] bits = new int[4];
    for(int i = 0; i<4; i++){
      int temp = 0;
      for(int j = i; j < n-(3-i); j+=4){
        temp = temp ^ array[j];
      }
      bits[i] = temp;
    }
    System.out.println("\n" + "The checksum is:");
    for(int i = 0; i < 4; i++){
      System.out.print(bits[i] + " ");
    }
    System.out.println("\n" + "The sending message is:");
    int[] send = new int[n+4];
    for(int i = 0; i < n; i++){
      send[i] = array[i];
    }
    for(int i = 0; i < 4; i++){
      send[n+i] = bits[i];
    }
    for(int i = 0; i < n+4; i++){
      System.out.print(send[i] + " ");
    }
```

```
    //receiver
    int n1 = n+4;
    int[] array1 = new int[n1];
    System.out.println("\n" + "Enter the received bits:");
    for(int i = 0; i < n1; i++){
      array1[i] = scan.nextInt();
    }
    System.out.println("The received message is:");
    for(int i = 0; i < n1; i++){
      System.out.print(array1[i] + " ");
    }
    int x1 = n1 / 4;
    int[] bits1 = new int[4];
    for(int i = 0; i<4; i++){
      int temp1 = 0;
      for(int j = i; j < n1-(3-i); j+=4){
        temp1 = temp1 ^ array1[j];
      }
      bits1[i] = temp1;
    }
    System.out.println("\n" + "The checksum is:");
    for(int i = 0; i < 4; i++){
      System.out.print(bits1[i] + " ");
    }
    int sum = 0;
    for(int i = 0; i<4; i++){
      sum = sum + bits1[i];
    }
    if(sum > 0){
      System.out.println("\n" + "Error");
    }else{
      System.out.println("\n" + "Correct");
    }
/*
    String string = "";
    for (int i = 0; i < bits1.length; i++) {
        string = string + bits1[i];
    }
    System.out.print(string);
    */
  }
}
```

OUTPUT

```
Enter the number of bits(multiple of 4 only):
8
Enter the bits to be sent:
1 1 1 1 0 0 0 0
```

The input is:
1 1 1 1 0 0 0 0
The checksum is:
1 1 1 1
The sending message is:
1 1 1 1 0 0 0 0 1 1 1 1
Enter the received bits:
1 1 1 1 0 0 1 0 1 1 1 1
The received message is:
1 1 1 1 0 0 1 0 1 1 1 1
The checksum is:
0 0 1 0
Error

5. Write a program to identify the class of a given IP address. Address will be input in dotted decimal format.

CODE

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

public class LocalAddress
{
  public static void main(String[] args)
  {
    System.out.println("Enter the host name :");
    Scanner sc = new Scanner(System.in);
    String address = sc.nextLine();
    String firstTriplet = address.substring(0,address.indexOf('.'));
    if(Integer.parseInt(firstTriplet) > 0 && Integer.parseInt(firstTriplet) < 255){
      if (Integer.parseInt(firstTriplet) < 128) {
        System.out.println("Class A IP");
      } else if (Integer.parseInt(firstTriplet) < 192) {
        System.out.println("Class B IP");
      } else if(Integer.parseInt(firstTriplet) < 224) {
        System.out.println("Class C IP");
      }else if(Integer.parseInt(firstTriplet) < 240) {
        System.out.println("Class D IP");
      }else{
        System.out.println("Class E IP");
      }
    }else{
      System.out.println("Wrong IP");
    }
  }
}
```

OUTPUT

```
Enter the host name :
192.168.5.247
Class C IP
```

6. Write a program using TCP socket to do the following jobs :
   a. Echo message

CODE

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

public class EchoServer
{
    public EchoServer(int portnum)
    {
            try
            {
                    server = new ServerSocket(portnum);
            }
            catch (Exception err)
            {
                    System.out.println(err);
            }
    }

    public void serve()
    {
            try
            {
                    while (true)
                    {
                            Socket client = server.accept();
                            BufferedReader r = new BufferedReader(new
InputStreamReader(client.getInputStream()));
                            PrintWriter w = new
PrintWriter(client.getOutputStream(), true);
                            w.println("Welcome to the Java EchoServer.  Type
'bye' to close.");
                            String line;
                            do
                            {
                                    line = r.readLine();
                                    if ( line != null )
                                            w.println("Got: "+ line.toUpperCase() + "
" + new StringBuilder(line).reverse().toString() );

