**Laboratorio N°02**

Alumno: Chañi Laura, Jose Carlos

**Laboratorio N°2 - Ing. Oscar Ramirez**

1. Ejercicio 1: Cifrado AES en modo CBC

from Crypto.Random import get\_random\_bytes

from Crypto.Cipher import DES, AES

from Crypto.Util.Padding import pad,unpad

from Crypto.Util import Counter

import base64

# Datos necesarios

key = get\_random\_bytes(16) # Clave aleatoria de 128 bits

IV = get\_random\_bytes(16) # IV aleatorio de 128 bits para CBC

BLOCK\_SIZE\_AES = 8 # Bloque de 64 bits

data1 = "Hola amigos de la seguridad".encode("utf-8") # Datos a cifrar

data2 = "Hola amigas de la seguridad".encode("utf-8") # Datos a cifrar

print('Texto1: ',data1)

print('Texto2: ',data2)

# CIFRADO #######################################################################

# Creamos un mecanismo de cifrado DES en modo CBC con un vector de inicialización IV

cipher = AES.new(key, AES.MODE\_CBC, IV)

# Ciframos, haciendo que la variable “data” sea múltiplo del tamaño de bloque

ciphertext1 = cipher.encrypt(pad(data1,BLOCK\_SIZE\_AES))

ciphertext2 = cipher.encrypt(pad(data2,BLOCK\_SIZE\_AES))

# Mostramos el cifrado por pantalla en modo binario y en modo base 64

print('Cifrado de Texto1: ',ciphertext1)

print('Cifrado de Texto2: ',ciphertext2)

encoded\_ciphertext1 = base64.b64encode(ciphertext1)

encoded\_ciphertext2 = base64.b64encode(ciphertext2)

print('b64 encode(text1): ',encoded\_ciphertext1)

print('b64 encode(text2): ',encoded\_ciphertext2)

# DESCIFRADO #######################################################################

# Creamos un mecanismo de (des)cifrado DES en modo CBC con un vector de

# inicialización IV para CBC

# Ambos, cifrado y descifrado, se crean de la misma forma

decipher\_des = AES.new(key, AES.MODE\_CBC, IV)

# Desciframos, eliminamos el padding, y recuperamos la cadena

new\_data1 = unpad(decipher\_des.decrypt(ciphertext1), BLOCK\_SIZE\_AES).decode("utf-8","ignore")

new\_data2 = unpad(decipher\_des.decrypt(ciphertext2), BLOCK\_SIZE\_AES).decode("utf-8","ignore")

# Imprimimos los datos descifrados

print('Data descifrada: ',new\_data1)

print('Data descifrada: ',new\_data2)

* 1. Si se observa los textos cifrado, es posible ver que ese cambio de una “o” por una “a” (amigos →amigas) impacta en ambos textos, ¿a qué se debe ese cambio?

Es debido a que con el cifrado AES en modo CBC el vector IV para el cifrado del primer bloque tiene como resultado el nuevo vector IV para el segundo bloque y así sucesivamente, de modo que un cambio pequeño en la cadena genera un nuevo texto cifrado para el resto del texto.

1. Ejercicio 2: AES en diferentes modos
   1. ECB

def crypAESmodeECB(data1):

key = get\_random\_bytes(16) # Clave aleatoria de 128 bits

BLOCK\_SIZE\_AES = 8 # Bloque de 64 bits

print('Texto1: ',data1)

# CIFRADO ########################################################################

# # Creamos un mecanismo de cifrado AES en modo ECB con un vector de inicialización IV

cipher = AES.new(key, AES.MODE\_CBC)

# Ciframos, haciendo que la variable “data” sea múltiplo del tamaño de bloque

ciphertext1 = cipher.encrypt(pad(data1,BLOCK\_SIZE\_AES))

# Mostramos el cifrado por pantalla en modo binario y en modo base 64

print('Cifrado de Texto1: ',ciphertext1)

encoded\_ciphertext1 = base64.b64encode(ciphertext1)

print('b64 encode(text1): ',encoded\_ciphertext1)

# DESCIFRADO #######################################################################

# Creamos un mecanismo de (des)cifrado DES en modo CBC con un vector de

# inicialización IV para CBC

# Ambos, cifrado y descifrado, se crean de la misma forma

decipher\_aes = AES.new(key, AES.MODE\_ECB)

# Desciframos, eliminamos el padding, y recuperamos la cadena

new\_data1 = unpad(decipher\_aes.decrypt(ciphertext1), BLOCK\_SIZE\_AES).decode("utf-8","ignore")

# Imprimimos los datos descifrados

print('Data descifrada: ',new\_data1)

