insssssssssssssssssss

caesar

import java.util.Scanner;

public class JavaApplication80{
public static void main(String[] args) {
Scanner sc = new Scanner(System.in);
System.out.println("enter :");
String plaintext = sc.nextLine();
System.out.println("key :");
int shift = sc.nextInt();
String encrypt = caesar(plaintext,shift);
System.out.println("encrypt message :"+encrypt);
String decrypt = caesar(encrypt,26-shift);
System.out.println("decrypt :"+decrypt);

```
}
public static String caesar(String message,int shift){
    StringBuilder result = new StringBuilder();
    for(char character : message.toCharArray()){
        if(Character.isLetter(character)){
            char base = Character.isLowerCase(character)? 'a' :'A';
            character = (char)(base + (character - base + shift)%26);
            result.append(character);

        }
    }
    return result.toString();
}
```

}

monoalphabetic :

import java.util.Scanner;

public class MonoalphabeticCipher {
public static void main(String[] args) {
Scanner scanner = new Scanner(System.in);
System.out.print("Enter a message: ");

String message = scanner.nextLine();
String key = "xplmoknijbqhdwfgvuasrtzecy";
String encryptedMessage = encrypt(message, key);
System.out.println("Encrypted Message: " + encryptedMessage);
String decryptedMessage = decrypt(encryptedMessage, key);
System.out.println("Decrypted Message: " + decryptedMessage);
scanner.close();
}

```
public static String encrypt(String message, String key) {
    StringBuilder result = new StringBuilder();
    for (char c : message.toCharArray()) {
        if (Character.isLetter(c)) {
            char base = Character.isUpperCase(c) ? 'A' : 'a';
            result.append(key.charAt(c - base));
        } else {
            result.append(c);
        }
    }
    return result.toString();
}

public static String decrypt(String message, String key) {
    StringBuilder result = new StringBuilder();
    for (char c : message.toCharArray()) {
        if (Character.isLetter(c)) {
            char base = Character.isUpperCase(c) ? 'A' : 'a';
            int index = key.indexOf(c);
            result.append((char) (base + index));
        } else {
            result.append(c);
        }
    }
    return result.toString();
}
```

}

Scanner sc = new Scanner(System.in);
System.out.println("enter the plaintext :");
String message = sc.nextLine();
String key = "fdhgddryhdgbdgtryhgyhhf";
String encrypted = encrypt(message,key);
String decrypted = decrypt(encrypted,key);

```
}
public static String encrypt(String message, String key){
    StringBuilder result  =  new StringBuilder();
    for(char character :message.toCharArray()){
        if(Character.isLetter(character)){
            char base = Character.isUpperCase(character)? 'A' : 'a';
            int index  = character - base ;
            result.append(key.charAt(base));
        }

    }
    return result.toString();
}
    public static String decrypt(String message , String key){
        StringBuilder result = new StringBuilder();
        for(char character :message.toCharArray()){
        if(Character.isLetter(character)){
            char base = Character.isUpperCase(character)? 'A' : 'a';
            int index = key.indexOf(character);
        }
        else {
            result.append((char)(base + index));
        }


         return result.toString();

    }
```

practical 2 : vernam :

public class JavaApplication81 {

```
public static void main(String[] args) {
   String plaintext = "hello";
   String key ="yspme";
   String encryptedtext = encrypt(plaintext,key);
   String decryptedtext = decrypt(encryptedtext,key);
    System.out.println("cipher text :"+encryptedtext);
    System.out.println("message :"+decryptedtext);
```

}
public static String encrypt(String text,String key){
StringBuilder res = new StringBuilder();
for(int i =0 ; i< text.length();i++){
int x= (text.charAt(i)- 'a' + key.charAt(i%key.length())-'a') %26 + 'a';
res.append((char)x);

```
}
 return res.toString();
}
public static String decrypt(String dec, String key){
    StringBuilder reso = new  StringBuilder();
    for(int i =0 ; i< dec.length();i++){
        int x= (dec.charAt(i)- 'a' - (key.charAt(i%key.length())-'a') +26) %26 + 'a';
        reso.append((char)x);
}
return reso.toString();
```

}
}

aes and des

Cipher ecipher;

Cipher decipher;

JavaApplication81(SecretKey key)throws Exception
{

ecipher = Cipher.getInstance("AES");

decipher = Cipher.getInstance("AES");

ecipher.init(Cipher.ENCRYPT_MODE, key);

decipher.init(Cipher.DECRYPT_MODE, key);

}
public String encrypt(String str)throws Exception
{

byte[] utf8 = str.getBytes("utf8");
byte[] enc = ecipher.doFinal(utf8);
return Base64.getEncoder().encodeToString(enc);
}
public String decrypt(String str) throws Exception
{
byte[] dec = Base64.getDecoder().decode(str);
byte[] utf8 = decipher.doFinal(dec);
return new String(utf8,"UTF8");
}

public static void main(String args[]) throws Exception
{
SecretKey key = KeyGenerator.getInstance("AES").generateKey();
JavaApplication84 encrypter = new JavaApplication84(key);
String secrettext = "Tommorrow is Tuesday";
String encrypted = encrypter.encrypt(secrettext);
String decrypted = encrypter.decrypt(encrypted);
System.out.println("Original Text: " + secrettext);
System.out.println("Encrypted Text: " + encrypted);
System.out.println("Decrypted text: " + decrypted);
}
}

diffie hellman

package javaapplication81;

import java.util.Scanner;

public class JavaApplication81 {
public static void main(String[] args) {
Scanner sc = new Scanner(System.in);

```
    System.out.println("Enter values for public keys Q and P:");
    long Q = sc.nextLong();
    long P = sc.nextLong();

    System.out.println("Enter private key a selected by User1:");
    long a = sc.nextLong();
    System.out.println("Enter private key b selected by User2:");
    long b = sc.nextLong();

    long x = calculatePower(Q, a, P);
    long y = calculatePower(Q, b, P);

    long ka = calculatePower(y, a, P);
    long kb = calculatePower(x, b, P);

    System.out.println("Secret key for User1 is: " + ka);
    System.out.println("Secret key for User2 is: " + kb);
}

private static long calculatePower(long x, long y, long P) {
```

```
    return (long) Math.pow(x, y) % P;
}
```