                            }
                            while ( !line.trim().equals("bye") );
                            client.close();
                            server.close();
                    }
            }
            catch (Exception err)
```

```
            {
                System.err.println(err);
            }
    }

    public static void main(String[] args)
    {
            EchoServer s = new EchoServer(9999);
            System.out.println("Server Started..");
            s.serve();
    }

    private ServerSocket server;
}

//Client
import java.io.*;
import java.net.*;

public class EchoClient
{
    public static void main(String[] args)
    {
            try
            {
                Socket s = new Socket("127.0.0.1", 9999);
                BufferedReader r = new BufferedReader(new
InputStreamReader(s.getInputStream()));
                PrintWriter w = new PrintWriter(s.getOutputStream(), true);
                BufferedReader con = new BufferedReader(new
InputStreamReader(System.in));
                String line;
                do
                {
                    line = r.readLine();
                    if ( line != null )
                        System.out.println(line);
                    line = con.readLine();
                    w.println(line);
                }
                while ( !line.trim().equals("bye") );
            }
            catch (Exception err)
            {
                System.err.println(err);
            }
    }
}
```

OUTPUT

```
// Server
Server Started..
java.net.SocketException: Socket is closed

//Client
Welcome to the Java EchoServer.  Type 'bye' to close.
Hello
Got: HELLO olleH
45
Got: 45 54
456
Got: 456 654
bye
```

b. Menu driven mathematical operation

CODE

```
//Server
import java.io.*;
 import java.net.*;
 public class Server1 {
     /**
      * @param args
      */
     public static void main(String[] args) {
         int port=5001;
         try {
             ServerSocket serverSocket=new ServerSocket(port);
             System.out.println("SERVER WAITING");
             Socket clientSocket=serverSocket.accept();
             BufferedReader fromClient=new BufferedReader(new
InputStreamReader(clientSocket.getInputStream()));
             PrintStream toClient=new
PrintStream(clientSocket.getOutputStream());
             while(true)
             {
                 int choice=Integer.parseInt(fromClient.readLine());
                 System.out.print(choice);
                 switch(choice)
                 {
                 case 1:
                     int no=Integer.parseInt(fromClient.readLine());
                     int result=1;
                     if (no!=0 && no!=1){
                         for(int i=2;i<=no;i++)
                             result*=i;
                     }
                     toClient.flush();
                     toClient.println(result);
                     break;
                 case 2:
                     no=Integer.parseInt(fromClient.readLine());
                     result=0;
                     while(no!=0){
                         result+=no%10;
                         no/=10;
                     }
                     toClient.flush();
                     toClient.println(result);
                     break;
                 case 3:
                     int stringChoice;
                     do{
```

```
stringChoice=Integer.parseInt(fromClient.readLine());;
switch(stringChoice){
case 1:
    String first_str=fromClient.readLine();
    String second_str=fromClient.readLine();
    int n1=Integer.parseInt(first_str);
    int n2 = Integer.parseInt(second_str);
    result = n1 + n2;
    String str_result = Integer.toString(result);
    /*result=first_str.indexOf(second_str);
    if(result==-1)
        str_result="NO";
    else
        str_result="YES"; */
    toClient.flush();
    toClient.println(str_result);
    break;
case 2:
    first_str=fromClient.readLine();
    second_str=fromClient.readLine();
    n1=Integer.parseInt(first_str);
    n2 = Integer.parseInt(second_str);
    result = n1 - n2;
    str_result = Integer.toString(result);
    /*result=first_str.indexOf(second_str);
    if(result==-1)
        str_result="NO";
    else
        str_result="YES"; */
    toClient.flush();
    toClient.println(str_result);
    break;
case 3:
    first_str=fromClient.readLine();
    second_str=fromClient.readLine();
    n1=Integer.parseInt(first_str);
    n2 = Integer.parseInt(second_str);
    result = n1 * n2;
    str_result = Integer.toString(result);
    /*result=first_str.indexOf(second_str);
    if(result==-1)
        str_result="NO";
    else
        str_result="YES"; */
    toClient.flush();
    toClient.println(str_result);
    break;
case 4:
    first_str=fromClient.readLine();
```