* 1. CTR

def crypAESmodeCTR(data1):

key = get\_random\_bytes(16)

nonce\_bytes = get\_random\_bytes(8)

print('Texto: ',data1)

cipher = AES.new(key, AES.MODE\_CTR, nonce=nonce\_bytes)

cipherText\_bytes = cipher.encrypt(data1)

print('Cifrado de Texto (bytes): ',cipherText\_bytes)

cipherText = base64.b64encode(cipherText\_bytes).decode('utf-8')

nonce= base64.b64encode(nonce\_bytes).decode('utf-8')

print('nonce: ',nonce)

print('Texto Cifrado: ', cipherText)

##Decipher

decipher\_des = AES.new(key, AES.MODE\_CTR, nonce=nonce\_bytes)

decipherText = decipher\_des.decrypt(cipherText\_bytes)

print('Mensaje Descifrado: ',decipherText)

* 1. OFB

def crypAESmodeOFB(data1):

key = get\_random\_bytes(16)

IV = get\_random\_bytes(16)

print('Texto: ',data1)

##Cifrado

cipher= AES.new(key, AES.MODE\_OFB, iv=IV)

cipherText\_bytes = cipher.encrypt(data1)

print("Texto cifrado (bytes): ",cipherText\_bytes)

cipherText = base64.b64encode(cipherText\_bytes).decode('utf-8')

print("Texto cifrado: ", cipherText)

print("Clave IV: ", base64.b64encode(IV).decode('utf-8'))

decipher\_des = AES.new(key, AES.MODE\_OFB, iv=IV)

decipherText = decipher\_des.decrypt(cipherText\_bytes)

print("Texto descifrado: ", decipherText)

* 1. CFB

def crypAESmodeCFB(data1):

key = get\_random\_bytes(16)

IV = get\_random\_bytes(16)

print('Texto: ',data1)

##Cifrado

cipher= AES.new(key, AES.MODE\_CFB, iv=IV)

cipherText\_bytes = cipher.encrypt(data1)

print("Texto cifrado (bytes): ",cipherText\_bytes)

cipherText = base64.b64encode(cipherText\_bytes).decode('utf-8')

print("Texto cifrado: ", cipherText)

print("Clave IV: ", base64.b64encode(IV).decode('utf-8'))

decipher\_des = AES.new(key, AES.MODE\_CFB, iv=IV)

decipherText = decipher\_des.decrypt(cipherText\_bytes)

print("Texto descifrado: ", decipherText)

* 1. GCM

def crypAESmodeGCM(header, data1):

key = get\_random\_bytes(16)

nonce = get\_random\_bytes(16)

mac\_len = 8

header=header.encode('utf-8')

print('Texto: ',data1)

##Cifrado

cipher= AES.new(key, AES.MODE\_GCM, nonce=nonce, mac\_len=mac\_len)

cipher.update(header)

cipherText\_bytes, tag = cipher.encrypt\_and\_digest(data1)

print("Texto cifrado (bytes): ",cipherText\_bytes)

cipherText = base64.b64encode(cipherText\_bytes).decode('utf-8')

print("Texto cifrado: ", cipherText)

print("nonce: ", base64.b64encode(nonce).decode('utf-8'))

decipher\_des = AES.new(key, AES.MODE\_GCM, nonce=nonce, mac\_len=mac\_len)

decipher\_des.update(header)

decipherText = decipher\_des.decrypt\_and\_verify(cipherText\_bytes,tag)

print("Texto descifrado: ", decipherText)

1. Clase AES\_CIPHER\_CBC

from Crypto.Random import get\_random\_bytes

from Crypto.Cipher import AES

from Crypto.Util.Padding import pad,unpad

from Crypto.Util import Counter

import base64

class AES\_CIPHER\_CBC:

BLOCK\_SIZE\_AES = 16

def \_\_init\_\_(self, key):

self.key = key

pass

def cifrar(self, cadena, IV ):

cipher = AES.new(key, AES.MODE\_CBC, IV)

cadena = cadena.encode('utf-8')

result = cipher.encrypt(pad(cadena, AES\_CIPHER\_CBC.BLOCK\_SIZE\_AES))

result = base64.b64encode(result)

return result

def descifrar(self, cifrado, IV):

descipher = AES.new(key, AES.MODE\_CBC, IV)

cifrado\_bytes = base64.b64decode(cifrado)

result = unpad(descipher.decrypt(cifrado\_bytes), AES\_CIPHER\_CBC.BLOCK\_SIZE\_AES)

result = result.decode('utf-8','ignore')

return result

key = get\_random\_bytes(16) # Clave aleatoria de 128 bits

IV = get\_random\_bytes(16) # IV aleatorio de 128 bits

datos = "Hola Mundo con AES en modo CBC"

d = AES\_CIPHER\_CBC(key)

cifrado = d.cifrar(datos, IV)

print(cifrado)

descifrado = d.descifrar(cifrado, IV)

print(descifrado)

