Universidad Autónoma de Yucatán

Facultad de Matemáticas

Fundamentos de Programación

Proyecto: Algoritmos de encriptación

1. DESCRIPCIÓN DEL PROYECTO

Este proyecto consiste en elaborar una aplicación de software, utilizando el lenguaje de programación C, que simule un encriptador de mensajes. El programa consiste en crear una interfaz que permita al usuario seleccionar un tipo de cifrado para encriptar un mensaje dado. Posteriormente, desencriptar el mensaje para obtener el mensaje original.

1.1 Reglas y terminología de los algoritmos de encriptación

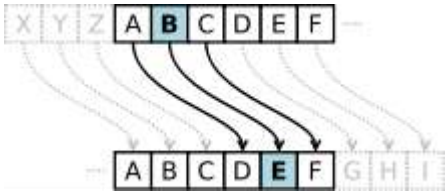
1.- Cifrado por sustitución simple

En criptografía, el cifrado por sustitución es un método de cifrado por el que unidades de texto plano son sustituidas con texto cifrado siguiendo un sistema regular; las "unidades" pueden ser una sola letra (el caso más común), pares de letras, tríos de letras, mezclas de lo anterior, entre otros. El receptor descifra el texto realizando la sustitución inversa.

Los cifrados por sustitución son comparables a los cifrados por transposición. En un cifrado por transposición, las unidades del texto plano son cambiadas usando una ordenación diferente y normalmente bastante compleja, pero las unidades en sí mismas no son modificadas. Por el contrario, en un cifrado por sustitución, las unidades del texto plano mantienen el mismo orden, lo que se cambia son las propias unidades del texto plano.

En el cifrado de sustitución simple una letra en el texto original es reemplazada por otra letra que se encuentra en una posición que está a un número determinado de espacios más adelante en el alfabeto. A esto se lo denomina alfabeto de sustitución. El alfabeto puede ser desplazado o invertido (creando unos cifrados de tipo Cesar y Atbash, respectivamente) o mezclados de una forma compleja, de esta forma se obtiene un alfabeto mezclado o alfabeto sin rango.

Cifrado desplazado: 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. Ejemplo:



Cifrado invertido: establece las parejas de sustitución invirtiendo el orden del alfabeto del texto en claro. Por tanto en castellano la A será sustituida por la Z, la B por la Y, y así sucesivamente. Ejemplo:

Alfabeto plano	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
Alfabeto cifrado	Z	Y	X	W	V	U	T	S	R	Q	P	O	N	M	L	K	J	I	H	G	F	E	D	C	B	A

Con base en lo anterior, un ejemplo de un alfabeto mezclado (desplazado 5 posiciones e invirtiendo) sería el siguiente:

Alfabeto plano	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
Alfabeto Cifrado	_	^	]	\	[	Z	Y	X	W	V	U	T	S	R	Q	P	O	N	M	L	K	J	I	H	G	F

Un mensaje del tipo: HOLA COMO ESTAS
Se cifra como: XQT_]QSQ [ML_M

2.- Cifrado XOR

En criptografía, el cifrado XOR es, como su nombre indica, un algoritmo de cifrado basado en el operador binario XOR:

A	**B**	**A XOR B**
0	0	0
0	1	1
1	0	1
1	1	0

Una cadena de texto puede ser cifrada aplicando el operador de bit XOR sobre cada uno de los caracteres utilizando una clave. Para descifrar la salida, solo hay que volver a aplicar el operador XOR con la misma clave.

Por ejemplo, la cadena “Wiki” (01010111 01101001 01101011 01101001 en 8-bit ASCII) puede ser cifrada con la clave “s” (01110011) de la siguiente manera:

$$\begin{array}{rl} & 01010111\ 01101001\ 01101011\ 01101001 \\ \oplus & 01110011\ 01110011\ 01110011\ 01110011 \\ \hline = & 00100100\ 00011010\ 00011000\ 00011010 \end{array}$$

Mensaje cifrado: $ ^Z ^X ^Z = $ ^Z^X^Z

(Utilizar como referencia el código ASCII del link: http://es.wikipedia.org/wiki/ASCII)

Y viceversa para descifrarlo:

$$\begin{array}{rl} & 00100100\ 00011010\ 00011000\ 00011010 \\ \oplus & 01110011\ 01110011\ 01110011\ 01110011 \\ \hline = & 01010111\ 01101001\ 01101011\ 01101001 \end{array}$$

3.- Cifrado Vigenère:

El cifrado Vigenère es un cifrado basado en diferentes series de caracteres o letras del cifrado César formando estos caracteres una tabla, llamada tabla de Vigenère, que se usa como clave. El cifrado de Vigenère es un cifrado de sustitución simple polialfabético. Este cifrado usa la función lineal de cifrado *f(x)=(x+b) mod n*, donde *x* es el mensaje a cifrar, *b* la clave de cifrado y *n* el número de letras del alfabeto. Ejemplo:

