

**INSTITUTO POLITÉCNICO NACIONAL**

**UNIDAD PROFESIONAL INTERDISCIPLINARIA DE INGENIERÍA Y CIENCIAS SOCIALES Y ADMINISTRATIVAS**

**GUZMÁN RIVAS JOSUE**

**HERNÁNDEZ CRUZ LUIS FERNANDO**

**FUNDAMENTOS DE INGENIERÍA DE SOFTWARE**

**“ENCRIPTACIÓN”**

**RAMIREZ FLORES GILBERTO ANTONIO**

**2NM40**

**Encriptación**

**¿Qué es la encriptación?**

Encriptar o cifrar una información es ocultar un menaje con una contraseña.

Desde un punto de vista informático consiste en aplicar un algoritmo asociado a una o varias contraseñas, que convierte la información en una cadena de letras, números, y símbolos sin sentido.

Encriptar es una manera de codificar la información para protegerla frente a terceros.

Por lo tanto, la encriptación informática sería la codificación la información de archivos o de un correo electrónico para que no pueda ser descifrado en caso de ser interceptado por alguien mientras esta información viaja por la red.

Es por medio de la encriptación informática como se codifican los datos. Solamente a través de un software de descodificación que conoce el autor de estos documentos encriptados es como se puede volver a decodificar la información.

Por lo que la encriptación informática es simplemente la codificación de la información que vamos a enviar a través de la red (Internet). Para poder descodificarla como dijimos es necesario un software o una clave que sólo conocen el emisor y el receptor de esta información.

La encriptación de la informática se hace cada vez más necesaria debido al aumento de los robos de claves de tarjetas de crédito, número de cuentas corrientes, y en general toda la información que viaja por la red, etc.

La encriptación no es algo nuevo. Hace 2.500 años se encriptaban mensajes durante las campañas militares para que el enemigo no pudiera leerlos, si los interceptaba. Los Aliados ganaron la Segunda Guerra Mundial (o al menos adelantaron su final uno o dos años) gracias a que descifraron el código de la máquina de encriptación Enigma de los nazis, y pudieron acceder a sus comunicaciones de alto secreto:

**Ventajas e inconvenientes**

Si encriptas un dispositivo nadie podrá ver lo que contiene. Ten en cuenta que es un sistema de seguridad mucho más fuerte que una contraseña en la pantalla de inicio del móvil o el PC, que sólo protege el acceso, pero no los datos. Basta con extraer el disco duro de un PC y usarlo en otro ordenador para romper la contraseña de acceso de Windows, y hay herramientas para arrancar el móvil en modo root y acceder a su contenido. La encriptación evita esto porque cada fichero, cada dato, están encriptado.

Sin embargo, si pierdes la contraseña de encriptación o el disco duro se corrompe o falla, perderás el acceso a todos los datos. Además, en dispositivos con poca potencia de proceso la encriptación ralentiza el acceso a los datos, pues deben ser encriptados y desencriptados en tiempo real.