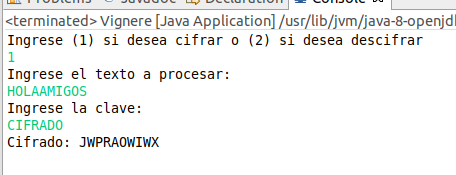


**Laboratorio N°2 - Ing. Lucy Delgado**

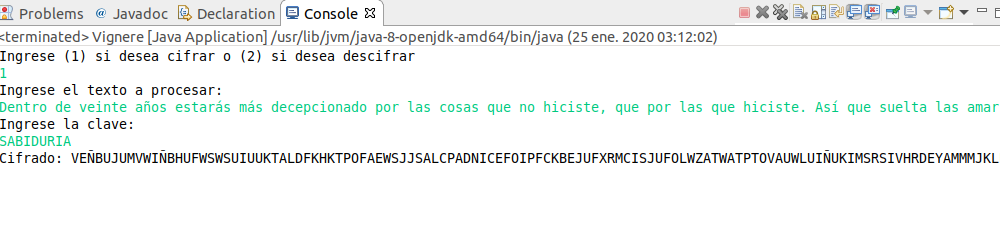
**Cifrado de Vignere**

**De acuerdo a lo explicado en clase, para la resolución de los siguientes ejercicios se trabajará solo en módulo 27.**

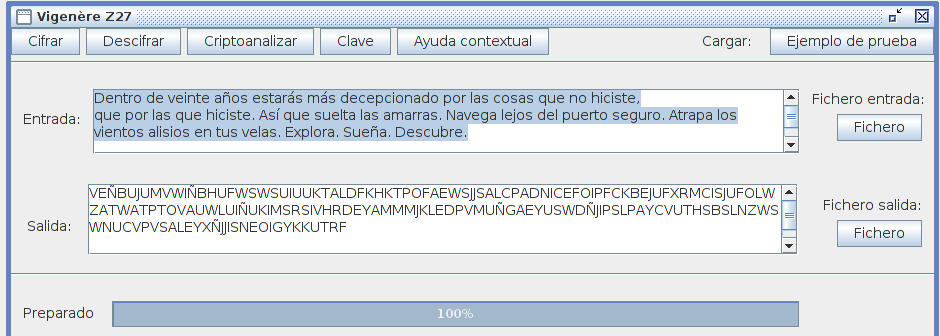
**1.** Implementar un cifrador de Vignere, donde se pueda seleccionar el módulo, alfabeto módulo 27 o módulo 191 (ASCII), ingresar el texto claro (en archivo o por interface) y genere la cifra resultante.



**2.** Verificar cifrando “Dentro de veinte años estarás más decepcionado por las cosas que no hiciste, que por las que hiciste. Así que suelta las amarras. Navega lejos del puerto seguro. Atrapa los vientos alisios en tus velas. Explora. Sueña. Descubre.” Usando la clave SABIDURIA.

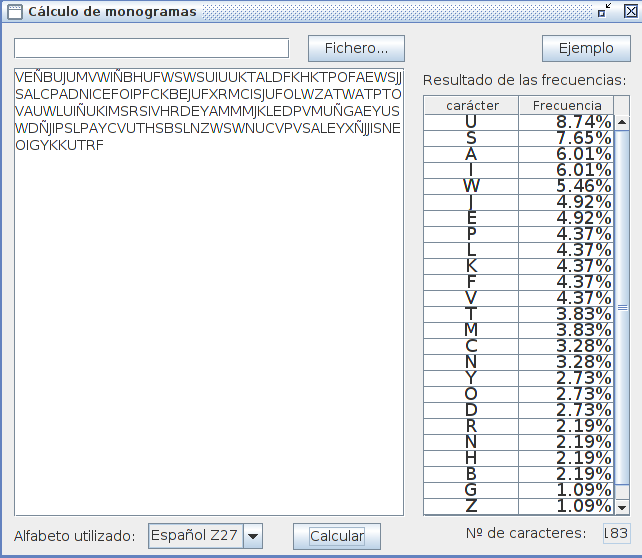


**3.** Verifica el resultado obtenido a partir del cifrador Criptoclásicos v2.1 (http://www.criptored.upm.es/software/sw\_m001c.htm ) haciendo las capturas de pantalla respectivas para los módulos 27 y 191

