1)Ceaser Cipher

**Code:**

import java.util.\*;

class Cipher

{

static String encryption;

static String decryption;

static char ch;

public static String encrypt(String str)

{

encryption="";

System.out.println("\n\t\t\*\*\*\*\*\*\*\*\*\*\*\*\*Encryption\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

for(int i=0;i<str.length();i++)

{

if(str.charAt(i)=='x')

{

encryption+='a';

}

else if(str.charAt(i)=='y')

{

encryption+='b';

}

else if(str.charAt(i)=='z')

{

encryption+='c';

}

else

{

ch=str.charAt(i);

encryption+=(char)(ch+3);

}

}

return encryption;

}

public static String decrypt(String str)

{

decryption="";

System.out.println("\n\t\t\*\*\*\*\*\*\*\*\*\*\*\*\*Decryption\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

for(int i=0;i<str.length();i++)

{

if(str.charAt(i)=='x')

{

decryption+='a';

}

else if(str.charAt(i)=='y')

{

decryption+='b';

}

else if(str.charAt(i)=='z')

{

decryption+='c';

}

else

{

ch=str.charAt(i);

decryption+=(char)(ch-3);

}

}

return decryption;

}

public static void main(String[] args)

{

String en;

Scanner sc=new Scanner(System.in);

System.out.println("\n\t\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*The Cesar Cipher Substitution Algo\*\*\*\*\*\*\*\*\n\n");

String str= sc.next();

en=encrypt(str);

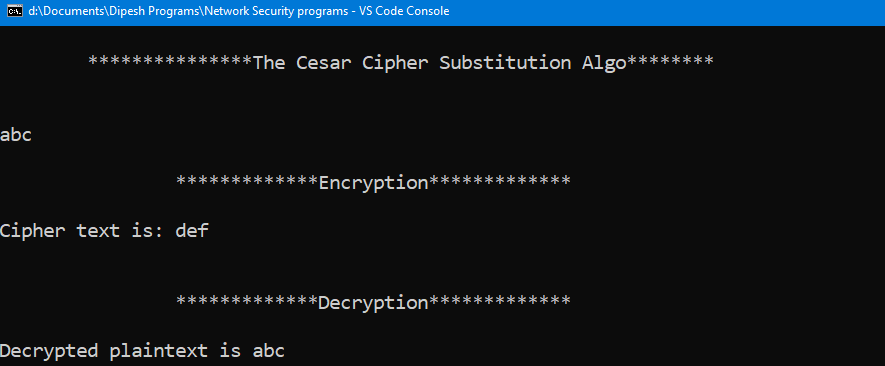
System.out.println("Cipher text is: "+en+"\n");

System.out.println("Decrypted plaintext is "+decrypt(en));

}

}

**Output:**



2) Rail fence cipher

**Code:**

import java.util.\*;

public class railfencecipher

{

railfencecipher()

{

try

{

String msg="";

String str1="";

String str2="";

Scanner sc=new Scanner(System.in);

System.out.println("Enter message:");

msg=sc.next();

for(int i=0;i<msg.length();i++)

{

if(i%2==0)

{

str1+=msg.charAt(i);

}

else

{

str2+=msg.charAt(i);

}

}

System.out.println("\nEncrypted Message:"+(str1+str2));

}

catch(Exception e)

{

System.out.println(e);

}

}

public static void main(String[] args)

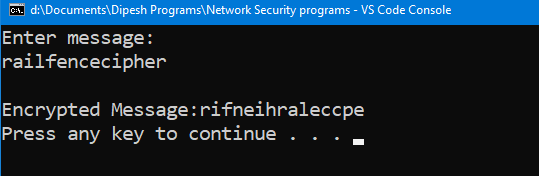
{

new railfencecipher();

}

}

**Output:**



3) Monoalphabetic cipher

**Code:**

import java.util.\*;

public class monoalphabetic

{

public static char p[]={'a','b','c','d','e','f','g','h','i','j','k','l','m','n','o','p','q','r','s','t','u','v','x','y','z'};

public static char ch[]={'Q','W','E','R','T','Y','U','I','O','P','A','S','D','F','G','H','J','K','L','Z','X','C','V','B','N','M'};

public void Encryption(String msg){

String encrypt="";

for(int i=0; i<msg.length();i++)

{

for(int j=0;j<p.length;j++)

{

if(msg.charAt(i)==p[j])

{

encrypt= encrypt+ch[j];

