Computer Networks Laboratory- 18CSL57

PART B (Implement the following in Java)

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

```
import java.util.Scanner;
public class crc {
      static int data[],cs[];
      static int g[]={1,0,0,0,1,0,0,0,0,0,0,1,0,0,0,0,1};
      static int n, i, e,c,pos;
      static int N=17;
      static void xor() {
             for(c=0;c<N;c++) cs[c]=((cs[c]==g[c])?0:1);
           }

           static void crc() {
             for(i=0;i<N;i++) cs[i]=data[i];
             do {
               if(cs[0]==1) xor();
               for(c=0;c<N-1;c++) cs[c]=cs[c+1];
               cs[c]=data[i++];
             }while(i<=n+N-1);
           }


      public static void main(String[] args) {
           cs=new int[100];



           Scanner br=new Scanner(System.in);
           System.out.println("Enter no of Data bits");
           n=br.nextInt();
           data=new int[100];
           System.out.println("\nEnter the data bits : ");
            for(int i=0;i<n;i++)
                data[i]=br.nextInt();


            System.out.println("\n\nCRC Divisor : ");
            for(int i=0;i<N;i++)
                 System.out.print(g[i]);


             for(i=n;i<n+N-1;i++) data[i]=0;

            System.out.println("\n\nModified Data is :  ");
            for(i=0;i<n+N-1;i++)
                System.out.print(data[i]);
             crc();
```

```
            System.out.println("\n\nCRC Checksum is : ");
            for(int i=0;i<N-1;i++)
            System.out.print(cs[i]);
            for(i=n;i<n+N-1;i++) data[i]=cs[i-n];

            System.out.println("\n\nFinal Codeword is :");
            for(i=0;i<n+N-1;i++)
          System.out.print(data[i]);

 System.out.println("\n\nTest Error detection 0(yes) 1(no) ? : ");
            e=br.nextInt();
            if(e==0) {
  System.out.println("Enter position where error is to inserted : ");
              pos=br.nextInt();

         data[pos]=(data[pos]==0)?1:0;

              System.out.println("\nErroneous data   : ");
              for(i=0;i<n+N-1;i++)
                    System.out.print(data[i]);
            }
            crc();

                    System.out.println("\n\nReceiver Checksum:");
                    for(int i=0;i<N;i++)
                         System.out.print(cs[i]);
          for(i=0;i<N-1;i++)
          {
               if(cs[i]!=0)
               {

                    System.out.println("\n\nERROR in Received
Codeword ");
                    System.exit(0);
               }
          }
          System.out.println("\nNo Error in Received Codeword");


    }
}
```

Output1:

```
Enter no of Data bits
4

Enter the data bits :
1
0
0
1


CRC Divisor :
10001000000100001

Modified Data is :
10010000000000000000

CRC Checksum is :
1001000100101001

Final Codeword is :
10011001000100101001

Test Error detection 0(yes) 1(no) ? :
0
Enter position where error is to inserted :
2

Erroneous data    :
10111001000100101001

Receiver Checksum:
00100000010000100

ERROR in Received Codeword
```

Output2:

```
Enter no of Data bits
4

Enter the data bits :
1
0
0
1


CRC Divisor :
10001000000100001

Modified Data is :
10010000000000000000

CRC Checksum is :
1001000100101001

Final Codeword is :
10011001000100101001

Test Error detection 0(yes) 1(no) ? :
1


Receiver Checksum:
00000000000000000
No Error in Received Codeword
```