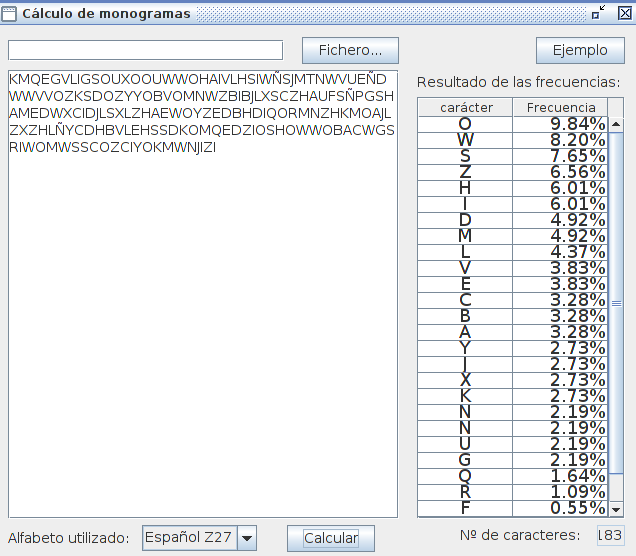


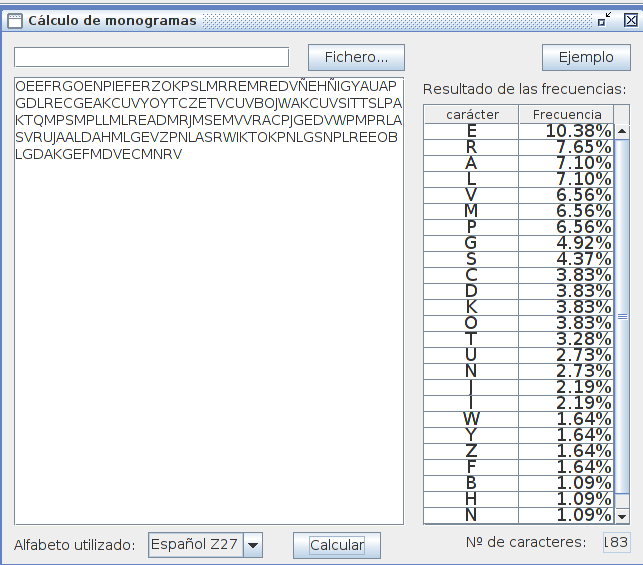
**4.** Muestra las frecuencias de cada letra del mensaje original usando como claves SABIDURIA, HIELO y MAR, compare y concluya sobre la variación de las frecuencias en base a la longitud de la clave.

SABIDURIA



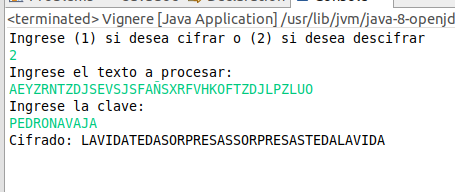
HIELO

  
MAR



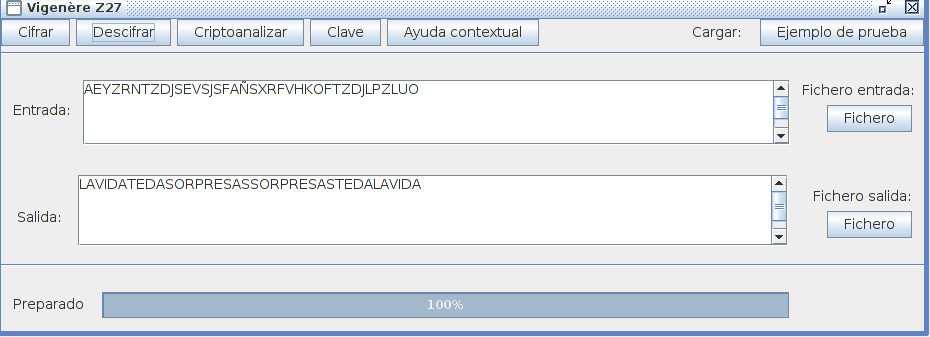
**5.** Desarrolle un algoritmo que encuentre el texto claro si recibió la cifra

AEYZRNTZDJSEVSJSFAÑSXRFVHKOFTZDJLPZLUO, y se sabe que ha cifrado con la clave PEDRONAVAJA



Separando las palabras sería: LA VIDA TE DA SORPRESAS SORPRESAS TE DA LA VIDA

**6.** Usando el software anterior, verifique el resultado, eligiendo el cifrado Vignere con módulo 27.



**7.** Usando matemáticas discretas, descifre manualmente YGVMSSKKOX si la clave fue

FORTALEZA en un alfabeto de 27 caracteres

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Y(25)** | **G(6)** | **V(22)** | **M(12)** | **S(19)** | **S(19)** | **K(10)** | **K(10)** | **O(15)** | **X(24)** |
| **F(5)** | **O(15)** | **R(18)** | **T(20)** | **A(0)** | **L(11)** | **E(4)** | **Z(26)** | **A(0)** | **F(5)** |
| **T-20** | **R-18** | **E-4** | **S-19** | **S-19** | **I-8** | **G-6** | **L-11** | **O-15** | **S-19** |