}

}

}

System.out.println("\nThe encrypted message is :\n" +encrypt);

Decryption(encrypt);

}

public void Decryption(String msg)

{

String decrypt="";

for(int i=0; i<msg.length();i++)

{

for(int j=0;j<ch.length;j++)

{

if(msg.charAt(i)==ch[j])

{

decrypt=decrypt + p[j];

}

}

}

System.out.println("\nThe decrypted message is : \n"+decrypt);

}

public static void main(String[] args)

{

String msg;

monoalphabetic obj1=new monoalphabetic();

Scanner sc=new Scanner(System.in);

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

try

{

msg=sc.next();

obj1.Encryption(msg);

}

catch(Exception e)

{

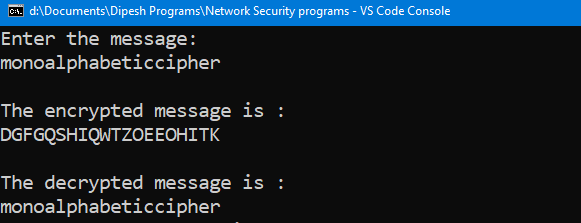
System.out.println(e);

}

}

}

**Output:**

****

4)Vernam Cipher

**Code:**

import java.util.\*;

public class VernamCipher

{

VernamCipher()

{

try

{

char[] alpha=new char[26];

for(int i=0;i<alpha.length;i++)

{

alpha[i]=(char)('A'+i);

}

String msg;

String key;

Scanner sc=new Scanner(System.in);

System.out.println("Enter message:");

msg=sc.next();

System.out.println("Enter Key:");

key=sc.next();

int sum=0,num1=0,num2=0;

System.out.println("Encrypted Text:");

for(int i=0;i<msg.length();i++)

{

char pt=msg.charAt(i);

char k=key.charAt(i);

for(int j=0; j<alpha.length; j++)

{

if(pt==alpha[j])

{

num1=j;

}

if(k==alpha[j])

{

num2=j;

}

sum=num1+num2;

if(sum>25)

{

sum=sum%26;

}

}

System.out.println(alpha[sum]);

}

}

catch(Exception e)

{

System.out.println(e);

}

}

public static void main(String[] args)

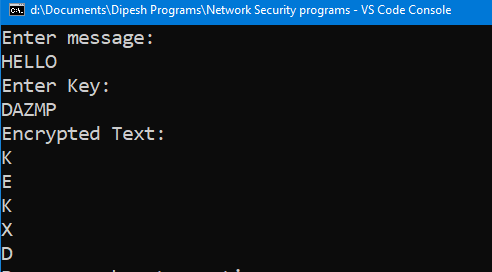
{

new VernamCipher();

}

}

**Output:**



5)Columnar Transposition Cipher

**Code:**

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

class ColumnarCipher

{

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

{

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

System.out.println("Enter message:");

String accept=br.readLine();

System.out.println("Enter number of rows:");

int rows=Integer.parseInt(br.readLine());

System.out.println("Enter number of columns:");

int col=Integer.parseInt(br.readLine());

int count=0;

char cont[][];

cont = new char[rows][col];

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

{

for(int j=0;j<col;j++)

{

if(count>=accept.length())

{

cont[i][j]=' ';

count++;

}

else

{

cont[i][j]=accept.charAt(count);

count++;

}

}

}

System.out.println("Enter the order of choices:");

int choice[]=new int[col];

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

{

System.out.println("choice"+k+"-->");

choice[k]=Integer.parseInt(br.readLine());

}

System.out.println("Cipher text in matrix is -->");

String cipher="";

for (int j=0;j<col;j++)

{

int k=choice[j];

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

{

System.out.println(i);

System.out.println(k);

cipher += cont[i][k];

}

cipher = cipher.trim();

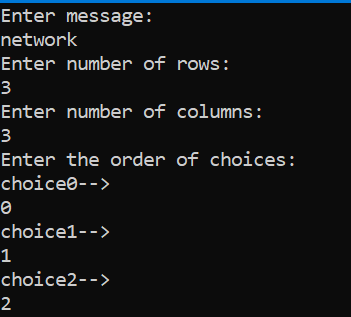
System.out.println(cipher);