```
                    second_str=fromClient.readLine();
                    n1=Integer.parseInt(first_str);
                    n2 = Integer.parseInt(second_str);
                    result = n1 / n2;
                    str_result = Integer.toString(result);
                    /*result=first_str.indexOf(second_str);
                    if(result==-1)
                        str_result="NO";
                    else
                        str_result="YES"; */
                    toClient.flush();
                    toClient.println(str_result);
                    break;
                case 5:
                    first_str=fromClient.readLine();
                    second_str=fromClient.readLine();
                    n1=Integer.parseInt(first_str);
                    n2 = Integer.parseInt(second_str);
                    result = n1 % n2;
                    str_result = Integer.toString(result);
                    /*result=first_str.indexOf(second_str);
                    if(result==-1)
                        str_result="NO";
                    else
                        str_result="YES"; */
                    toClient.flush();
                    toClient.println(str_result);
                    break;
                case 6:
                    break;
                }
            }while(stringChoice!=6);
            break;
        case 4:
            break;
        }
    }
    } catch (Exception e) {
        // TODO Auto-generated catch block
    }
  }
}

//Client
import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.io.PrintStream;
import java.net.ServerSocket;
import java.net.Socket;
```

```
public class Client1 {
    public static void main(String[] args) {
        int port=5001;
        try {
            Socket clientSocket=new Socket("localhost",port);
            System.out.println("Client Started");
            InputStreamReader isr = new InputStreamReader(System.in);
            BufferedReader br = new BufferedReader(isr);
            PrintStream toServer=new
PrintStream(clientSocket.getOutputStream());
            BufferedReader fromServer=new BufferedReader(new
InputStreamReader(clientSocket.getInputStream()));
            String menu="SOCKET PROGRAMMING
MENU\n1.FACTORIAL\n2.SUM OF DIGITS OF A
NO\n3.Calculator\n4.EXIT\nENTER UR CHOICE";
            int choice=-1;
            do
            {
                toServer.flush();
                System.out.println(menu);
                choice=Integer.parseInt(br.readLine());
                toServer.println(choice);
                toServer.flush();
                switch(choice)
                {
                case 1:
                    System.out.println("ENTER A NO:");
                    int no=Integer.parseInt(br.readLine());
                    toServer.println(no);
                    int result=Integer.parseInt(fromServer.readLine());
                    System.out.println("FACTORIAL IS:"+result);
                    break;
                case 2:
                    System.out.println("ENTER A NO:");
                    no=Integer.parseInt(br.readLine());
                    toServer.println(no);
                    result=Integer.parseInt(fromServer.readLine());
                    System.out.println("SUM OF EACH DIGITS IS:"+result);
                    break;
                case 3:
                    String stringmenu="Calculator OPERATION
MENU\n1.sum\n2.diff\n3.mul\n4.div\n5.remainder\n6.EXIT\nENTER UR
CHOICE";
                    int stringChoice;
                    do
                    {
                        System.out.println(stringmenu);
                        stringChoice=Integer.parseInt(br.readLine());
                        toServer.println(stringChoice);
```

```
toServer.flush();
switch(stringChoice)
{
case 1:
    System.out.println("ENTER First no:");
    String first_str=br.readLine();
    toServer.println(first_str);
    System.out.println("ENTER Second no:");
    String second_str=br.readLine();
    toServer.println(second_str);
    String str_result=fromServer.readLine();
    System.out.println("add IS "+str_result);
    break;
case 2:
    System.out.println("ENTER First no:");
    first_str=br.readLine();
    toServer.println(first_str);
    System.out.println("ENTER Second no:");
    second_str=br.readLine();
    toServer.println(second_str);
    str_result=fromServer.readLine();
    System.out.println("diff IS "+str_result);
    break;
case 3:
    System.out.println("ENTER First no:");
    first_str=br.readLine();
    toServer.println(first_str);
    System.out.println("ENTER Second no:");
    second_str=br.readLine();
    toServer.println(second_str);
    str_result=fromServer.readLine();
    System.out.println("mul IS "+str_result);
    break;
case 4:
    System.out.println("ENTER First no:");
    first_str=br.readLine();
    toServer.println(first_str);
    System.out.println("ENTER Second no:");
    second_str=br.readLine();
    toServer.println(second_str);
    str_result=fromServer.readLine();
    System.out.println("div IS "+str_result);
    break;
case 5:
    System.out.println("ENTER First no:");
    first_str=br.readLine();
    toServer.println(first_str);
    System.out.println("ENTER Second no:");
    second_str=br.readLine();
```