Separando las letras sería: TRES SIGLOS

**Cifrado con autoclave**

**8.** Descifra el texto, usando la clave UNODELOSMASGRANDESCRIPTOGRAFOS:

XHGDQESDMPKÑDEEDKNGJZPFJSUIFZOLFCINFJCESVZTGBFXCIUDAYNUUDIZYW

WZBEYNVQWIVUNKZEPHDODQUZZLBDNDRWTHQSERÑIVMLERCMGIFLSORZXTS

DIGLOXQSDJHWVCIWQXQJCKMBPOKMPSKMUVIMNJDNBLCSZHXHNYYUIXDBSO

XHZLXWVGDJGXHWLTDWKÑSAQIMZLNBVMLXHUOQQXIQGWGUFTWKZKMOKU

DNINSIFJDUOZIJBSVVOWFAIEÑGYOWPSOAP

**Descifrado el texto y con espacios sería:** DURANTE LA PRIMERA GUERRA MUNDIAL WILLIAM FREDERICK FRIEDMAN SIRVIO COMO TENIENTE EN LA UNIDAD DE CRIPTOLOGIA DE LA FUERZA EXPEDICIONARIA ESTADOUNIDENSE DISTINGUIENDOSE POR SUS TRABAJOS Y PROEZAS EN EL ANALISIS DE CODIGOS ENEMIGOS INVIOLABLES POR LO CUAL RECIBIO EL RECONOCIMIENTO DEL GOBIERNO ESTADOUNIDENSE