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

```
import java.util.*;
class DVT
{
     public static void main(String args[])
     {
          int dist[][]=new int[20][20];
          int from[][]=new int[20][20];
          int costmat[][]=new int[10][10];
          int i,j,k,nodes;
         Scanner s=new Scanner(System.in);

         System.out.println("\nEnter the number of nodes : ");
         nodes=s.nextInt();

         System.out.println("\nEnter the cost matrix :\n");
         for(i=1;i<=nodes;i++)
         {
              for( j=1;j<=nodes;j++)
              {
                   costmat[i][j]=s.nextInt();

                   costmat[i][i]=0;

                   dist[i][j]=costmat[i][j];
                   from[i][j]=j;
              }
         }


         for( i=1;i<=nodes;i++)
         {
              for( j=1;j<=nodes;j++)
              {
                   for( k=1;k<=nodes;k++)
                   {
                        if((dist[i][j])>dist[i][k]+dist[k][j])
                        {
                             dist[i][j]=dist[i][k]+dist[k][j];
                             from[i][j]=k;
                        }
                   }
              }
         }
         for( i=1;i<=nodes;i++)
```

```
            {
                System.out.println("\n\nFrom Router Node :"+i);
                System.out.println("\nDesti Node\tNext-
                Hop\tdistance\n");
                for( j=1;j<=nodes;j++)
                {
                    System.out.println(j +"\t \t " +from[i][j]+"
                    \t\t "+dist[i][j]);
                }
            }
            System.out.println("\n\n");
        }
}
```

Output:

```
Enter the number of nodes :
5

Enter the cost matrix :

0 1 2 3 999
1 0 999 1 2
2 999 0 2 999
3 1 2 0 1
999 2 999 1 0


From Router Node :1

Desti Node      Next- Hop       distance

1                1               0
2                2               1
3                3               2
4                2               2
5                2               3


From Router Node :2

Desti Node      Next- Hop       distance

1                1               1
2                2               0
3                1               3
4                4               1
5                5               2
```

```
From Router Node :3

Desti Node      Next- Hop       distance

1                1               2
2                1               3
3                3               0
4                4               2
5                4               3


From Router Node :4

Desti Node      Next- Hop       distance

1                2               2
2                2               1
3                3               2
4                4               0
5                5               1


From Router Node :5

Desti Node      Next- Hop       distance

1                2               3
2                2               2
3                4               3
4                4               1
5                5               0
```

Program 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.

Server Program:

```
import java.net.*;
import java.io.*;
public class ContentsServer
{
  public static void main(String args[]) throws Exception
  {
 // establishing the connection with the server
     ServerSocket sersock = new ServerSocket(4000);
     System.out.println("Server ready for connection");
     Socket sock = sersock.accept();  // binding with port: 4000
     System.out.println("Connection is successful and wating for
     chatting");

     // reading the file name from client
     InputStream istream = sock.getInputStream( );
      BufferedReader fileRead =new BufferedReader(new
     InputStreamReader(istream));
     String fname = fileRead.readLine( );
     // reading file contents
     BufferedReader contentRead = new BufferedReader(new
     FileReader(fname) );

  // keeping output stream ready to send the contents
     OutputStream ostream = sock.getOutputStream( );
     PrintWriter pwrite = new PrintWriter(ostream, true);

     String str;
     // reading line-by-line from file

     while((str = contentRead.readLine()) !=  null)
     {
         pwrite.println(str); // sending each line to client
      }
     System.out.println("Contents of the file is sent…");
     sock.close();  sersock.close();  // closing network sockets
     pwrite.close();  fileRead.close(); contentRead.close();
  }
```

Client Program:

```
import java.net.*;
import java.io.*;
public class ContentsClient
{
  public static void main( String args[ ] ) throws Exception
  {
     Socket sock = new Socket( "127.0.0.1", 4000);

      // reading the file name from keyboard. Uses input stream
     System.out.print("Enter the file name");
     BufferedReader keyRead = new BufferedReader(new
     InputStreamReader(System.in));
     String fname = keyRead.readLine();

      // sending the file name to server. Uses PrintWriter
     OutputStream  ostream = sock.getOutputStream( );
     PrintWriter pwrite = new PrintWriter(ostream, true);
     pwrite.println(fname);

      // receiving the contents from server.  Uses input stream

     System.out.println("Contents of the File:");
     InputStream istream = sock.getInputStream();
     BufferedReader socketRead = new BufferedReader(new
     InputStreamReader(istream));

     String str;
     while((str = socketRead.readLine())  !=  null)
     // reading line-by-line
     {
         System.out.println(str);
     }
     pwrite.close(); socketRead.close(); keyRead.close();
  }
}
```

Output:

Server:

```
Server ready for connection
Connection is successful and wating for chatting

Contents of file is sent...
```