```
                        toServer.println(second_str);
                        str_result=fromServer.readLine();
                        System.out.println("rem IS "+str_result);
                        break;
                    case 6:
                        break;
                    }
                }while(stringChoice!=6);
                break;
            case 4:
                break;
            default:
                System.out.println("INVALID CHOICE");
            }
        }while(choice!=4);
    } catch (Exception e) {
        // TODO Auto-generated catch block
    }
  }
}
```

OUTPUT

```
//Server
SERVER WAITING
34
//Client
Client Started
SOCKET PROGRAMMING MENU
1.FACTORIAL
2.SUM OF DIGITS OF A NO
3.Calculator
4.EXIT
ENTER UR CHOICE
3
Calculator OPERATION MENU
1.sum
2.diff
3.mul
4.div
5.remainder
6.EXIT
ENTER UR CHOICE
3
ENTER First no:
3
ENTER Second no:
5
mul IS 15
Calculator OPERATION MENU
1.sum
```

2.diff
3.mul
4.div
5.remainder
6.EXIT
ENTER UR CHOICE
6
SOCKET PROGRAMMING MENU
1.FACTORIAL
2.SUM OF DIGITS OF A NO
3.Calculator
4.EXIT
ENTER UR CHOICE
4

---

c. Client will send “DAYTIME” to the server and the server will be send back the corresponding date and time with a greetings like “ Good Morning” or “Good Evening” depending on the time.

CODE

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

public class DaytimeServer{

    public static void main(String[] args) throws IOException{
            ServerSocket s1=new ServerSocket(1342);
            System.out.println("Server started.....");
            Socket ss=s1.accept();
            Scanner sc=new Scanner(ss.getInputStream());
            PrintStream p=new PrintStream(ss.getOutputStream());
            String t1 = sc.nextLine();
    String datetimestring = (Calendar.getInstance()).getTime().toString();
            String hours = datetimestring.substring(11,13);
            int hr = Integer.parseInt(hours);

   if(t1.equals("DAYTIME")){
     if(hr >= 5 && hr < 12){
                        p.println("Good Morning " + datetimestring);
                }else if(hr >= 12 && hr < 21){
                        p.println("Good Evening " + datetimestring);
                }else{
                        p.println("Good Night " + datetimestring);
                }
   }else{
     p.println("Wrong Input From Client ");
   }
            System.out.println("received " + t1);
    }
}

//Client
import java.io.IOException;
import java.io.PrintStream;
import java.net.Socket;
import java.net.UnknownHostException;
import java.util.Scanner;

public class DaytimeClient {

    public static void main(String[] args)throws IOException{
```

```
            Scanner sc=new Scanner(System.in);
            Socket s=new Socket("127.0.0.1",1342);
            Scanner sc1=new Scanner(s.getInputStream());

    System.out.println("Type the word 'DAYTIME' below....");

    String time = sc.nextLine();
            PrintStream p=new PrintStream(s.getOutputStream());

    p.println(time);

    String temp = sc1.nextLine();
            System.out.println(temp);


     }
}
```

OUTPUT

```
//Server
Server started.....
received DAYTIME

//Client
Type the word 'DAYTIME' below....
DAYTIME
Good Morning Sun May 28 10:32:54 IST 2017
```