**Ataque de Kasiski**

**9.** Criptoanalizar el siguiente criptograma mod 27, encontrar la clave y el texto en claro.

MAXYHGAVAPUUGZHEGZQOWOBNIPQKRNÑMEXIGONIICUCAWIGCTEAGMNOLR

SZJNLWÑAWWIGLDDZSNIZDNBIXGZLAYMXÑCVEKIETMOEOPBEWPTNIXCXUIHM

ECXLNOCECYXEQPBWUFANIICÑJIKISCZUAILBGSOANKBFWUAYWNSCHLCWYDZ

HDZAQVMPTVGFGPVAJWFVPUOYMXCWERVLQCZWECIFVITUZSNCZUAIKBFMÑA

LIEGLBSZLQUXÑOHWOCGHNYWÑQKDANZUDIFOIMXNPHNUWQOKLMVBNNKRM

KONDPDPNMIKAWOXMEEIVEKGBGSFHVADWPGOYMHOIUEEIPGOLENZBSCHAGK

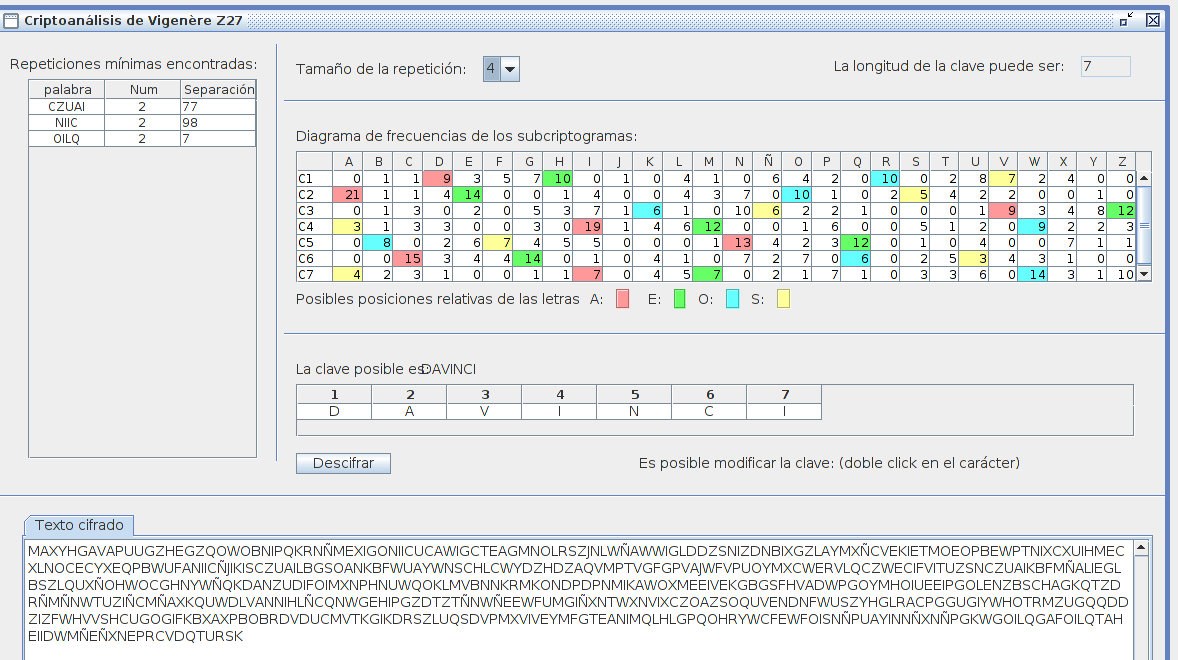
QTZDRÑMÑNWTUZIÑCMÑAXKQUWDLVANNIHLÑCQNWGEHIPGZDTZTÑNWÑEEW

FUMGIÑXNTWXNVIXCZOAZSOQUVENDNFWUSZYHGLRACPGGUGIYWHOTRMZUG

QQDDZIZFWHVVSHCUGOGIFKBXAXPBOBRDVDUCMVTKGIKDRSZLUQSDVPMXVI

VEYMFGTEANIMQLHLGPQOHRYWCFEWFOISNÑPUAYINNÑXNÑPGKWGOILQGAFO

ILQTAHEIIDWMÑEÑXNEPRCVDQTURSK



Siendo el texto descifrado:

JACQUESSA UNIERE EL RENOMBRADO CONSERVADOR AVANZABA TAMBALEANDOSE BAJO LA BOVEDA DE LA GRAN GALERIA DEL MUSEO ARREMETIO CONTRA LA PRIMERA PINTURA QUE VIO UN CARAVAGGIO AGARRANDO EL MARCO DORADO AQUEL HOMBRE DE SETENTA Y SEIS AÑOS TIRO DE LA OBRA DE ARTE HASTA QUE LA ARRANCO DE LA PARED Y SE DESPLOMO CAYENDO BOCA ARRIBA CON EL LIENZO ENCIMA TAL COMO HABIA PREVISTO CERCA SE OYO EL CHASQUIDO DE UNA REJA DE HIERRO QUE AL CERRAR SE BLOQUEABA EL ACCESO A LA SALA EL SUELO DE MADERA TEMBLO LEJOS SE DISPARO UNA ALARMA EL CONSERVADOR SE QUEDO AHI TENDIDO UN MOMENTO JADEANDO EVALUANDO LA SITUACION TODAVIA ESTOY VIVO SEDIO LA VUELTA SE DESEMBARAZO DEL LIENZO Y BUSCO CON LA MIRADA ALGUN SITIO DONDE ESCONDERSE EN AQUEL ESPACIO CAVERNOSO

**CONCLUSIONES**

**Emitir al menos ocho conclusiones en torno a las técnica de cifrado por sustitución**

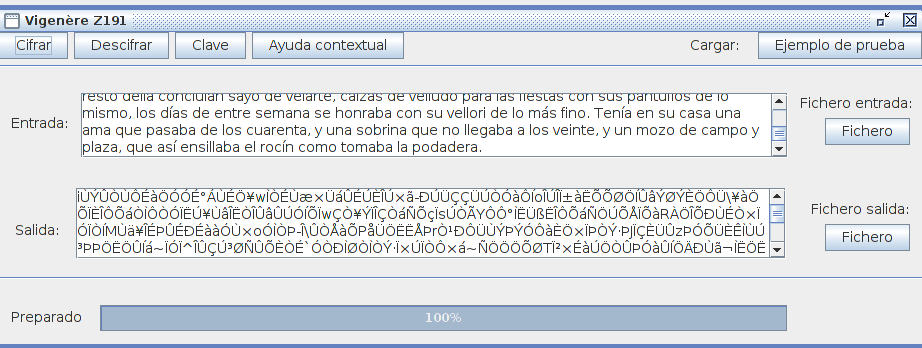
* El preprocesamiento ayuda a simplificar los algoritmos de encriptación.
* El encriptamiento por Vignere presenta una vulnerabilidad mediante la cual se puede averiguar la clave y con ello el mensaje entero.
* El análisis estadístico es una herramienta muy útil para el criptoanálisis de códigos.
* Para el criptoanálisis por Kasiski es muy importante conocer las frecuencias de los caracteres.
* El proceso de descifrado del método Vignere es solo el inverso del desplazamiento, por ello si se conoce la clave es muy fácil encontrar el mensaje.
* Para evitar esta vulnerabilidad se usa el método autoclave con el cual no solo dependemos de la clave sino del mensaje en sí.
* Python ofrece librerías para trabajar de manera más cómoda con los caracteres y el código ASCII.
* Para poder comprender mejor estos sistemas de encriptación es bueno usar un software ya desarrollado y que se pueda usar de punto de partida y guía.