Client:

```
Enter the file name:
test.txt

Contents of the file:
CANARA ENGINEERING COLLEGE

MANGALURU-575219
```

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

Server Program:

```
//DSender.java
import java.net.*;
import java.util.*;
public class DSender
{
     public static void main(String[] args) throws Exception
     {
           DatagramSocket ds = new DatagramSocket();
          Scanner s=new Scanner(System.in);
          System.out.println("Enter the Message and press ENTER
          to Send");
          String str = s.nextLine();
           InetAddress ip = InetAddress.getByName("127.0.0.1");

          DatagramPacket dp = new DatagramPacket(str.getBytes(),
          str.length(), ip, 21);
          ds.send(dp);
          ds.close();
     }
}
```

Client Program:

```
//DReceiver.java
import java.net.*;
public class DReceiver
{
      public static void main(String[] args) throws Exception
     {
           DatagramSocket ds = new DatagramSocket(21);
           byte[] buf = new byte[1024];
           DatagramPacket dp = new DatagramPacket(buf,
           1024);
          ds.receive(dp);
          String str = new String(dp.getData(), 0,
          dp.getLength());
          System.out.println("Message from Server:");
          System.out.println(str);
          ds.close();
     }
}
```

Output:

Server

```
Enter the Message and press ENTER to Send
Hello Canara
```

Client

```
Message from Server:
Hello Canara
```

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

```
import java.util.*;
import java.io.*;
class RSA
{

     static int mult(int x,  int y,  int n)
     {
          int k=1;
          int j;
           for (j=1; j<=y; j++) k = (k * x) % n;
          return ( int) k;
     }

     public static void main (String arg[])throws Exception
     {
          Scanner s=new Scanner(System.in);

          InputStreamReader r=new InputStreamReader(System.in);
          BufferedReader br=new BufferedReader(r);

          String msg1;
          int pt[]=new int[100];
          int ct[]=new int[100];
          int a,b, n, d, e,Z, p, q, i,temp,et;
          System.out.println("Enter prime No.s p,q :");
          p=s.nextInt();
          q=s.nextInt();
          n = p*q;
          Z=(p-1)*(q-1);

          System.out.println("\nSelect e value:");
          e=s.nextInt();

          System.out.printf("Enter message : ");
          msg1=br.readLine();
          char msg[]=msg1.toCharArray();

          for(i=0;i<msg.length;i++)
            pt[i]=msg[i];
          for(d=1;d<Z;++d)
               if(((e*d)%Z)==1) break;
                    System.out.println("p="+"
                    "+p+"\tq="+q+"\tn="+n+"\tz="+Z+"\te="+e+"\td
                    ="+d);
```

```
            System.out.println("\nCipher Text = ");
            for(i=0; i<msg.length; i++)
                  ct[i] = mult(pt[i], e,n);
             for(i=0; i<msg.length; i++)
                  System.out.print("\t"+ct[i]);
            System.out.println("\nPlain Text = ");
            for(i=0; i<msg.length; i++)
                 pt[i] = mult(ct[i], d,n) ;
            for(i=0; i<msg.length; i++)
            System.out.print((char)pt[i]);
      }
}
```

Output:

```
Enter prime No.s p,q :
13
23

Select e value:
19

Enter message :
CANARA ENGINEERING COLLEGE
Public Key:(19,299)
Private Key:(139,299)
Encrypting Message
Cipher Text:
892211322118622159691372239136969186239137259892748080697269

Decrypting Ciphertext
Plain Text:
CANARA ENGINEERING COLLEGE
```

