

## PART-B

### 1. Write a program for error detecting code using CRC-CCITT (16- bits).

#### International Standard CRC Polynomials

##### Source 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 remainder
        {
            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:

```
File Edit View Terminal Help
[root@localhost ~]# javac Crc.java
[root@localhost ~]# java Crc
Enter number of data bits :
7
Enter data bits :
1
0
1
1
0
0
1
Enter number of bits in divisor :
3
Enter Divisor bits :
1
0
1
Dividend (after appending 0's) are : 101100100

CRC code :
101100111
Enter CRC code of 9 bits :
1
0
1
1
0
0
1
0
0
1
Error
THANK YOU.... :)
[root@localhost ~]#
```

## 2. Write a program to find the shortest path between vertices using bellman-ford algorithm.

### Source code:

```
import java.util.Scanner;

public class BellmanFord
{
    private int D[];
    private int num_ver;
    public static final int MAX_VALUE = 999;
    public BellmanFord(int num_ver)
    {
        this.num_ver = num_ver;
        D = new int[num_ver + 1];
    }
    public void BellmanFordEvaluation(int source, int A[][])
    {
        for (int node = 1; node <= num_ver; node++)
        {
            D[node] = MAX_VALUE;
        }
        D[source] = 0;
        for (int node = 1; node <= num_ver - 1; node++)
        {
            for (int sn = 1; sn <= num_ver; sn++)
            {
                for (int dn = 1; dn <= num_ver; dn++)
                {
                    if (A[sn][dn] != MAX_VALUE)
                    {
                        if (D[dn] > D[sn] + A[sn][dn])
                            D[dn] = D[sn] + A[sn][dn];
                    }
                }
            }
        }
    }
}
```

```

    }
    for (int sn = 1; sn <= num_ver; sn++)
    {
        for (int dn = 1; dn <= num_ver; dn++)
        {
            if (A[sn][dn] != MAX_VALUE)
            {
                if (D[dn] > D[sn] + A[sn][dn])
                System.out.println("The Graph contains negative egde cycle");
            }
        }
    }
    for (int vertex = 1; vertex <= num_ver; vertex++)
    {
        System.out.println("distance of source " + source + " to " + vertex + " is " + D[vertex]);
    }
}

```

```

public static void main(String[ ] args)
{
    int num_ver = 0;
    int source;
    Scanner scanner = new Scanner(System.in);
    System.out.println("Enter the number of
    vertices"); num_ver = scanner.nextInt();
    int A[][] = new int[num_ver + 1][num_ver + 1];
    System.out.println("Enter the adjacency matrix");
    for (int sn = 1; sn <= num_ver; sn++)
    {
        for (int dn = 1; dn <= num_ver; dn++)
        {
            A[sn][dn] = scanner.nextInt();
            if (sn == dn)
            {
                A[sn][dn] = 0;
                continue;
            }
        }
    }
}

```

```

    }
    if (A[sn][dn] == 0)
    {
        A[sn][dn] = MAX_VALUE;
    }
}
}

```

```

System.out.println("Enter the source vertex");
source = scanner.nextInt();
BellmanFord b = new BellmanFord (num_ver);
b.BellmanFordEvaluation(source, A);
scanner.close();