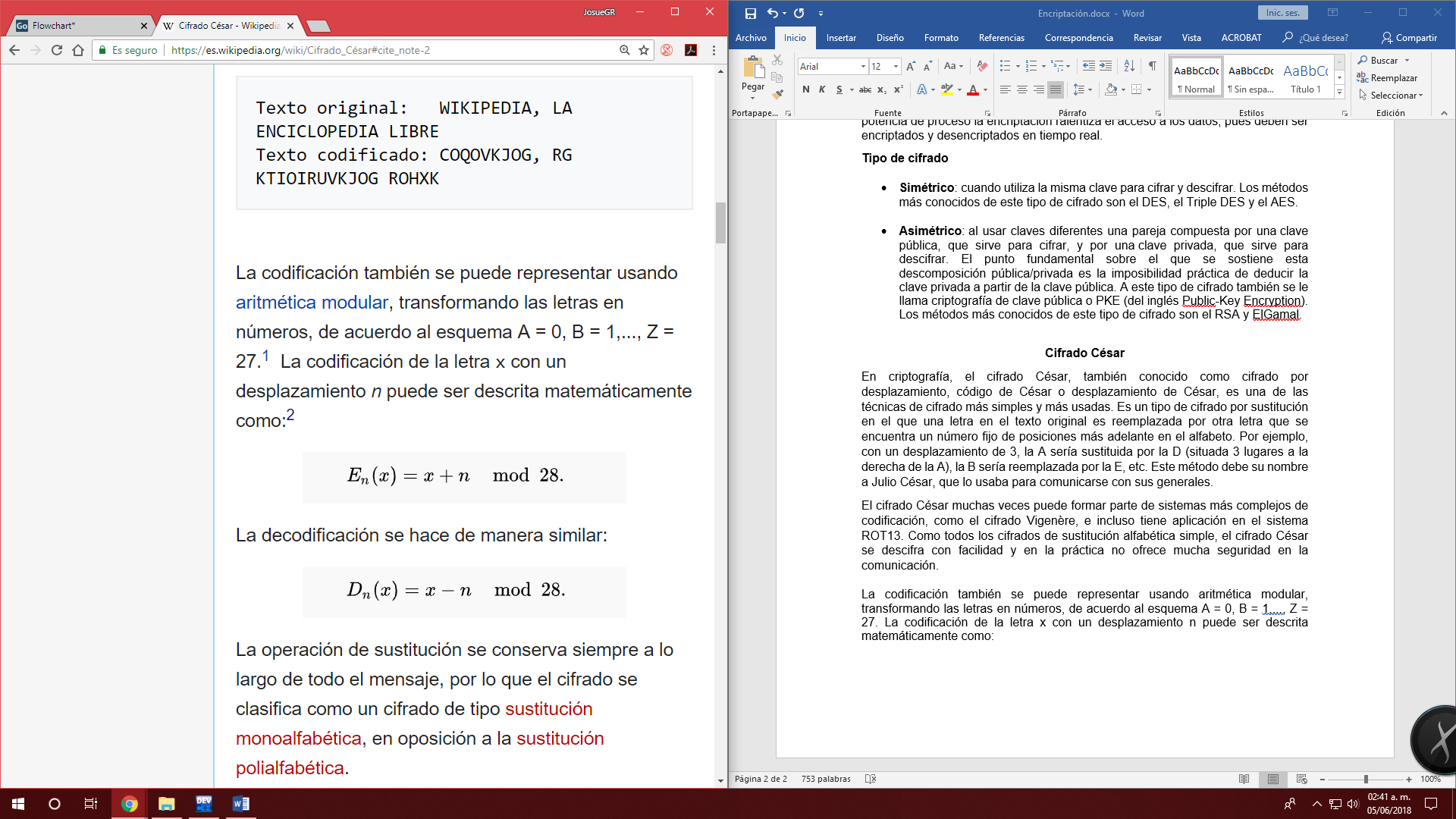
**Tipo de cifrado**

* **Simétrico**: cuando utiliza la misma clave para cifrar y descifrar. Los métodos más conocidos de este tipo de cifrado son el [DES](https://es.wikipedia.org/wiki/Data_Encryption_Standard), el [Triple DES](https://es.wikipedia.org/wiki/Triple_DES) y el [AES](https://es.wikipedia.org/wiki/Advanced_Encryption_Standard).
* **Asimétrico**: al usar claves diferentes una pareja compuesta por una clave pública, que sirve para cifrar, y por una clave privada, que sirve para descifrar. El punto fundamental sobre el que se sostiene esta descomposición pública/privada es la imposibilidad práctica de deducir la clave privada a partir de la clave pública. A este tipo de cifrado también se le llama [criptografía de clave pública](https://es.wikipedia.org/wiki/Criptograf%C3%ADa_de_clave_p%C3%BAblica) o PKE (del [inglés](https://es.wikipedia.org/wiki/Idioma_ingl%C3%A9s) Public-Key Encryption). Los métodos más conocidos de este tipo de cifrado son el [RSA](https://es.wikipedia.org/wiki/RSA) y [ElGamal](https://es.wikipedia.org/wiki/Cifrado_ElGamal" \o "Cifrado ElGamal).