}

rsa

/*

- Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
- Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Main.java to edit this template
  */
  package javaapplication81;

import java.util.Scanner;

import java.math.BigInteger;

class JavaApplication81 {

```
public static void main(String[] args)
{
    int p=13, q = 11, e, d;
    BigInteger n, c, msg,z,de;
    n = BigInteger.valueOf(p * q);
    z = BigInteger.valueOf((p - 1) * (q - 1));
    for (e = 2; z.gcd(BigInteger.valueOf(e)).intValue() > 1; e++) ;
    d = BigInteger.valueOf(e).modInverse(z).intValue();
    msg = BigInteger.valueOf(76);
    c = msg.modPow(BigInteger.valueOf(e), n);
    System.out.println("Encrypted message: " + c);
    de = c.modPow(BigInteger.valueOf(d), n);
    System.out.println("Decrypted message: " + de);
}
```

}

rail fence

public class JavaApplication82 {
public static String encrypt(String plaintext, int rails)
{
StringBuilder[] fence = new StringBuilder[rails];

for (int i = 0; i < rails; i++)
{
fence[i] = new StringBuilder();
}

```
    int rail = 0;
    boolean down = true;

    for (char c : plaintext.toCharArray())
    {
        fence[rail].append(c);
        if (rail == 0)
        {
            down = true;
        }
        else if (rail == rails - 1)
        {
            down = false;
        }
        rail += down ? 1 : -1;
    }

    StringBuilder ciphertext = new StringBuilder();
    for (StringBuilder railString : fence)
    {
        ciphertext.append(railString.toString());
    }

    return ciphertext.toString();
}

public static String decrypt(String ciphertext, int rails) {
    StringBuilder[] fence = new StringBuilder[rails];
    for (int i = 0; i < rails; i++)
    {
        fence[i] = new StringBuilder();
    }

    int rail = 0;
    boolean down = true;

    for (int i = 0; i < ciphertext.length(); i++)
    {
        fence[rail].append(' '); // Fill the fence with spaces
        if (rail == 0)
        {
            down = true;
        }
        else if (rail == rails - 1)
        {
            down = false;
        }
        rail += down ? 1 : -1;
```

```
    }

    int index = 0;
    for (int i = 0; i < rails; i++)
    {
        for (int j = 0; j < fence[i].length(); j++)
        {
            fence[i].setCharAt(j, ciphertext.charAt(index++));
        }
    }

    rail = 0;
    down = true;
    StringBuilder plaintext = new StringBuilder();

    for (int i = 0; i < ciphertext.length(); i++)
    {
        plaintext.append(fence[rail].charAt(0));
        fence[rail].deleteCharAt(0);
        if (rail == 0)
        {
            down = true;
        } else if (rail == rails - 1) {
            down = false;
        }
        rail += down ? 1 : -1;
    }

    return plaintext.toString();
}

public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    System.out.println("enter the plain text :");
    String plaintext = sc.nextLine();
    int rails = 5;

    // Encryption
    String encrypted = encrypt(plaintext, rails);
    System.out.println("Encrypted: " + encrypted);

    // Decryption
    String decrypted = decrypt(encrypted, rails);
    System.out.println("Decrypted: " + decrypted);
}
```

}

MD5

```
import java.math.;
import java.security.;
public class JavaApplication82
{
public static String getMd5(String input)throws NoSuchAlgorithmException
{
MessageDigest md = MessageDigest.getInstance("MD5");
byte[] messageDigest = md.digest(input.getBytes());
BigInteger no = new BigInteger(1, messageDigest);
String hashtext = no.toString(18);
while (hashtext.length() < 32)
{
hashtext = "0" + hashtext;
}
return hashtext;
}
public static void main(String args[]) throws NoSuchAlgorithmException
{
String s = " Today is Tomorrow and i m rajat ";
System.out.println("Your HashCode Generated by MD5 is: " + getMd5(s));
}
}
```

SHA1 :

```
import java.math.;
import java.security.;
public class JavaApplication82
{
public static String encryptThisString(String input)
{
try {
    MessageDigest md = MessageDigest.getInstance("SHA-1");
    byte[] messageDigest = md.digest(input.getBytes());
    BigInteger no = new BigInteger(1, messageDigest);
    String hashtext = no.toString(16);
    while (hashtext.length() < 32) {
      hashtext = "0" + hashtext;
```

```
    }
    return hashtext;
  }
  catch (NoSuchAlgorithmException e) {
    throw new RuntimeException(e);
  }
}
public static void main(String args[]) throws NoSuchAlgorithmException
{
  System.out.println("HashCode Generated by SHA-1 for: ");
  String s1 = "Today is Tommorrow";
  System.out.println("\\n" + s1 + " : " + encryptThisString(s1));

}
```

}