```

```

}

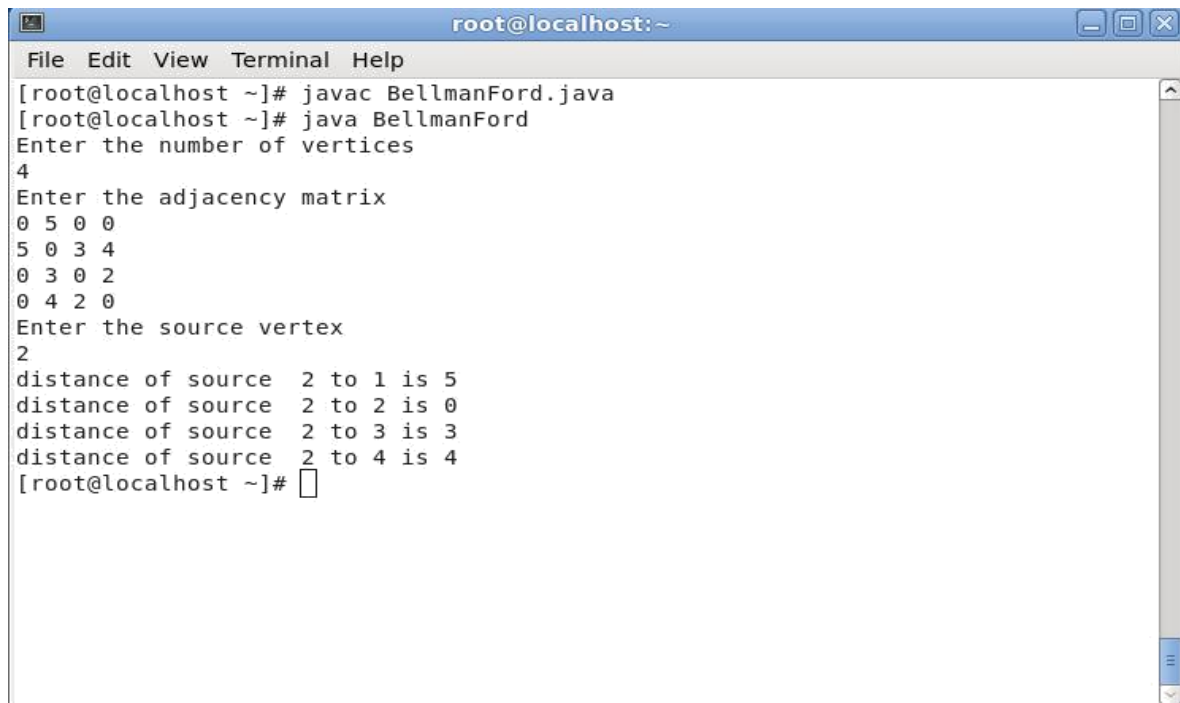
```

```

}

```

## Output:



```

root@localhost:~
File Edit View Terminal Help
[root@localhost ~]# javac BellmanFord.java
[root@localhost ~]# java BellmanFord
Enter the number of vertices
4
Enter the adjacency matrix
0 5 0 0
5 0 3 4
0 3 0 2
0 4 2 0
Enter the source vertex
2
distance of source 2 to 1 is 5
distance of source 2 to 2 is 0
distance of source 2 to 3 is 3
distance of source 2 to 4 is 4
[root@localhost ~]# 

```





**3. Using TCP/IP sockets, write a client – server program to make the client send the file name and to make the server send back the contents of the requested file if present. Implement the above program using as message queues or FIFOs as IPC channels.**

**Source Code:**

**TCP Server**

```
import java.net.*;

import java.io.*;

class TCPserver{

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

ServerSocket ss=new ServerSocket(4000);

System.out.println("Server ready for connection");

Socket s=ss.accept();

System.out.println("Connection is successful and waiting for chatting");

InputStream istream=s.getInputStream();

BufferedReader fileRead=new BufferedReader(new InputStreamReader(istream));

String fname=fileRead.readLine();

BufferedReader contentRead=new BufferedReader(new FileReader(fname));

OutputStream ostream=s.getOutputStream();

PrintWriter pwrite=new PrintWriter(ostream,true);

String str;

while((str=contentRead.readLine())!=null)

pwrite.println(str);
```

```

s.close();
ss.close();
pwrite.close();
fileRead.close();
contentRead.close();

}

}

```

## **How to Compile & Run This Program**

### **Terminal 1: TCPserver Program**

```

> javac TCPserver.java
> java TCPserver
Server ready for connection

```

### **TCP Client**

```

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

class TCPclient
{
    public static void main(String args[]) throws Exception {
        Socket s=new Socket("127.0.0.1",4000);

        System.out.println("Enter the file name");

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

        String fname=keyRead.readLine();

        OutputStream ostream=s.getOutputStream();

        PrintWriter pwrite=new PrintWriter(ostream,true);
    }
}

```

```
pwrite.println(fname);

InputStream istream=s.getInputStream();

BufferedReader socketRead=new BufferedReader(new
InputStreamReader(istream));

String str;

while((str=socketRead.readLine())!=null)

System.out.println(str);

pwrite.close();

socketRead.close();

keyRead.close();

}

}
```

### **How to Compile & Run This Program**

#### **Terminal 2: TCPclient Program**

>javac TCPclient.java

>java TCPclient

Enter the file name

**Example.txt**                      **(Give any existing File Name)**

Hi

Wel Come to Java World

**4. Write a program on datagram socket for client/server to display the messages on client side, typed at the server side.**

**Source Code:**

**UDP Server**

```
import java.io.*;

import java.net.*;

class UDPServer {

    public static void main(String args[])

    {

        try{

            String message=args[0];

            int serverPortNumber=Integer.parseInt(args[1]);

            ServerSocket connectionSocket=new ServerSocket(serverPortNumber);

            Socket dataSocket=connectionSocket.accept();

            PrintStream socketOutput=new PrintStream(dataSocket.getOutputStream());

            socketOutput.println(message);

            socketOutput.println("Sent response to client");

            socketOutput.flush();

            dataSocket.close();

            connectionSocket.close();

        }

        catch( Exception e)

        {

            e.printStackTrace();

        }

    }

}
```

```
}
```

```
}
```

### **How to Compile & Run This Program**

#### **Terminal 1: UDPServer Program**

```
> javac UDPServer.java
```

```
> java UDPServer "Hi Wel Come" 1000
```

## **UDP Client**

```
import java.io.*;
import java.net.*;
class UDPClient
{
    public static void main(String args[])
    {
        try
        {
            InetAddress acceptorHost=InetAddress.getByName(args[0]);
            int ServerPortNum=Integer.parseInt(args[1]);
            Socket clientSocket=new Socket(acceptorHost,ServerPortNum);
            BufferedReader br=new BufferedReader(new
            InputStreamReader(clientSocket.getInputStream()));
            System.out.println(br.readLine());
            clientSocket.close();
        }
        catch (Exception e)
        {
            e.printStackTrace();
        }
    }
}
```