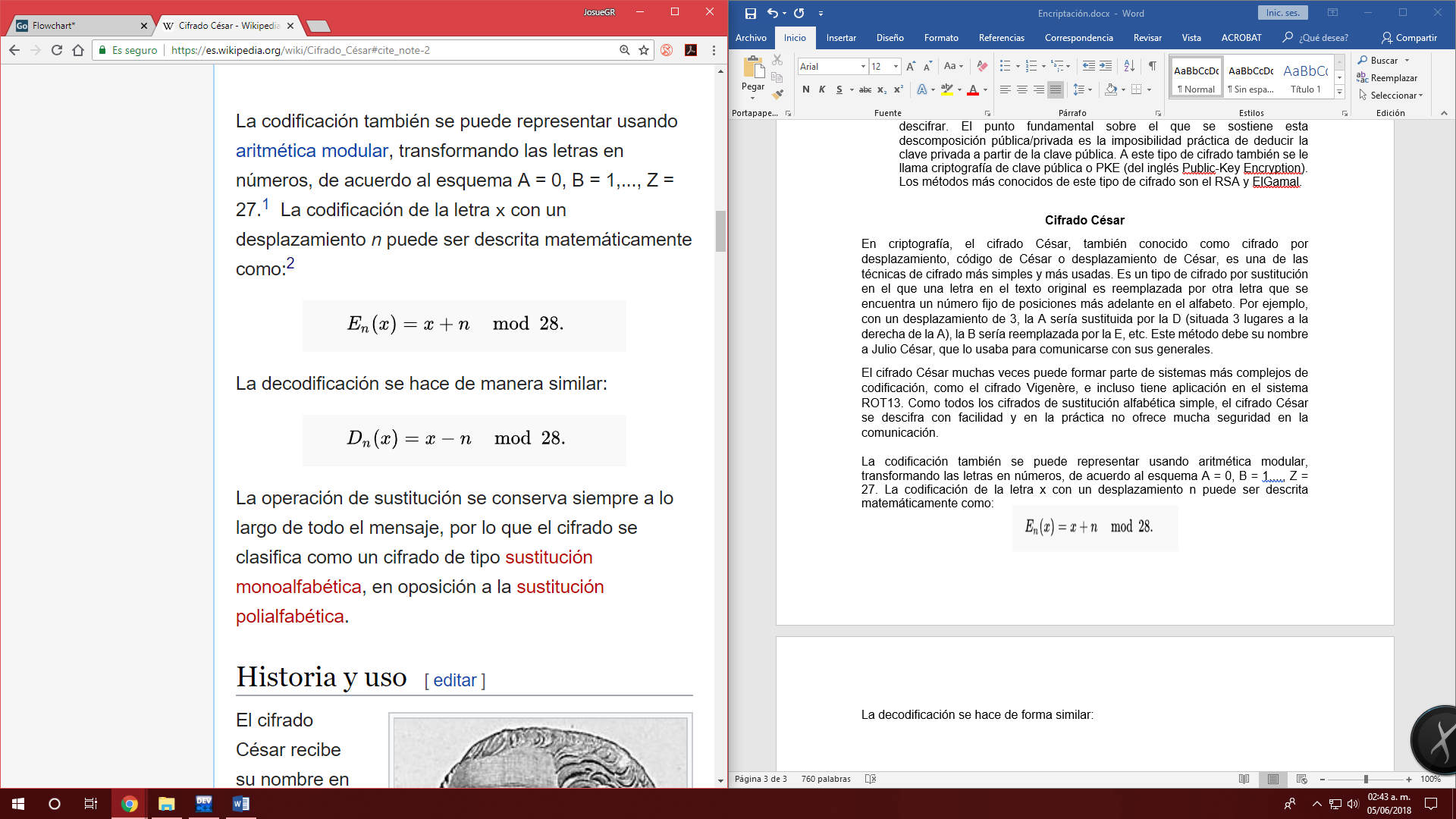
**Cifrado César**

En criptografía, el cifrado César, también conocido como cifrado por desplazamiento, código de César o desplazamiento de César, es una de las técnicas de cifrado más simples y más usadas. Es un tipo de cifrado por sustitución en el que una letra en el texto original es reemplazada por otra letra que se encuentra un número fijo de posiciones más adelante en el alfabeto. Por ejemplo, con un desplazamiento de 3, la A sería sustituida por la D (situada 3 lugares a la derecha de la A), la B sería reemplazada por la E, etc. Este método debe su nombre a Julio César, que lo usaba para comunicarse con sus generales.

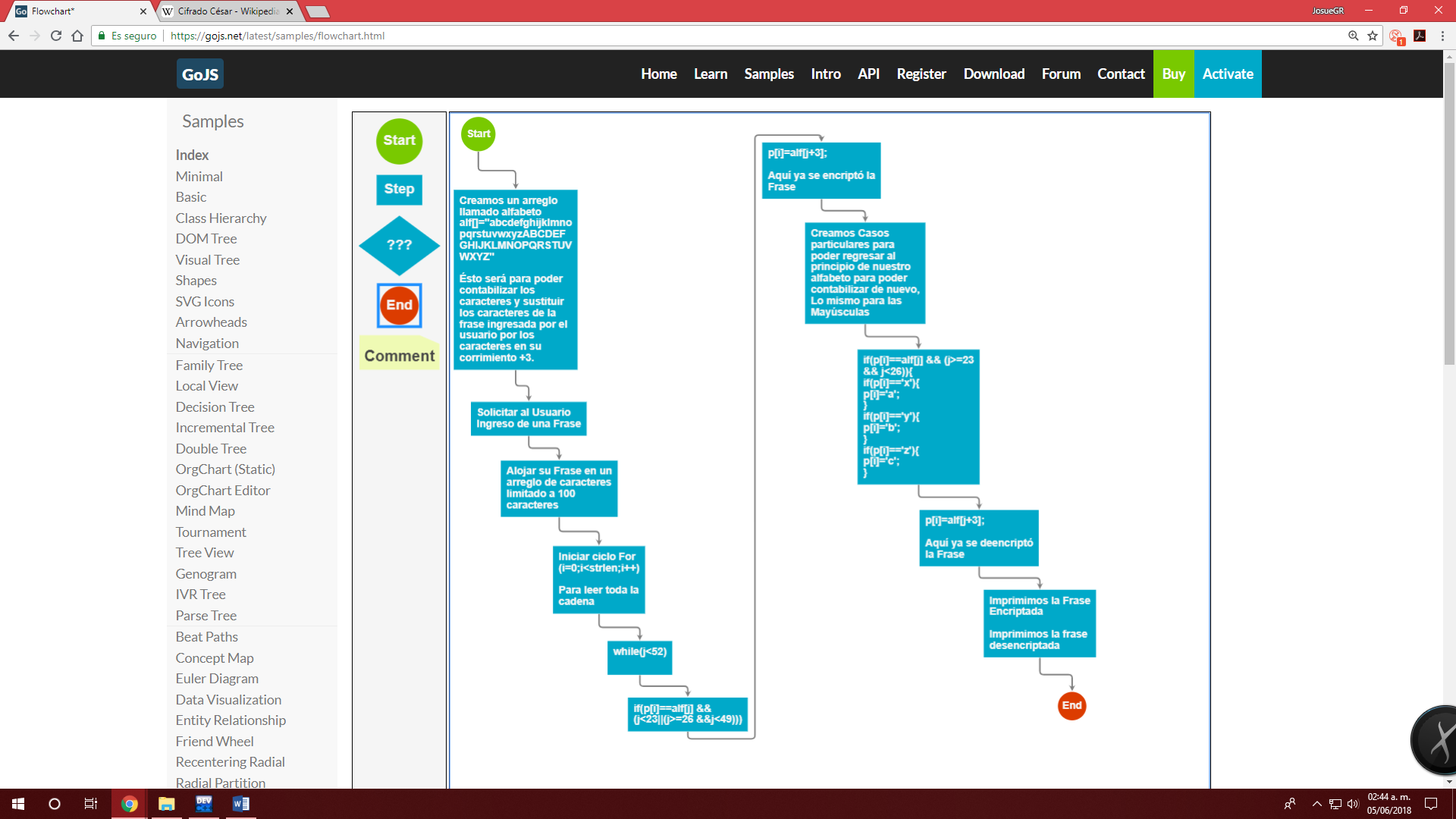
El cifrado César muchas veces puede formar parte de sistemas más complejos de codificación, como el cifrado Vigenère, e incluso tiene aplicación en el sistema ROT13. Como todos los cifrados de sustitución alfabética simple, el cifrado César se descifra con facilidad y en la práctica no ofrece mucha seguridad en la comunicación.