**CUESTIONARIO FINAL**

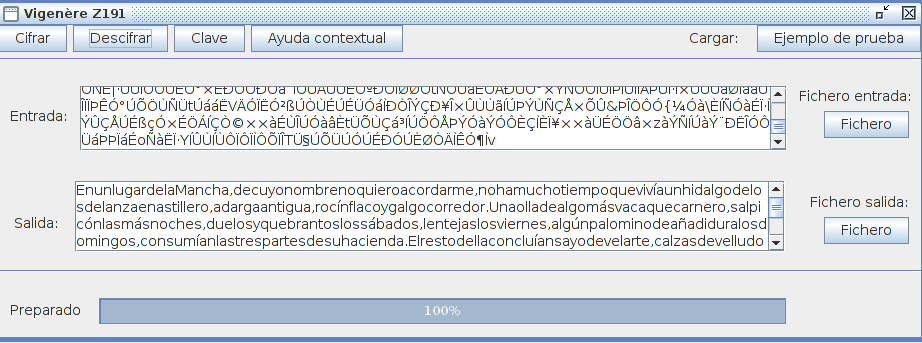
**1. Trabajando en módulo 191** (un subconjunto imprimible del código ASCII del software

Criptoclásicos), cifra el siguiente texto en claro con la clave: El ingenioso hidalgo.

En un lugar de la Mancha, de cuyo nombre no quiero acordarme, no ha mucho tiempo que vivía un hidalgo de los de lanza en astillero, adarga antigua, rocín flaco y galgo corredor. Una olla de algo más vaca que carnero, salpicón las más noches, duelos y quebrantos los sábados, lentejas los viernes, algún palomino de añadidura los domingos, consumían las tres partes de su hacienda. El resto della concluían sayo de velarte, calzas de velludo para las fiestas con sus pantuflos de lo mismo, los días de entre semana se honraba con su vellori de lo más fino. Tenía en su casa una ama que pasaba de los cuarenta, y una sobrina que no llegaba a los veinte, y un mozo de campo y plaza, que así ensillaba el rocín como tomaba la podadera.



**2. Descifra el criptograma en el mismo software ¿Por qué crees que el software no permite hacer un criptoanálisis?**

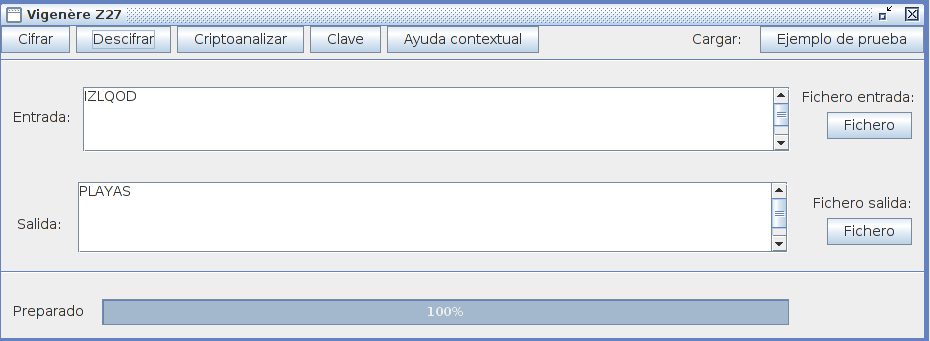


**¿Por qué crees que el software no permite hacer un criptoanálisis?**

Debido a que cuando trabajamos con 27 caracteres obtenemos solo 4 letras de mayor frecuencia (AEOS) pero si trabajamos con 191 caracteres este valor se dispararía y el calculo seria aun mas pesado.

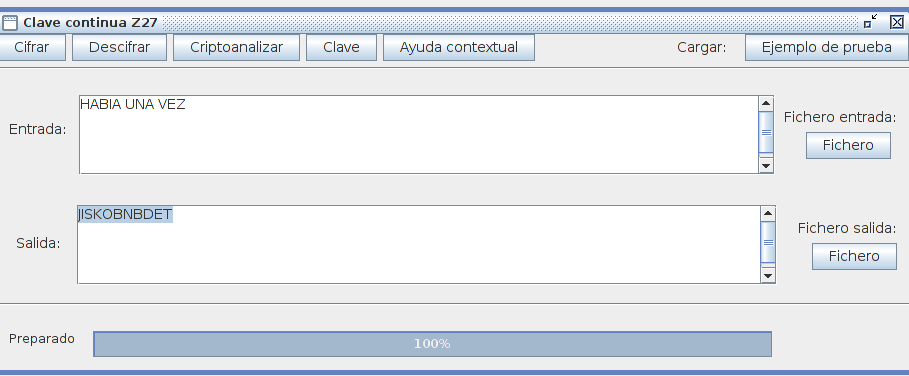
**3. Si el cifrado de Vigenere es IZLQOD y la clave SOL, ¿cuál era el mensaje en claro?**

El mensaje seria: PLAYAS



**4. ¿Cuál será la cifra con autoclave del texto HABIA UNA VEZ, con la clave CIRCO?**

El mensaje cifrado es: JISKOBNBDET



**5. En el ataque a Vigenere por Kasiski ¿Qué buscamos preferentemente?**

Se busca los caracteres que tengan mayor frecuencia y cadenas de 3 o más letras que se repitan además de la distancia entre estos.