7. Write a program using UDP socket to do the following jobs :
   d. Echo message

CODE

```
//Server
import java.net.*;
import java.io.*;
public class UDPEchoServer
{
 public static void main(String args[])
 {
 int port = 8000;
 // create the server...
 DatagramSocket serverDatagramSocket = null;
 try
 {
 serverDatagramSocket = new DatagramSocket(port);
 System.out.println("Created UDP Echo Server on port" + port);
 }
 catch(IOException e)
 {
 System.out.println(e);
 System.exit(1);
 }
 try
 {
 byte buffer[] = new byte[1024];
 DatagramPacket datagramPacket = new
 DatagramPacket(buffer, buffer.length);
 String input;
 while(true)
 {
 // listen for datagram packets
 serverDatagramSocket.receive(datagramPacket);
 input = new String(datagramPacket.getData(), 0,
 datagramPacket.getLength());
 System.out.println("Received from server:" + input);
 // send received packet back to the client
 serverDatagramSocket.send(datagramPacket);
 }
 }
 catch(IOException e)
 {
 System.out.println(e);
 }
 }
}

//Client
import java.net.*;
```

```
import java.io.*;
public class UDPEchoClient
{
 public static class UDPEchoReader extends Thread
 {
 public UDPEchoReader(DatagramSocket socket)
 {
 datagramSocket = socket;
 active = true;
 }
 public void run()
 {
 byte[] buffer = new byte[1024];
 DatagramPacket incoming = new DatagramPacket(buffer,
 buffer.length);
 String receivedString;
 while(active)
 {
 try
 {
 // listen for incoming datagram packet
 datagramSocket.receive(incoming);
 // print out received string
 receivedString = new String(incoming.getData(),
 0, incoming.getLength());
 System.out.println("Received from server:" + receivedString);
 }
 catch(IOException e)
 {
 System.out.println(e);
 active = false;
 }
 }
 }
 public boolean active;
 public DatagramSocket datagramSocket;
 }
 public static void main(String[] args)
 {
 InetAddress address = null;
 int port = 8000;
 DatagramSocket datagramSocket = null;
 BufferedReader keyboardReader = null;
 // Create a Datagram Socket...
 try
 {
 address = InetAddress.getByName("127.0.0.1");
 datagramSocket = new DatagramSocket();
 keyboardReader = new BufferedReader(new
```

```
 InputStreamReader(System.in));
 }
 catch (IOException e)
 {
 System.out.println(e);
 System.exit(1);
 }
 // Start the listening thread...
 UDPEchoReader reader = new UDPEchoReader(datagramSocket);
 reader.setDaemon(true);
 reader.start();
 System.out.println("Ready to send your messages...");
 try
 {
 String input;
 while (true)
 {
 // read input from the keyboard
 input = keyboardReader.readLine();
 // send datagram packet to the server
 DatagramPacket datagramPacket = new DatagramPacket
 (input.getBytes(), input.length(), address, port);
 datagramSocket.send(datagramPacket);
 }
 }
 catch(IOException e)
 {
 System.out.println(e);
 }
 }
}
```

OUTPUT

```
//Server
Created UDP Echo Server on port8000
Received from server:Hello
Received from server:56
//Client
Ready to send your messages...
Hello
Received from server:Hello
56
Received from server:56
```

e. Menu driven mathematical operation

CODE

OUTPUT

f. Client will send “DAYTIME” to the server and the server will be send back the corresponding date and time with a greetings like “ Good Morning” or “Good Evening” depending on the time.