La codificación también se puede representar usando aritmética modular, transformando las letras en números, de acuerdo al esquema A = 0, B = 1,..., Z = 27. La codificación de la letra x con un desplazamiento n puede ser descrita matemáticamente como:

La decodificación se hace de forma similar:



**Diagrama de Flujo**



**Flowchart GoJS**

{ "class": "go.GraphLinksModel",

"linkFromPortIdProperty": "fromPort",

"linkToPortIdProperty": "toPort",

"nodeDataArray": [

{"key":-1, "category":"Start", "loc":"-471.44671706314966 -147.06156030481725", "text":"Start"},

{"text":"Solicitar al Usuario Ingreso de una Frase", "key":-2, "loc":"-400.19576954565855 254.9651385712244"},

{"text":"Alojar su Frase en un arreglo de caracteres limitado a 100 caracteres", "key":-3, "loc":"-357.1664879397563 353.71688390438425"},

{"text":"Iniciar ciclo For (i=0;i<strlen;i++)\n\nPara leer toda la cadena", "key":-4, "loc":"-300.9674703685318 482.4036493049922"},

{"text":"while(j<52)\n", "key":-5, "loc":"-243.25241527521632 592.6453985318151"},

{"text":"if(p[i]==alf[j] && (j<23||(j>=26 &&j<49)))", "key":-6, "loc":"-175.57584821228232 672.5027088838117"},

{"text":"p[i]=alf[j+3];\n\nAquí ya se encriptó la Frase ", "key":-7, "loc":"13.235735000868544 -95.46972645685699"},

{"text":"if(p[i]==alf[j] && (j>=23 && j<26)){\n\t if(p[i]=='x'){\n\t p[i]='a';\n\t }\n\t if(p[i]=='y'){\n\t p[i]='b';\n\t }\n\t if(p[i]=='z'){\n\t p[i]='c';\n\t }", "key":-8, "loc":"150.35180973437525 252.44362007994883"},

{"text":"Creamos Casos particulares para poder regresar al principio de nuestro alfabeto para poder contabilizar de nuevo, Lo mismo para las Mayúsculas", "key":-9, "loc":"74.99675002790849 49.42142979943671"},

{"text":"Creamos un arreglo llamado alfabeto alf[]=\"abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ\"\n\nÉsto será para poder contabilizar los caracteres y sustituir los caracteres de la frase ingresada por el usuario por los caracteres en su corrimiento +3.", "key":-10, "loc":"-418.7152929915352 58.68593565260463"},

{"text":"p[i]=alf[j+3];\n\nAquí ya se deencriptó la Frase ", "key":-11, "loc":"235.85683274999985 423.4702224375002"},

{"text":"Imprimimos la Frase Encriptada\n\nImprimimos la frase desencriptada", "key":-12, "loc":"321.6229537500001 544.0788300937502"},

{"category":"End", "text":"End", "key":-13, "loc":"367.1862055312498 660.6671508281252"}

],

"linkDataArray": [

{"from":-3, "to":-4, "fromPort":"B", "toPort":"T", "points":[-357.1664879397563,393.967781121181,-357.1664879397563,403.967781121181,-357.1664879397563,414.09140445258856,-300.9674703685318,414.09140445258856,-300.9674703685318,424.2150277839961,-300.9674703685318,434.2150277839961]},

{"from":-6, "to":-7, "fromPort":"B", "toPort":"T", "points":[-175.57584821228232,696.8781574922101,-175.57584821228232,706.8781574922101,-80.74237182177134,706.8781574922101,-80.74237182177134,-145.72062367365385,13.235735000868544,-145.72062367365385,13.235735000868544,-135.72062367365385]},