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

```
import java.util.*;

class LB
{

public static void main(String arg[])
{
          int no_of_clk,storage,output_pkt_size;
          int input_pkt_size,bucket_size,size_left;

          //initial packets in the bucket
          storage=0;

          //total no. of times bucket content is checked
          Scanner s=new Scanner(System.in);
          System.out.println("Enter Bucket runtime:");

          no_of_clk=s.nextInt();

          //total no. of packets that can
          // be accomodated in the bucket
          System.out.println("Enter Bucket Size:");
          bucket_size=s.nextInt();

          //no. of packets that enters the bucket at a time
          Random randomGenerator = new Random();


          //no. of packets that exits the bucket at a time
          System.out.println("Enter Ouput Rate:");
          output_pkt_size=s.nextInt();
          for(int i=0;i<no_of_clk;i++)
          {
               System.out.println("-------------------------------");
               System.out.printf("At ClockTick:%d\n",i+1);
               System.out.println("-------------------------------");
               size_left=bucket_size-storage; //space left
               input_pkt_size=randomGenerator.nextInt(10);

               System.out.println("Incoming Burst
Size:"+input_pkt_size);

               if(input_pkt_size==0)
                    System.out.println("No incoming Flow");

               else if(input_pkt_size<=(size_left))
               {

                    storage+=input_pkt_size;
```

```
                        //System.out.println("Buffer size= "+storage+"
out of bucket size= "+bucket_size);
                  }
                  else
                  {
                        System.out.println("Bucket Overflow!!!No. of
Packets Dropped = "+(input_pkt_size-(size_left)));

                              //full size
                        storage=bucket_size;
                  }

            if(storage==0)
            System.out.println("Empty Bucket!!!Underflow");

            else if(storage<output_pkt_size)
            {
            System.out.println(storage+ " Packets sent out of Bucket");
            storage=0;
            }
            else{

           System.out.println("No. of Packets Sent out of the
Bucket="+output_pkt_size);

            storage-=output_pkt_size;
            System.out.println("Buffer size= "+storage+" used out of
bucket size= "+bucket_size);
            }
           System.out.printf("No. of Packets left in the
Bucket=%d",storage);
            System.out.println();
           System.out.println("----------------------------------------
");
            }
}
}
```

Output1:

```
Enter Bucket runtime:
5
Enter Bucket Size:
10
Enter Ouput Rate:
2
------------------------------
At ClockTick:1
------------------------------
Incoming Burst Size:7
No. of Packets Sent out of the Bucket=2
Buffer size= 5 used out of bucket size= 10
No. of Packets left in the Bucket=5
--------------------------------------
------------------------------
At ClockTick:2
------------------------------
Incoming Burst Size:8
Bucket Overflow!!!No. of Packets Dropped = 3
No. of Packets Sent out of the Bucket=2
Buffer size= 8 used out of bucket size= 10
No. of Packets left in the Bucket=8
--------------------------------------
------------------------------
At ClockTick:3
------------------------------
Incoming Burst Size:1
No. of Packets Sent out of the Bucket=2
Buffer size= 7 used out of bucket size= 10
No. of Packets left in the Bucket=7
--------------------------------------
------------------------------
At ClockTick:4
------------------------------
Incoming Burst Size:8
Bucket Overflow!!!No. of Packets Dropped = 5
No. of Packets Sent out of the Bucket=2
Buffer size= 8 used out of bucket size= 10
No. of Packets left in the Bucket=8
--------------------------------------
------------------------------
At ClockTick:5
------------------------------
Incoming Burst Size:4
Bucket Overflow!!!No. of Packets Dropped = 2
No. of Packets Sent out of the Bucket=2
Buffer size= 8 used out of bucket size= 10
No. of Packets left in the Bucket=8
--------------------------------------
```

Output2:

```
Enter Bucket runtime:
5
Enter Bucket Size:
10
Enter Ouput Rate:
8
-----------------------------
At ClockTick:1
-----------------------------
Incoming Burst Size:3
3 Packets sent out of Bucket
No. of Packets left in the Bucket=0
-------------------------------------
-----------------------------
At ClockTick:2
-----------------------------
Incoming Burst Size:1
1 Packets sent out of Bucket
No. of Packets left in the Bucket=0
-------------------------------------
-----------------------------
At ClockTick:3
-----------------------------
Incoming Burst Size:2
2 Packets sent out of Bucket
No. of Packets left in the Bucket=0
-------------------------------------
-----------------------------
At ClockTick:4
-----------------------------
Incoming Burst Size:6
6 Packets sent out of Bucket
No. of Packets left in the Bucket=0
-------------------------------------
-----------------------------
At ClockTick:5
-----------------------------
Incoming Burst Size:0
No incoming Flow
Empty Bucket!!!Underflow
No. of Packets left in the Bucket=0
-------------------------------------
```