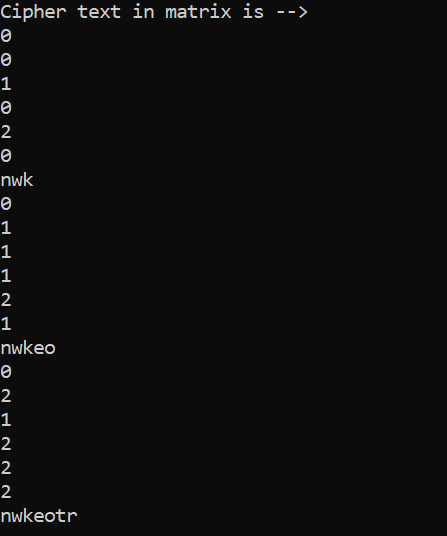
}

}

}

**Output:**

****

****

6)Diffie-Hellman Algorithm

**Code:**

import java.io.\*;

import java.math.BigInteger;

class DiffieHellman

{

public static void main(String args[])

{

try

{

BigInteger n,g,x,y,A,B,k1,k2;

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

//Accept values of n and g

System.out.println("\nEnter two large Prime Numbers:");

System.out.println("Enter value of n:");

n=new BigInteger(br.readLine());

chkPrime(n);

System.out.println("Enter value of g:");

g=new BigInteger(br.readLine());

chkPrime(g);

//Accept values for x and y

System.out.println("Enter the value of x:");

x=new BigInteger(br.readLine());

System.out.println("Enter value of y:");

y=new BigInteger(br.readLine());

//calculate A

A=g.modPow(x,n);

//calculate B

B=g.modPow(y,n);

//calculate key k1 and k2

k1=B.modPow(x,n);

k2=A.modPow(y,n);

if(k1.equals(k2))

{

System.out.println("key 1 is: "+k1+" key 2 is:"+k2);

System.out.println("Both keys are equalll hence Successfull!!!!");

}

}

catch(Exception e)

{

System.out.println(e);

}

}

public static void chkPrime(BigInteger n)

{

try

{

if(n.isProbablePrime(10))

{

System.out.println("Prime, Continue");

}

else

{

System.out.println("Not prime");

System.out.println("Enter values:");

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

BigInteger x=new BigInteger(br.readLine());

chkPrime(x);

}

}

catch(Exception e)

{

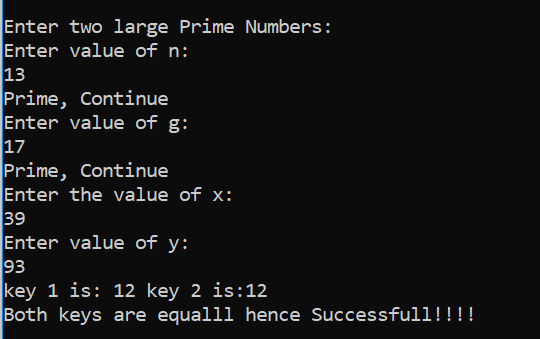
System.out.println(e);

}

}

}

**Output:**

****

7)RSA

**Code:**

import java.security.\*;

import java.math.\*;

public class RSA {

public static void main(String[] args)

{

SecureRandom r;

BigInteger p,q,p1,q1,n,n1,e,d,msg,ct,pt;

int bitLength = 512;

int certainity = 100;

r = new SecureRandom();

// Step 1 : Generate prime no. p and q

p = new BigInteger(bitLength,certainity,r);

q = new BigInteger(bitLength,certainity,r);

// Step 2 n=p\*q

n = p.multiply(q);

System.out.println("Prime no P is: "+p.intValue());

System.out.println("Prime no Q is: "+q.intValue());

System.out.println("n = p\*q is : "+n.intValue());

// Step 3 : generating public key(E)

p1 = p.subtract(new BigInteger("1"));

q1 = q.subtract(new BigInteger("1"));

n1 = p1.multiply(q1);

e = new BigInteger("2");

while(n1.gcd(e).intValue()>1 || e.compareTo(p1)!=-1)

{

e = e.add(new BigInteger("1"));

}

System.out.println("Public Key is("+n.intValue()+","+e.intValue()+")");

//Step 4: D = E^-1 mod(P-1)(Q-1)

d = e.modInverse(n1);

System.out.println("Private Key is("+n.intValue()+","+d.intValue()+")");

// Step 5: Encryption CT = (PT)^e mod n

msg = new BigInteger("5");

ct = msg.modPow(e, n);

System.out.println("Encrypted Text is: "+ct.intValue());

pt = ct.modPow(d, n);

System.out.println("Decrypted Text is: "+pt.intValue());

}

}

**Output:**