CODE

OUTPUT

8. Write a program to demonstrate RSA algorithm. Try RSA algorithm using Socket programming.

CODE

```
//Server
import java.math.BigInteger;
import java.security.*;
import java.security.spec.*;
import java.io.*;
import javax.crypto.Cipher;
import java.net.*;
import java.util.*;

public class RSAServer
{
public static void main(String args[])throws IOException
{

  ServerSocket s1=new ServerSocket(1342);
  System.out.println("Server started.....");
  Socket ss=s1.accept();
  Scanner sc=new Scanner(ss.getInputStream());
  PrintStream p=new PrintStream(ss.getOutputStream());

  String srci = sc.nextLine();

  /*try{
  BufferedReader br=new BufferedReader(new
InputStreamReader(System.in));
  System.out.println("Please enter any string you want to encrypt");
  srci=br.readLine();
  }
  catch(IOException ioe)
  {
  System.out.println(ioe.getMessage());
}*/

  try{
  KeyPairGenerator kpg=KeyPairGenerator.getInstance("RSA");
  kpg.initialize(512);//initialize key pairs to 512 bits ,you can also take 1024
or 2048 bits
```

```
  KeyPair kp=kpg.genKeyPair();
  System.out.println(kp);
  PublicKey publi=kp.getPublic();
  System.out.println(publi);
  p.println(publi);
  Cipher cipher = Cipher.getInstance("RSA");
  cipher.init(Cipher.ENCRYPT_MODE, publi);
  byte[] src=srci.getBytes();//converting source data into byte array
  byte[] cipherData = cipher.doFinal(src);//use this method to finally encrypt
data
  String srco=new String(cipherData);//converting byte array into string
  System.out.println();
  //System.out.println("Encrypted data is:-"+srco);
  //p.println(srco);
  PrivateKey privatei=kp.getPrivate();//Generating private key
  System.out.println(privatei);
  Cipher cipheri=Cipher.getInstance("RSA");//Intializing 2nd instance of
Cipher class
  cipheri.init(Cipher.DECRYPT_MODE, privatei);//Setting to decrypt_mode
  byte[] cipherDat = cipheri.doFinal(cipherData);//Finally decrypting data
  String decryptdata=new String(cipherDat);
  System.out.println("Decrypted data:-"+decryptdata);
  }
  catch(Exception e)
  {
  System.out.println(e.getMessage());
  }
}
}

//Client
import java.math.BigInteger;
import java.security.*;
import java.security.spec.*;
import java.io.*;
import javax.crypto.Cipher;
import java.net.*;
import java.util.*;

public class RSAClient
{
public static void main(String args[])throws IOException
{
  Scanner sc=new Scanner(System.in);
  Socket s=new Socket("127.0.0.1",1342);
  Scanner sc1=new Scanner(s.getInputStream());

  System.out.println("Enter message to encrypt:");
```

```
  String msg = sc.nextLine();
  PrintStream p=new PrintStream(s.getOutputStream());

  p.println(msg);

  String temp = sc1.nextLine();
  System.out.println("Encryption Complete...\n" + temp);
    String temp1 = sc1.nextLine();
    System.out.println(temp1);
    String temp2 = sc1.nextLine();
    System.out.println(temp2);
}
}
```

OUTPUT

```
//Server
Server started.....
java.security.KeyPair@11978b
Sun RSA public key, 512 bits
  modulus:
7398694417386517396913990233991170747871849542129875593868834675249159032576151488172645779902606177328624465569818232536597214164228319051723214666115449
  public exponent: 65537

sun.security.rsa.RSAPrivateCrtKeyImpl@4215
Decrypted data:-Hello

//Client
Enter message to encrypt:
Hello
Encryption Complete...
Sun RSA public key, 512 bits
  modulus:
7398694417386517396913990233991170747871849542129875593868834675249159032576151488172645779902606177328624465569818232536597214164228319051723214666115449
  public exponent: 65537
```

9. Write a program to simulate Distance-vector Routing for the following adjacency matrix. Assume source node is A, Destination node is F. Find out the shortest path.

| | A | B | C | D | E | F | G | H |
|---|---|---|---|---|---|---|---|---|
| A | 0 | 2 | Inf | 3 | Inf | 4 | Inf | 1 |
| B | 2 | 0 | 4 | 1 | 5 | Inf | 1 | Inf |
| C | Inf | 4 | 0 | 6 | 1 | 3 | 1 | Inf |
| D | 3 | 1 | 6 | 0 | 5 | Inf | Inf | 2 |
| E | Inf | 5 | 1 | 5 | 0 | 4 | 5 | 2 |
| F | 4 | Inf | 3 | Inf | 4 | 0 | 3 | Inf |
| G | Inf | 1 | 1 | Inf | 5 | 3 | 0 | Inf |
| H | 1 | Inf | 3 | 2 | 2 | Inf | Inf | 0 |

