Product cipher

import java.util.\*;

public class ProductCipher

{

public static void main(String args[]) {

int sub\_key, len, i, x=0, y=0, j=0, z=0;

String pt, cipher;

try (Scanner sc = new Scanner(System.in)) {

System.out.println("\nEnter Plain text message");

pt = sc.nextLine();

pt = pt.toUpperCase();

len = pt.length();

char[] c1 = new char[len];

char[] ct = new char[len];

char[] rfc = new char[len];

c1 = pt.toCharArray();

System.out.println("\nEnter Key value ");

sub\_key = sc.nextInt();

System.out.println("...........ENCRYPTION............ \nPlain Text:- "+pt);

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

{

x = (c1[i]+sub\_key);

if(x>90) {

y = x-90;

ct[i] = (char)(65+(y-1)); }

else

ct[i] = (char)(x);

}

for(i=0;i<len;i=i+2)

{

rfc[j]=ct[i];

j++;

}

for(i=1;i<len;i=i+2)

{

rfc[j]=ct[i];

j++;

}

cipher = new String(rfc);

System.out.println("Cipher Text:- "+cipher);

System.out.println(".........DECRYPTION........\n CIPHER TEXT:- "+cipher);

len = cipher.length();

if(len%2!=0) {

z = (len/2);

z = z+1;}

else

z = (len/2);

rfc = cipher.toCharArray();

j=z;

for(i=0, x=0;i<z && j<len&& x<len;i++,j++) {

ct[x] = rfc[i];

x++;

ct[x] = rfc[j];

x++;

}

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

{

x = (ct[i]-sub\_key);

if(x<65) {

y = 65-x;

rfc[i] = (char)(90-(y-1)); }

else

rfc[i] = (char)(x);

}

cipher = new String(rfc);

}

System.out.println("PLAIN TEXT:- "+cipher);

}

}

RSA

// Java Program to Implement the RSA Algorithm  
import java.math.\*;  
import java.util.\*;  
   
class RSA {  
    public static void main(String args[])  
    {  
        int p, q, n, z, d = 0, e, i;  
   
        // The number to be encrypted and decrypted  
        int msg = 12;  
        double c;  
        BigInteger msgback;  
   
        // 1st prime number p  
        p = 3;  
   
        // 2nd prime number q  
        q = 11;  
        n = p \* q;  
        z = (p - 1) \* (q - 1);  
        System.out.println("the value of z = " + z);  
   
        for (e = 2; e < z; e++) {  
   
            // e is for public key exponent  
            if (gcd(e, z) == 1) {  
                break;  
            }  
        }  
        System.out.println("the value of e = " + e);  
        for (i = 0; i <= 9; i++) {  
            int x = 1 + (i \* z);  
   
            // d is for private key exponent  
            if (x % e == 0) {  
                d = x / e;  
                break;  
            }  
        }  
        System.out.println("the value of d = " + d);  
        c = (Math.pow(msg, e)) % n;  
        System.out.println("Encrypted message is : " + c);  
   
        // converting int value of n to BigInteger  
        BigInteger N = BigInteger.valueOf(n);  
   
        // converting float value of c to BigInteger  
        BigInteger C = BigDecimal.valueOf(c).toBigInteger();  
        msgback = (C.pow(d)).mod(N);  
        System.out.println("Decrypted message is : "  
                           + msgback);  
    }  
   
    static int gcd(int e, int z)  
    {  
        if (e == 0)  
            return z;  
        else  
            return gcd(z % e, e);  
    }  
}

Vigenère Cipher

class Main

{

static String generateKey(String str, String key)

{

int x = str.length();

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

{

if (x == i)

i = 0;

if (key.length() == str.length())

break;

key+=(key.charAt(i));

}

return key;

}

static String cipherText(String str, String key)

{

String cipher\_text="";

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

{

int x = (str.charAt(i) + key.charAt(i)) %26;

x += 'A';

cipher\_text+=(char)(x);

}

return cipher\_text;

}

static String originalText(String cipher\_text, String key)

{

String orig\_text="";

for (int i = 0 ; i < cipher\_text.length() &&

i < key.length(); i++)

{

// converting in range 0-25

int x = (cipher\_text.charAt(i) -

key.charAt(i) + 26) %26;

// convert into alphabets(ASCII)

x += 'A';

orig\_text+=(char)(x);

}

return orig\_text;

}

static String LowerToUpper(String s)

{

StringBuffer str =new StringBuffer(s);

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

{

if(Character.isLowerCase(s.charAt(i)))

{

str.setCharAt(i, Character.toUpperCase(s.charAt(i)));

}

}

s = str.toString();

return s;

}

public static void main(String[] args)

{

String Str = "save the king from tiger";

String Keyword = "attack";

String str = LowerToUpper(Str);

String keyword = LowerToUpper(Keyword);

String key = generateKey(str, keyword);

String cipher\_text = cipherText(str, key);

System.out.println("Ciphertext : "

+ cipher\_text + "\n");

System.out.println("Original/Decrypted Text : "

+ originalText(cipher\_text, key));

}

}

MessageDigestExample

// import required classes and package if any

import java.security.MessageDigest;

import java.util.Scanner;

// create class MessageDigestExample to understand the use of MessageDigest class

public class MessageDigestExample {

// main() method start

public static void main(String args[]) throws Exception{

// create an instance of Scanner class

Scanner sc = new Scanner(System.in);

System.out.println("Enter the message of any arbitrary length:");

String msg = sc.nextLine();

// close Scanner class

sc.close();

//create an instance of the MessageDigest by using the getInstance() method with the MD5 algorithm

MessageDigest obj = MessageDigest.getInstance("MD5");

//use update() method for passing data to the created MessageDigest Object

obj.update(msg.getBytes());

//use the digest() method for computing the message digest

byte[] byteArray = obj.digest();

System.out.println(byteArray);

//convert the byte array in to Hex String format

StringBuffer hexData = new StringBuffer();

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

hexData.append(Integer.toHexString(0xFF & byteArray[i]));

}

System.out.println("Data in Hex format : " + hexData.toString());

}

}