{"from":-5, "to":-6, "fromPort":"B", "toPort":"T", "points":[-243.25241527521632,617.0208471402134,-243.25241527521632,627.0208471402134,-243.25241527521632,632.5740537078134,-175.57584821228232,632.5740537078134,-175.57584821228232,638.1272602754133,-175.57584821228232,648.1272602754133]},

{"from":-4, "to":-5, "fromPort":"B", "toPort":"T", "points":[-300.9674703685318,530.5922708259883,-300.9674703685318,540.5922708259883,-300.9674703685318,549.4311103747025,-243.25241527521632,549.4311103747025,-243.25241527521632,558.2699499234166,-243.25241527521632,568.2699499234166]},

{"from":-7, "to":-9, "fromPort":"B", "toPort":"T", "points":[13.235735000868544,-55.2188292400601,13.235735000868544,-45.2188292400601,13.235735000868544,-38.89959693710858,74.99675002790849,-38.89959693710858,74.99675002790849,-32.580364634157064,74.99675002790849,-22.580364634157064]},

{"from":-9, "to":-8, "fromPort":"B", "toPort":"T", "points":[74.99675002790849,121.42322423303042,74.99675002790849,131.42322423303042,74.99675002790849,139.0259384833939,150.35180973437525,139.0259384833939,150.35180973437525,146.62865273375738,150.35180973437525,156.62865273375738]},

{"from":-1, "to":-10, "fromPort":"B", "toPort":"T", "points":[-471.4467170631496,-122.28822021266836,-471.4467170631496,-112.28822021266836,-471.4467170631496,-95.58407456152601,-418.7152929915352,-95.58407456152601,-418.7152929915352,-78.87992891038365,-418.7152929915352,-68.87992891038365]},

{"from":-10, "to":-2, "fromPort":"B", "toPort":"T", "points":[-418.7152929915352,186.2518002155929,-418.7152929915352,196.2518002155929,-418.7152929915352,208.4207450892094,-400.19576954565855,208.4207450892094,-400.19576954565855,220.58968996282587,-400.19576954565855,230.58968996282587]},

{"from":-2, "to":-3, "fromPort":"B", "toPort":"T", "points":[-400.19576954565855,279.34058717962273,-400.19576954565855,289.34058717962273,-400.19576954565855,296.40328693360505,-357.1664879397563,296.40328693360505,-357.1664879397563,303.4659866875873,-357.1664879397563,313.4659866875873]},

{"from":-8, "to":-11, "fromPort":"B", "toPort":"T", "points":[150.35180973437525,348.25858742614025,150.35180973437525,358.25858742614025,150.35180973437525,365.7389563234218,235.85683274999985,365.7389563234218,235.85683274999985,373.21932522070335,235.85683274999985,383.21932522070335]},

{"from":-11, "to":-12, "fromPort":"B", "toPort":"T", "points":[235.85683274999985,463.7211196542971,235.85683274999985,473.7211196542971,235.85683274999985,479.8056641135256,321.6229537500001,479.8056641135256,321.6229537500001,485.8902085727541,321.6229537500001,495.8902085727541]},

{"from":-12, "to":-13, "fromPort":"B", "toPort":"T", "points":[321.6229537500001,592.2674516147463,321.6229537500001,602.2674516147463,321.6229537500001,616.1562866911652,367.1862055312498,616.1562866911652,367.1862055312498,630.045121767584,367.1862055312498,640.045121767584]}

]}

**Código**

/\*cifrado Cesar :

consta de un desplazamiento de tres espacios al alfabeto por

ejemplo a--->d ;Z--->C\*/

#include<iostream>

#include<cstring>

using namespace std;

void Menu();

int main()

{

char p[100];

char alf[]="abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ";

int j;

cout<<"Dame una Frase: ";

gets(p);

////////////// Encriptador /////////////////////

for(int i=0;i<strlen(p);i++)

{

j=0;

while(j<52)

{

if(p[i]==alf[j]&&(j<23||(j>=26&&j<49)))

{

p[i]=alf[j+3];

j=52;

}

if(p[i]==alf[j]&&(j>=23&&j<26))

{

if(p[i]=='x')

{

p[i]='a';

}

if(p[i]=='y')

{

p[i]='b';

}

if(p[i]=='z')

{

p[i]='c';

}

j=52;

}

if(p[i]==alf[j] && (j>=49 && j<52))

{

if(p[i]=='X')

{

p[i]='A';

}

if(p[i]=='Y')

{

p[i]='B';

}

if(p[i]=='Z')

{

p[i]='C';

}

j=52;

}

j++;

}

}

cout<<"Frase Encriptada: ";

cout<<p;

///////////// Desencriptador //////////////

for(int i=0;i<strlen(p);i++)

{

j=0;

while(j<52)

{

if(p[i]==alf[j]&&(j<23||(j>=26&&j<49)))

{

p[i]=alf[j-3];

j=52;

}

if(p[i]==alf[j]&&(j>=23&&j<26))

{

if(p[i]=='x')

{

p[i]='u';

}

if(p[i]=='y')

{

p[i]='v';

}

if(p[i]=='z')

{

p[i]='w';

}

j=52;

