4

```
import java.util.*;

public class CRC

{

        void div(int a[],int k)

        {

         int gp[]={1,0,0,0,1,0,0,0,0,0,0,1,0,0,0,0,1};

         int count=0;

         for(int i=0;i<k;i++)

         {

          if(a[i]==gp[0])

          {

           for(int j=i;j<17+i;j++)

           {

            a[j]=a[j]^gp[count++];

           }

           count=0;

          }

         }

        }


        public static void main(String[] args)

        {

         int a[]=new int[100];

         int b[]=new int[100];

         int len,k;

         CRC ob=new CRC();

         System.out.println("Enter the length of Data Frame:");

         Scanner sc=new Scanner(System.in);

         len=sc.nextInt();

         int flag=0;

         System.out.println("Enter the Message:");

         for(int i=0;i<len;i++)

         {

          a[i]=sc.nextInt();
```

```
}
for(int i=0;i<16;i++)
{
 a[len++]=0;
}
k=len-16;
for(int i=0;i<len;i++)
{
 b[i]=a[i];
}
ob.div(a,k);
for(int i=0;i<len;i++)
a[i]=a[i]^b[i];
System.out.println("Data to be transmitted: ");
for(int i=0;i<len;i++)
{
 System.out.print(a[i]+" ");
}
System.out.println();
System.out.println("Enter the Received Data: ");
for(int i=0;i<len;i++)
{
 a[i]=sc.nextInt();
}
ob.div(a, k);
for(int i=0;i<len;i++)
{
 if(a[i]!=0)
 {
  flag=1;
  break;
 }
}
if(flag==1)
 System.out.println("error in data");
```

```
                else
                 System.out.println("no error");
                }
               }
```

6

```
package adii;
import java.util.Scanner;
public class BellmanFord
{
   private int d[];
   private int nov;
   public static final int MAX_VALUE = 999;

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

   public void BellmanFordEvaluation(int s, int am[][])
   {
      for (int n = 1; n <= nov; n++)
      {
         d[n] = MAX_VALUE;
      }

      d[s] = 0;
      for (int n = 1; n <= nov - 1; n++)
      {
         for (int sn = 1; sn <= nov; sn++)
         {
            for (int dn = 1; dn <= nov; dn++)
 {
  if (am[sn][dn] != MAX_VALUE)
  {
```

```
if (d[dn] > d[sn] + am[sn][dn])
                    d[dn] = d[sn] + am[sn][dn];
          }
          }
       }
   }
   for (int sn = 1; sn <= nov; sn++)
   {
      for (int dn = 1; dn <= nov; dn++)
 {
if (am[sn][dn] != MAX_VALUE)
{
if(d[dn] > d[sn] + am[sn][dn])

               {
 System.out.println("The Graph contains negative egde cycle");
 break;
}
          }
       }
   }

   for (int v = 1; v <= nov; v++)
   {
      System.out.println("Distance of source  " + s + " to " + v + " is " + d[v]);
   }
}

 public static void main(String args[])
{
   int nov = 0;
   int s;
   Scanner scanner = new Scanner(System.in);

   System.out.println("Enter the number of vertices");
```

```
        nov = scanner.nextInt();

        int am[][] = new int[nov + 1][nov + 1];
        System.out.println("Enter the adjacency matrix");
        for (int sn = 1; sn <= nov; sn++)
        {
           for (int dn = 1; dn <= nov; dn++)
           {
              am[sn][dn] = scanner.nextInt();
              if (sn == dn)
              {
              am[sn][dn] = 0;
              continue;
              }
              if (am[sn][dn] == 0)
              {
              am[sn][dn] = MAX_VALUE;
              }
              }
              }
              System.out.println("Enter the source vertex");
              s = scanner.nextInt();
              BellmanFord bellmanford = new BellmanFord(nov);
              bellmanford.BellmanFordEvaluation(s, am);
              scanner.close();
              }
              }
```

8

```
package adiii;

import java.net.*;

import java.util.Scanner;
```

```
public class UPDCient {
		public static void main(String[] args)
		{
		 DatagramSocket skt;
		 Scanner scan = new Scanner(System.in);
		 try
		 {
		 System.out.println("Enter Message:");
		 String msg= scan.next();
		 skt=new DatagramSocket();

		 byte[] b = msg.getBytes();
		 InetAddress host=InetAddress.getByName("127.0.0.1");
		 int serverPort=6788;
		 DatagramPacket request =new DatagramPacket (b,b.length,host,serverPort);
		 skt.send(request);
		 byte[] buffer =new byte[1000];
		 DatagramPacket reply= new DatagramPacket(buffer,buffer.length);
		 skt.receive(reply);
		 String s1 = new String(reply.getData());
		 System.out.println("Client received: " + s1.trim());
		 skt.close();
		}
		catch(Exception ex)
		{
		}
		}
		}


package adiii;
import java.net.*;
public class UDPServer {
		public static void main(String[] args)
		 {
```

```
                DatagramSocket skt=null;
                try
                {
                 skt=new DatagramSocket(6788);
                 byte[] buffer = new byte[1000];
                 System.out.println("Listening on port 6788");
                 while(true)
                 {
                  DatagramPacket request = new DatagramPacket(buffer,buffer.length);
                  skt.receive(request);
                  String message = new String(request.getData());
                  System.out.println("server received request ");
                  String toUpper = message.toUpperCase();
                  byte[] sendMsg= toUpper.getBytes();
                  System.out.println("server sending response ");
                  DatagramPacket reply = new DatagramPacket(sendMsg,sendMsg.length,
                  request.getAddress(),request.getPort());
                  skt.send(reply);
                 }
                }
                catch(Exception ex)
                {
                }
               }
}
```

10

```
package adiiiii;

import java.util.Scanner;
public class LeakyBucket {
        public static void main(String args[])
        {
         int n, outgoing, store, bucketSize;
         int incoming[];
```

```
Scanner scan = new Scanner(System.in);
System.out.println("Enter number of inputs");
n = scan.nextInt();
incoming = new int[n];
for(int i = 0 ;i< n ; i++)
{
 System.out.println("Enter incoming packet size "+(i+1));
 incoming[i] = scan.nextInt();
}
System.out.println("Enter bucket size");
bucketSize = scan.nextInt();
System.out.println("Enter outgoing rate");
outgoing = scan.nextInt();
store = 0;
int i = 0;
System.out.println("Packet Recieved | Packet Dropped | Packet Sent | Packet Left");

do
{
                int pktReceived = 0, pktSent = 0,pktDrop = 0;
                 if(i < n)
                 {
                  pktReceived = incoming[i];

                  if(pktReceived  <= (bucketSize - store))
                  {
                   store += pktReceived ;
                  }
                  else
                  {

                   pktDrop = pktReceived -(bucketSize - store);
                   store = bucketSize;
                  }
                 }
```

```
                if(store > outgoing)
                {
                 store -= outgoing;
                 pktSent = outgoing;
                }
                else
                {
                 pktSent = store;
                 store = 0;
                }
                System.out.println(pktReceived +"\t\t"+pktDrop+"\t\t"+pktSent+"\t\t"+store);
                try
                {
                                 Thread.sleep(2000);
                                }
                                catch(Exception e)
                                {
                                }
                                i++;

                               }while(store != 0 || i < n);

                              }

}
```