**6. Encontradas las cadenas repetidas en el criptograma, con separación d1, d2, d3 y d4 ¿Cuál sería la longitud L de la clave?**

La posible longitud de la clave seria: maximo comun divisor de (d1,d2,d3,d4)

**7. Si las distancias entre repeticiones de cadenas en un criptograma son 35, 112, 70. ¿Cuál sería la longitud L de la clave?**

La posible longitud de la clave seria: 7

**8. ¿Qué diferencia la regla AEOS de AEO en Kasiski?**

Regla AEOS: Busca en cada uno de los subcriptograma las **cuatro** frecuencias más altas

Regla AEO: Busca en cada uno de los subcriptograma las **tres** frecuencias más altas

**CÓDIGO:**

**Vignere.java**

import java.util.Scanner;

import javax.swing.text.html.HTMLDocument.HTMLReader.PreAction;

public class Vignere {

public ABECEDARIO abc;

public Vignere() {

abc = new ABECEDARIO();

}

public static void main(String []arg) {

Vignere cifrado = new Vignere();

System.out.println(cifrado.abc);

Scanner sc = new Scanner(System.in);

System.out.println("Ingrese (1) si desea cifrar o (2) si desea descifrar");

int option = Integer.parseInt(sc.nextLine());

System.out.println("Ingrese el texto a procesar: ");

String texto = sc.nextLine();

System.out.println("Ingrese la clave: ");

String clave = sc.nextLine();

texto= cifrado.preProcesar(texto);

clave = cifrado.preProcesar(clave);

if(option == 1) {

System.out.println("Cifrado: "+cifrado.cifrar(texto, clave));

}else if(option == 2) {

System.out.println("Cifrado: "+cifrado.descifrar(texto, clave));

}else {

System.out.println("Opcion Incorrecta");

}

}

public String preProcesar(String text) {

char[] caracteres = text.toCharArray();

String cadenaSusti ="";

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

int iniOrder = (int) caracteres[i];

if(iniOrder>=97 && iniOrder<=122) {

iniOrder = (iniOrder-97)+65;

}

iniOrder = sustituir(iniOrder);

if((iniOrder>=65 && iniOrder<=90)|| iniOrder==209) {

cadenaSusti=cadenaSusti+ (char)iniOrder;

}

}

return cadenaSusti;

}

public int sustituir(int tilde) {

//a

if(tilde ==225 || tilde==193)

return 65;

//e

else if(tilde == 233 || tilde == 201)

return 69;

//i

else if (tilde == 237 || tilde == 205)

return 73;

//o

else if(tilde == 243 || tilde==211 )

return 79;

//u

else if(tilde ==250 || tilde==218)

return 85;

//ñ

else if(tilde == 241 || tilde == 209 )

return 209;

else

return tilde;

}

public String cifrar(String texto, String clave) {

char[] arrTexto = texto.toCharArray();

char[] arrClave = clave.toCharArray();

String resultado = "";

int index = 0;

while(index< arrTexto.length) {

int lugarInicial = abc.obtenerPosicion( arrTexto[index]);

int recorrido =abc.obtenerPosicion(arrClave[index % clave.length()]);

resultado= resultado+abc.obtenerLetra((lugarInicial+recorrido)%27);

index++;

}

return resultado;

}

public String descifrar(String texto, String clave) {

char[] arrTexto = texto.toCharArray();

char[] arrClave = clave.toCharArray();

String resultado = "";

int index = 0;

while(index< arrTexto.length) {

int lugarInicial = abc.obtenerPosicion( arrTexto[index]);

int recorrido =abc.obtenerPosicion(arrClave[index % clave.length()]);

int fin = (lugarInicial-recorrido)%27;

if(fin<0)

fin = 27-fin\*(-1);

resultado= resultado+abc.obtenerLetra(fin);

index++;

}

return resultado;

}

}

**ABECEDARIO.java**

public class ABECEDARIO {

public int len = 27;

char letras[];

public ABECEDARIO() {

letras=new char[len];

int posicion=0;

for(int i=0;i<len-1;i++) {

if(i<14) {

letras[i] = (char)(i+65);

}else

letras[i+1] = (char)(i+65);

}

letras[14] = (char)209;

}

public String toString() {

String result="";

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

result = result + letras[i]+"("+i+")"+" ";

}

return result;

}

public char obtenerLetra(int letra) {

if(letra==-1000) {

return letras[letras.length-1];

}

return letras[letra];

}

public int obtenerPosicion(char caracter) {

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

if(letras[i]==caracter)

return i;

}

return -1;

}

}