}

if(p[i]==alf[j] && (j>=49 && j<52))

{

if(p[i]=='X')

{

p[i]='A';

}

if(p[i]=='Y')

{

p[i]='B';

}

if(p[i]=='Z')

{

p[i]='C';

}

j=52;

}

j++;

}

}

cout<<endl;

cout<<"Frase Desencriptada: ";

cout<<p;

return 0;

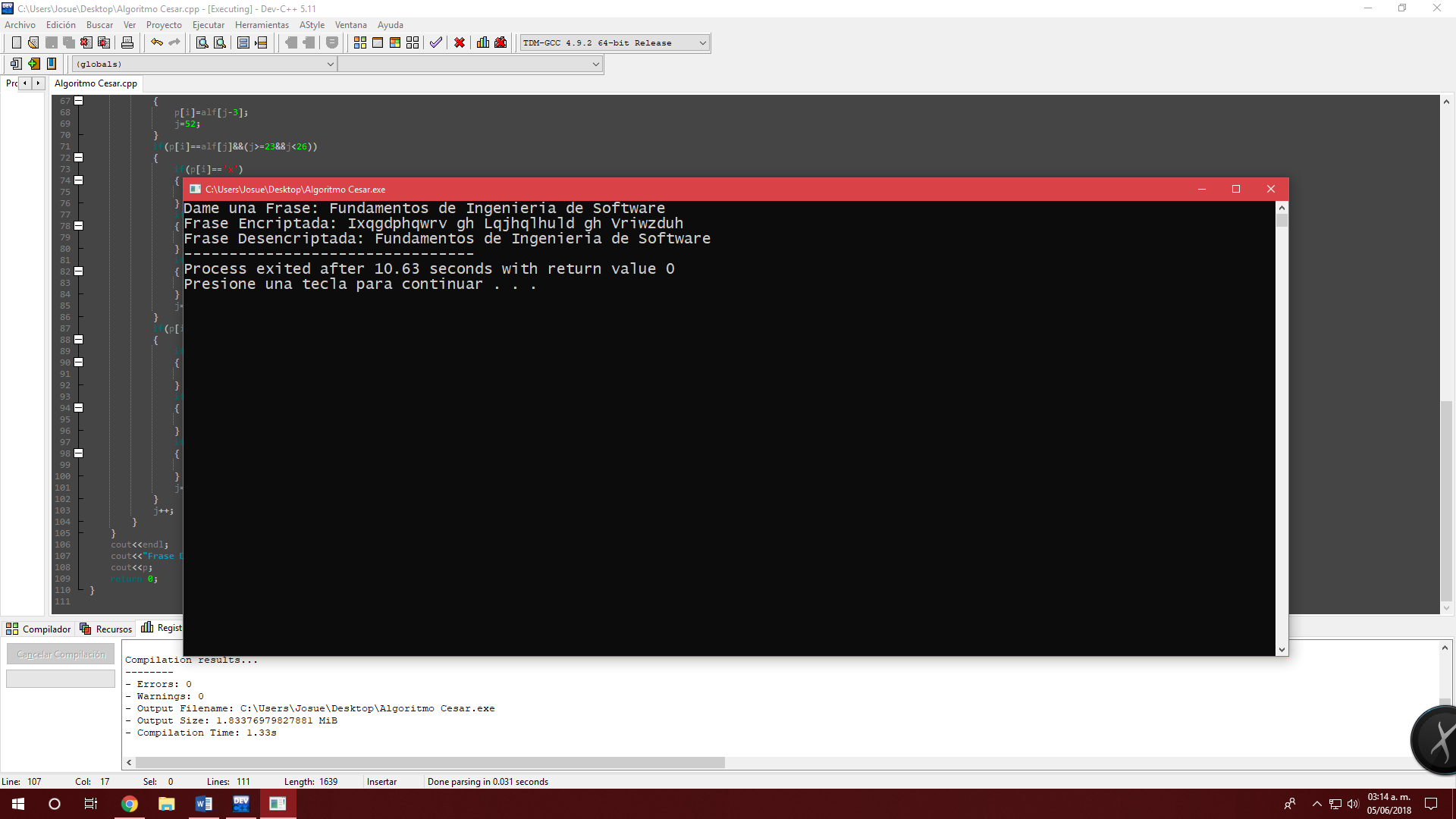
}

**Prueba de Escritorio**

**Ingrese frase: Fundamentos de Ingenieria de Software**

**Frase Encriptada: Ixqgdphqwrv gh Lqjhqlhuld gh Vriwzduh**

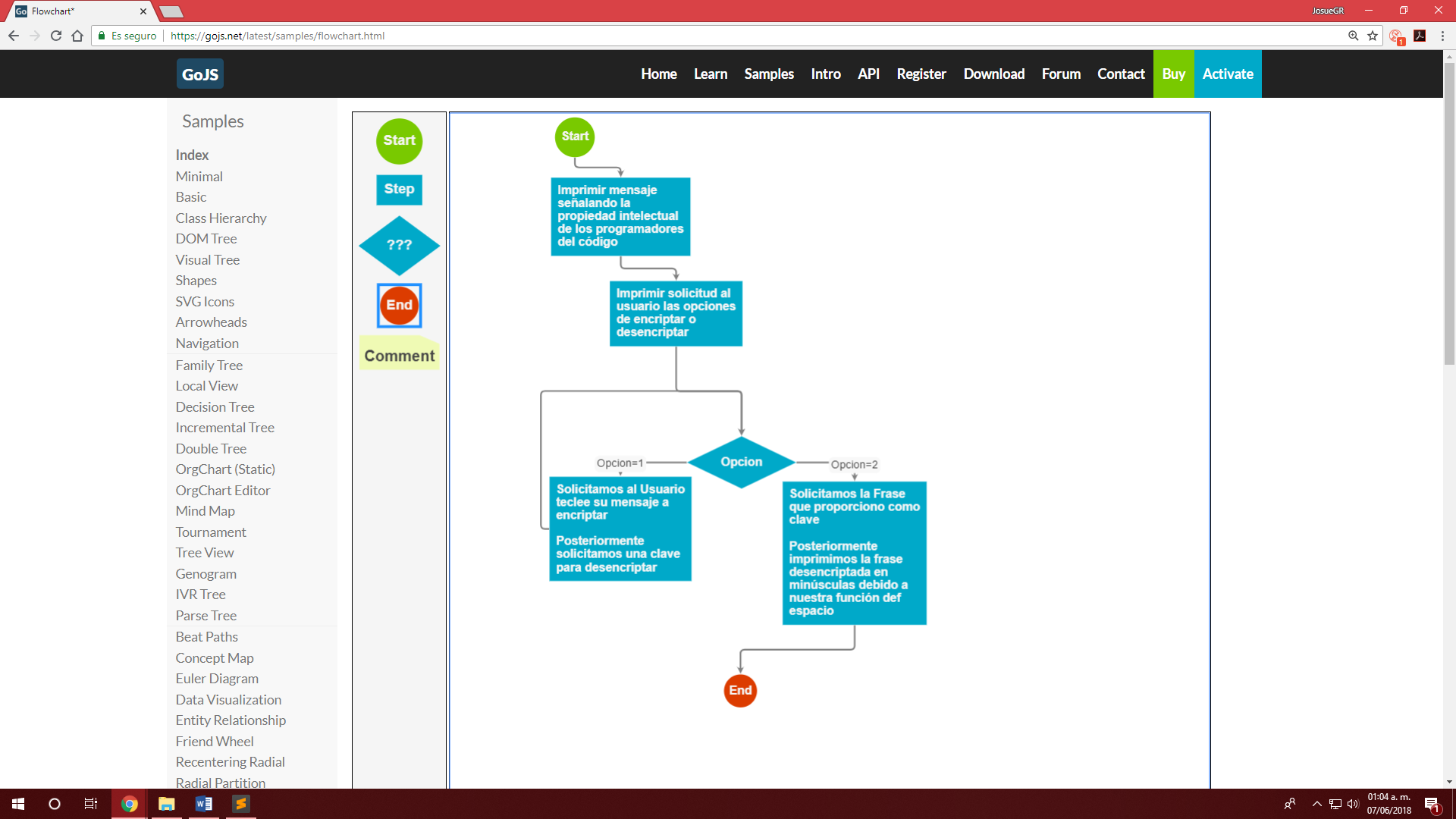
**Frase Desencriptada: Fundamentos de Ingenieria de Software**



**Encriptador con Python**

Éste encriptador recibe un texto ingresado por el usuario utilizando una clave proporcionada por el mismo usuario.

**Diagrama de Flujo**



**Flowchart GoJS**

{ "class": "go.GraphLinksModel",

"linkFromPortIdProperty": "fromPort",

"linkToPortIdProperty": "toPort",

"nodeDataArray": [

{"key":-1, "category":"Start", "loc":"-29.00000000000003 3", "text":"Start"},

{"text":"Imprimir mensaje señalando la propiedad intelectual de los programadores del código", "key":-2, "loc":"27 100"},

{"text":"Imprimir solicitud al usuario las opciones de encriptar o desencriptar", "key":-3, "loc":"94.60308837890625 218.22665990785111"},

{"text":"Opcion", "figure":"Diamond", "key":-4, "loc":"174.4186249999999 399.6642499999999"},

{"text":"Solicitamos al Usuario teclee su mensaje a encriptar\n\nPosteriormente solicitamos una clave para desencriptar", "key":-5, "loc":"26.668499999999895 480.69987652587895"},

{"text":"Solicitamos la Frase que proporciono como clave\n\nPosteriormente imprimimos la frase desencriptada en minúsculas debido a nuestra función def espacio", "key":-6, "loc":"312.333625 510.40512652587904"},

{"category":"End", "text":"End", "key":-7, "loc":"173.06493749999998 678.36825"}

],

"linkDataArray": [

{"from":-1, "to":-2, "fromPort":"B", "toPort":"T", "points":[-28.999999999999993,27.773340092148892,-28.999999999999993,37.77334009214889,-28.999999999999993,39.792359285576396,27,39.792359285576396,27,41.811378479003906,27,51.811378479003906]},

