```
//Write a Program for Error Detecting Code using CRC-CCITT (16 bit).
import java.util.Scanner;
import java.io.*;
public class CRC1 {
   public static void main(String args[]) {
   Scanner sc = new Scanner(System.in);
   System.out.print("Enter message bits: ");
   String message = sc.nextLine();
   System.out.print("Enter generator: ");
   String generator = sc.nextLine();
int data[] = new int[message.length() + generator.length() - 1];
int divisor[] = new int[generator.length()];
for(int i=0;i<message.length();i++)
            data[i] = Integer.parseInt(message.charAt(i)+"");
for(int i=0;i<generator.length();i++)
            divisor[i] = Integer.parseInt(generator.charAt(i)+"");
for(int i=0;i<message.length();i++)   {
if(data[i]==1)
for(int j=0;j<divisor.length;j++)
data[i+j] ^= divisor[j];      }
System.out.print("The checksum code is: ");
for(int i=0;i<message.length();i++)
data[i] = Integer.parseInt(message.charAt(i)+"");
for(int i=0;i<data.length;i++)
  System.out.print(data[i]);
System.out.println();
System.out.print("Enter checksum code: ");
message = sc.nextLine();
System.out.print("Enter generator: ");
generator = sc.nextLine();
data = new int[message.length() + generator.length() - 1];
```

```
divisor = new int[generator.length()];
for(int i=0;i<message.length();i++)
data[i] = Integer.parseInt(message.charAt(i)+"");
for(int i=0;i<generator.length();i++)
divisor[i] = Integer.parseInt(generator.charAt(i)+"");
for(int i=0;i<message.length();i++) {
if(data[i]==1)
for(int j=0;j<divisor.length;j++)
data[i+j] ^= divisor[j];
boolean valid = true;
for(int i=0;i<data.length;i++)
if(data[i]==1){
valid = false;
break;   }
if(valid==true)
System.out.println("Data stream is valid");
else
System.out.println("Data stream is invalid. CRC error occurred.");
} }
```

```
// Write a program to find the shortest path between vertices using Bellman Ford Algorithm.
import java.util.Scanner;
 public class ford   {
   private int D[];
   private int num_ver;
   public static final int MAX_VALUE = 999;
    public ford(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();
      ford b = new ford (num_ver);
       b.BellmanFordEvaluation(source, A);
       scanner.close(); }}
```

```
 //Write a program for simple RSA algorithm to encrypt and decrypt the data.
import java.util.Scanner;
public class Rsa {
public 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 k; }
public static int gcd(int m,int n) {
if(n==0)
return m;
else
return (gcd(n,m%n)); }
public static int isprime(int num)  {
 int temp;
 boolean isprime=true;
 for(int k=2;k<=num/2;k++)  {
 temp=num%k;
 if(temp==0)  {
 isprime=false;
 break;  }}
 if(isprime==false)  {
```

```
 System.out.println(num + "not a prime number");
return 0;  }
else  {
System.out.println(num + "is a prime number");
return num;   }}
public static void main(String[] args)  {
int msg,plaintext,ciphertext;
int n,d=0,e,z,p,q,i;
Scanner scanner = new Scanner(System.in);
System.out.println("enter two values p and q: ");
 p = scanner.nextInt();
 q = scanner.nextInt();
 int a=isprime(p);
int b=isprime(q);
 if((a==p && b==q) && (a!=0 && b!=0) )
{
System.out.println("enter message ");
msg=scanner.nextInt();
n=p*q;
z=(p-1)*(q-1);
do   {
System.out.print("choose the value of e such that gcd(z,e)=1: ");
e=scanner.nextInt();
} while (gcd(z,e)!=1);
i=2;
while (((i*e)%z)!=1) {
i++;
d=i;  }
System.out.println("The public key pair is (" + e + "," + n + ")");
System.out.println("The private key pair is (" + d + "," + n + ")");
ciphertext = mult(msg,e,n);
```

```
System.out.println("cipher text = " + ciphertext);
plaintext = mult(ciphertext,d,n);
System.out.println("plain text = " + plaintext); }}}
```

Develop a java program to implement sliding window protocol in data link layer.

```
import java.util.Scanner;
public class SlidingWindowProtocol {
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);
      System.out.print("Enter the total number of frames: ");
      int totalFrames = scanner.nextInt();
      System.out.print("Enter the window size: ");
      int windowSize = scanner.nextInt();
      int sentFrames = 0;
      int ackFrame;
      while (sentFrames < totalFrames) {
         for (int i = 0; i < windowSize && sentFrames < totalFrames; i++) {
            System.out.println("Frame " + sentFrames + " sent.");
            sentFrames++;   }
         System.out.print("Enter the last acknowledged frame: ");
         ackFrame = scanner.nextInt();
         if (ackFrame < sentFrames) {
            sentFrames = ackFrame + 1;  }
else {
  System.out.println("Invalid acknowledgment. Try again.");
     }  }
      System.out.println("All frames sent successfully!");
      scanner.close(); }  }
```

```
//Write a program for congestion control using leaky bucket algorithm
import java.util.Scanner;
import java.lang.*;
```

```
public class lab7 {
public static void main(String[] args) {
int i;
int a[]=new int[20];
int buck_rem=0,buck_cap=4,rate=3,sent,recv;
Scanner in = new Scanner(System.in);
System.out.println("Enter the number of packets");
int n = in.nextInt();
System.out.println("Enter the packets");
for(i=1;i<=n;i++)
a[i]= in.nextInt();
System.out.println("Clock \t packet size \t accept \t sent \t remaining");
for(i=1;i<=n;i++)  {
if(a[i]!=0) {
if(buck_rem+a[i]>buck_cap)
recv=-1;
else {
recv=a[i];
buck_rem+=a[i];  } }
else
recv=0;
if(buck_rem!=0)  {
if(buck_rem<rate)
{sent=buck_rem;
buck_rem=0;  }
else  {
sent=rate;
buck_rem=buck_rem-rate; }}
else
sent=0;
if(recv==-1)
```

```
System.out.println(+i+ "\t\t" +a[i]+ "\t dropped \t" +  sent +"\t" +buck_rem);
else
System.out.println(+i+ "\t\t" +a[i] +"\t\t" +recv +"\t" +sent + "\t" +buck_rem);
}}}
```

_____-