CODE

```
import java.util.Scanner;

public class BellmanFord
{
   private int distances[];
   private int numberofvertices;
   public static final int MAX_VALUE = 999;

   public BellmanFord(int numberofvertices)
   {
      this.numberofvertices = numberofvertices;
      distances = new int[numberofvertices + 1];
   }

   public void BellmanFordEvaluation(int source, int adjacencymatrix[][])
   {
      for (int node = 1; node <= numberofvertices; node++)
      {
         distances[node] = MAX_VALUE;
      }

      distances[source] = 0;
      for (int node = 1; node <= numberofvertices - 1; node++)
      {
         for (int sourcenode = 1; sourcenode <= numberofvertices; sourcenode++)
         {
            for (int destinationnode = 1; destinationnode <= numberofvertices;
destinationnode++)
            {
               if (adjacencymatrix[sourcenode][destinationnode] != MAX_VALUE)
               {
                  if (distances[destinationnode] > distances[sourcenode]
                        + adjacencymatrix[sourcenode][destinationnode])
                     distances[destinationnode] = distances[sourcenode]
                        + adjacencymatrix[sourcenode][destinationnode];
```

```
                }
            }
        }
    }

    for (int sourcenode = 1; sourcenode <= numberofvertices; sourcenode++)
    {
        for (int destinationnode = 1; destinationnode <= numberofvertices;
destinationnode++)
        {
            if (adjacencymatrix[sourcenode][destinationnode] != MAX_VALUE)
            {
                if (distances[destinationnode] > distances[sourcenode]
                        + adjacencymatrix[sourcenode][destinationnode])
                    System.out.println("The Graph contains negative egde cycle");
            }
        }
    }

    for (int vertex = 1; vertex <= numberofvertices; vertex++)
    {
        System.out.println("distance of source  " + source + " to "
                + vertex + " is " + distances[vertex]);
    }
  }

  public static void main(String... arg)
  {
    int numberofvertices = 0;
    int source;
    Scanner scanner = new Scanner(System.in);

    System.out.println("Enter the number of vertices");
    numberofvertices = scanner.nextInt();

    int adjacencymatrix[][] = new int[numberofvertices + 1][numberofvertices +
1];
    System.out.println("Enter the adjacency matrix");
    for (int sourcenode = 1; sourcenode <= numberofvertices; sourcenode++)
    {
        for (int destinationnode = 1; destinationnode <= numberofvertices;
destinationnode++)
        {
            adjacencymatrix[sourcenode][destinationnode] = scanner.nextInt();
              if (sourcenode == destinationnode)
            {
                adjacencymatrix[sourcenode][destinationnode] = 0;
                continue;
            }
```

```
                if (adjacencymatrix[sourcenode][destinationnode] == 0)
                {
                    adjacencymatrix[sourcenode][destinationnode] = MAX_VALUE;
                }
            }
        }

        System.out.println("Enter the source vertex");
        source = scanner.nextInt();

        BellmanFord bellmanford = new BellmanFord(numberofvertices);
        bellmanford.BellmanFordEvaluation(source, adjacencymatrix);
        scanner.close();
    }
}
```

OUTPUT

```
0 2 0 3 0 4 0 1

2 0 4 1 5 0 1 0

0 4 0 6 1 3 1 3

3 1 6 0 5 0 0 2

0 5 1 5 0 4 5 2

4 0 3 0 4 0 3 0

0 1 1 0 5 3 0 0

1 0 3 2 2 0 0 0
```

| Vertex | Distance | Path |
|---|---|---|
| 0 -> 1 | 2 | 0 ->1 |
| A->B | | A->B |
| 0 -> 2 | 4 | 0 ->7-> 2 |
| A->C | | A->H->C |
| 0 -> 3 | 3 | 0 ->3 |
| A->D | | A->D |
| 0 -> 4 | 3 | 0-> 7 ->4 |
| A->E | | A->H->E |
| 0 -> 5 | 4 | 0 ->5 |
| A->F | | A->F |

| | | |
|---|---|---|
| 0 -> 6 | 3 | 0 ->1-> 6 |
| A->G | | A->B->G |
| 0 -> 7 | 1 | 0 ->7 |
| A->H | | A->H |