{"from":-2, "to":-3, "fromPort":"B", "toPort":"T", "points":[27,148.1886215209961,27,158.1886215209961,27,163.08219210602516,94.60308837890625,163.08219210602516,94.60308837890625,167.97576269105423,94.60308837890625,177.97576269105423]},

{"from":-3, "to":-4, "fromPort":"B", "toPort":"T", "points":[94.60308837890625,258.477557124648,94.60308837890625,268.477557124648,94.60308837890625,312.88317925812476,174.4186249999999,312.88317925812476,174.4186249999999,357.2888013916015,174.4186249999999,367.2888013916015]},

{"from":-4, "to":-5, "fromPort":"L", "toPort":"T", "visible":true, "points":[107.42468273925772,399.6642499999999,97.42468273925772,399.6642499999999,26.668499999999895,399.6642499999999,26.668499999999895,403.1500281982422,26.668499999999895,406.63580639648444,26.668499999999895,416.63580639648444], "text":"Opcion=1"},

{"from":-4, "to":-6, "fromPort":"R", "toPort":"T", "visible":true, "points":[241.4125672607421,399.6642499999999,251.4125672607421,399.6642499999999,312.333625,399.6642499999999,312.333625,406.0960667419434,312.333625,412.52788348388685,312.333625,422.52788348388685], "text":"Opcion=2"},

{"from":-5, "to":-4, "fromPort":"L", "toPort":"T", "points":[-60.449969238281355,480.69987652587895,-70.44996923828135,480.69987652587895,-70.44996923828135,312.55875000000003,174.4186249999999,312.55875000000003,174.4186249999999,357.2888013916015,174.4186249999999,367.2888013916015]},

{"from":-6, "to":-7, "fromPort":"B", "toPort":"T", "points":[312.333625,598.2823695678712,312.333625,608.2823695678712,312.333625,628.0142952536651,173.06493749999998,628.0142952536651,173.06493749999998,647.7462209394588,173.06493749999998,657.7462209394588]}

]}

**Código**

#coding: utf-8

print "Encriptador GuzmanRivasJosue-HernandezCruzLuisFernando"

texto = "Fundamentos de Ingenieria de Software"

opcion= 0

print texto

opcion = int(raw\_input("ecriptar= 1 , desencriptar= 2: ") )

abc= ['!','a','b','c','d','e','f','g','h','i','j','k','l','m','n','#','o','p','q','r','s','t','u','v','w','x','y','z','-','&','@','.',';','0','1','2','3','4','5','6','7','8','9','|'] #45

rpas =len(abc) #contamos la cantidad de letras de abc

#funcion para invertir cadena

def invertir(var):

return var[::-1]

#funcion de resta

def restar(x,y):

if x>y:

return x-y

else:

return y-x

#funcion para eliminar los espacios

def sin(txt):

nuevo=""

for x in txt:

if x=='|':

nuevo = nuevo+' '

else:

nuevo = nuevo+x

return nuevo

def espacio(texto):

espacios= ""

for x in texto:

if x==' ':

espacios = espacios+'|'

else:

espacios = espacios+x.lower() #convertimos a minusculas

return espacios

#opcion de encriptacion

if opcion == 1:

print"opcion de encriptacion"

mensaje=""

clave =""

mensaje = raw\_input("introduzca mensaje a ecriptar: ")

clave = raw\_input("introduzca palabra clave: ")

n = len(mensaje) #cuento la cantidad de caracteres

posicion= 0

ab=clave[0]

index = abc.index(ab)

encoding= ""

suma = 0

espacios=""

espacios=espacio(mensaje)

for x in espacios:

for y in range(rpas):

li = 0

if x==abc[y]:

li=y+index

if li <= rpas:

encoding =encoding+abc[li]

else :

suma= restar(rpas,li)

encoding =encoding+abc[suma]+'$'

suma = 0

#print encoding

print "mensaje cifrado: ",invertir(encoding)

print "su clave es:", clave

if opcion == 2:

print"opcion de desencriptacion"

mensaje=""

mensaje= ''

mensaje = raw\_input("Introduzca mensaje Encriptado: ")

clave = raw\_input("Introduzca clave: ")

ab=clave[0] #seleccionamos la letra de la posicion 0

index = abc.index(ab) #usamos nuestra letra la ver en donde cae en la lista de abc

contador = 0

decoding= ''

letra= ''

tx= ''

suma = 0

rango = len(mensaje) #cuento la cantidad de caracteres

w=0

while w < rango:

letra=mensaje[w]

if letra =='$': #cuando esta opcion se cumple restamos 44

incr = w+1

li = 0

tx=mensaje[incr]

posicion= w

for y in range(rpas):

if tx == abc[y]:

li=restar(index,y)

#print index, " - ", y,"=",li

suma=restar(rpas,li)

decoding =decoding+abc[suma]

w+=1

else:

li = 0

tx=mensaje[w]

posicion= w

for s in range(rpas):

if tx == abc[s]:

li= restar(s,index)

decoding =decoding+abc[li]

#print posicion

w+=1

#print decoding

mensaje =invertir(decoding)

#print mensaje

print "mensaje abierto: ",sin(mensaje)

**Prueba de Escritorio**