## **Steps to Run the Program**

> javac UDPClient.java

> java UDPClient 127.0.0.1 1000

Hi Wel Come

**5. Write a program for simple RSA algorithm to encrypt and decrypt the data.**

**Source Code:**

**RSA Key Generation**

```
import java.util.*;
import java.math.BigInteger;
import java.lang.*;

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

        Random rand1=new
        Random(System.currentTimeMillis()); Random
        rand2=new Random(System.currentTimeMillis()*10);

        int pubkey=Integer.parseInt(args[0]);

        BigInteger bigB_p=BigInteger.probablePrime(32, rand1);
        BigInteger bigB_q=BigInteger.probablePrime(32, rand2);

        BigInteger bigB_n=bigB_p.multiply(bigB_q);

        BigInteger bigB_p_1=bigB_p.subtract(new BigInteger("1"));
        BigInteger bigB_q_1=bigB_q.subtract(new BigInteger("1"));

        BigInteger bigB_p_1_q_1=bigB_p_1.multiply(bigB_q_1);
```

```

while(true)
{

    BigInteger BigB_GCD=bigB_p_1_q_1.gcd(new
        BigInteger(""+pubkey));
    if(BigB_GCD.equals(BigInteger.ONE))

        {
            break;
        }
    pubkey++;
}
BigInteger bigB_pubkey=new BigInteger(""+pubkey);

BigInteger
bigB_prvkey=bigB_pubkey.modInverse(bigB_p_1_q_1);
System.out.println("public key : "+bigB_pubkey+","+bigB_n);
System.out.println("private key :
"+bigB_prvkey+","+bigB_n);
}
}

```

## Steps to Compile & Run the Programme

### Terminal 1: RSAkeygen Program

**> javac RSAkeygen.java**

**> java RSAkeygen 20**

**public key : 23,11070015003782698357**

**private key : 3850439998963313063,11070015003782698357**



## RSA Encryption and Decryption

```
import java.math.BigInteger;

import java.util.*;

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

        BigInteger bigB_pubkey = new
        BigInteger(args[0]); BigInteger bigB_prvkey =
        new BigInteger(args[1]); BigInteger bigB_n =
        new BigInteger(args[2]); int
        asciiVal=Integer.parseInt(args[3]);

        BigInteger bigB_val=new BigInteger(""+asciiVal);

        BigInteger
        bigB_cipherVal=bigB_val.modPow(bigB_pubkey, bigB_n);
        System.out.println("Cipher text: " + bigB_cipherVal);

        BigInteger
        bigB_plainVal=bigB_cipherVal.modPow(bigB_prvkey, bigB_n); int
        plainVal=bigB_plainVal.intValue();

        System.out.println("Plain text:" + plainVal);
    }
}
```

## **Steps to Compile & Run the Programme**

### **Terminal 1: RSAEncDec Program**

```
> javac RSAEncDec.java
```

```
> Java RSAEncDec 23 2293891546998192587 10551901122856148501 10
```

```
Cipher text: 10184959815136804524
```

```
Plain text:10
```

## 6 Write a program for congestion control using leaky bucket algorithm.

### Source Code:

```
import java.io.*;

import java.util.*;

class Queue

{
    int q[],f=0,r=0,size;

    void insert(int n)
    {
        Scanner in = new Scanner(System.in);
        q=new int[10];
        for(int i=0;i<n;i++)
        {
            System.out.print("\nEnter " + i + " element: ");
            int ele=in.nextInt();
            if(r+1>10)
            {

                System.out.println("\nQueue is full \nLost Packet:
                "+ele); break;

            }
            else
            {
                r++;
                q[i]=ele;
            }
        }
    }
}
```

```

        }

    }

}

void delete()
{
    Scanner in = new Scanner(System.in);
    Thread t=new Thread();

    if(r==0)
        System.out.print("\nQueue empty ");
    else
    {
        for(int i=f;i<r;i++)
        {
            try
            {
                t.sleep(1000);
            }

            catch(Exception e){}
            System.out.print("\nLeaked Packet: "+q[i]);
            f++;
        }
    }
    System.out.println();
}

}

```

```

class Leaky extends Thread
{
    public static void main(String ar[]) throws
    Exception {
        Queue q=new Queue();
        Scanner src=new Scanner(System.in);
        System.out.println("\nEnter the packets to be sent:");
        int size=src.nextInt();
        q. insert(size);
        q.delete();
    }
}

```

### **How To Compile & Run This Program**

**> javac Queue.java**

**>java Leaky**

Enter the packets to be sent:

7

Enter 0 element: 2

Enter 1 element: 3

Enter 2 element: 4

Enter 3 element: 5

Enter 4 element: 6

Enter 5 element: 7

Enter 6 element: 8

Leaked Packet: 2

Leaked Packet: 3

Leaked Packet: 4

Leaked Packet: 5

Leaked Packet: 6

Leaked Packet: 7

Leaked Packet: 8