Considerando el alfabeto de 26 letras donde A=0, B=1 , C=2 ... Z=25
En términos matemáticos puede expresarse como:

$$Yi = (Xi + Zi) mod T$$

Donde X_i es el número de ubicación de la una letra del mensaje, Z_i es el número de ubicación de la letra correspondiente de la clave, y la letra T es el total de elementos del alfabeto. Ejemplo:

```
mensaje:      P A R I S  V A U T  B I E N  U N E  M E S S E
clave:        L O U P L  O U P L  O U P L  O U P  L O U P L
Msg. cifrado: A O L X D  J U J E  P C T Y  I H T  X S M H P
```

Comenzando por el lado izquierdo, cuando X_i= “P” le corresponde al número 15 en del alfabeto y Z_i = “L” le corresponde al número 11, y la letra T es el total de números del alfabeto. Entonces la ecuación quedará de la siguiente manera: Yi= (15 + 11) mod 26. El resultado es: 0, donde 0 es igual a A.

Para efectuar este cifrado se usaba la siguiente tabla:

	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
A	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
B	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	A
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	Z	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

4.- Cifrado Run length:

Con esta técnica se puede reducir cualquier secuencia de caracteres cuando el nivel de ocurrencia es de tres o más caracteres iguales consecutivos. Cuando se encuentran 3 o más caracteres iguales consecutivos lo que se hace es sustituir esta secuencia por:

1. Un carácter especial indicador de compresión,
2. El carácter que se comprime,
3. Número que indica la cantidad de caracteres que se comprimen.

Ejemplo de compresión en el emisor

Mensaje:

fghhhhjjerØØØØØ ØØØØtttrrrrrywqad

Una vez realizada la compresión, la cadena resultante será:

fg#**h4**jjer#**Ø9#t3#r5**ywqad

Donde **Ø** representa un carácter blanco y **#** es el carácter especial indicador de compresión.

En el proceso de descompresión, el receptor recorre la cadena de datos que llega a través del canal. Cuando encuentra un carácter especial que indique compresión sabrá que en esa posición se ha realizado una compresión y que el siguiente carácter indica el carácter que ha sido comprimido y a continuación aparece el número que indica cuantos caracteres fueron comprimidos y de esta forma podrá reconstruir la cadena original.

Ejemplo de descompresión en el receptor

La cadena recibida a través del canal es la siguiente:

gt**#r5**juli**#Ø4**jkl**#p3**hj

Una vez realizada la descompresión obtenemos que la cadena original era:

gtrrrrrjuliØØØØ jklpppphj

1.2 Especificaciones del proyecto

La aplicación software a elaborar deberá realizar lo siguiente:

- Debe permitir elegir el tipo de cifrado a utilizar.
- Se debe validar el tipo de mensaje introducido, con base en el alfabeto utilizado.
- Se debe imprimir el mensaje cifrado.
- Debe permitir descifrar e imprimir el mensaje cifrado.
- Si se implementan las cuatro aplicaciones para que trabajen con archivos de texto (entrada y salida) se otorgarán 10 puntos adicionales.

2. FASES DEL PROYECTO.

El proyecto consta de las siguientes fases:

1. Análisis del problema
2. Elaboración de algoritmos en lenguaje natural.
3. Elaboración de Diagramas de Flujo.
4. Codificación.
5. Elaboración del manual de usuario.

2.1 Análisis del problema

Este apartado Consiste en definir el problema y especificar claramente lo que es necesario para resolverlo. Para ello se hay que responder a las siguientes preguntas:

Especificaciones de entrada:
¿Qué datos son de entrada?
¿Cuántos datos se introducirán?
¿Qué datos de entrada son válidos?

Especificaciones de salida:
¿Cuáles son los datos de salida esperados?

¿Qué método produce la salida deseada a partir de los datos de entrada?

En esta fase se debe entregar un documento con las respuestas a las preguntas anteriores.
Nombre del documento: NumEquipo_Analisis_Problema.docx

Primera entrega: jueves 12 de noviembre.

2.2 Elaboración de algoritmos en lenguaje natural.

Este apartado consiste en identificar y redactar en lenguaje natural los algoritmos que se requieren para dar solución al problema identificado y comprendido en la fase anterior.
En esta fase se debe entregar un documento que liste el conjunto de algoritmos necesarios para dar solución al problema planteado. Los algoritmos identificados deberán ser redactados en lenguaje natural.
Nombre del documento: NumEquip_Algoritmos_Lenguaje_Natural.docx

Segunda entrega: jueves 19 de noviembre.

2.3 Elaboración de Diagramas de Flujo.

Este apartado consiste en representar gráficamente los algoritmos en lenguaje natural redactados en la fase anterior.

En esta fase se debe entregar un documento que muestre gráficamente mediante DFDs cada uno de los algoritmos en lenguaje natural previamente redactados.
Nombre del documento: NumEquipo_Algoritmos_DFDs.docx

Segunda entrega: jueves 19 de noviembre.

2.4 Codificación.

Este apartado consiste en traducir los algoritmos en DFDs elaborados en la fase anterior a Lenguaje de Programación C.

En esta fase se irán entregando los avances (en fechas establecidas) de los distintos módulos o programas que representan a cada uno de los algoritmos en DFDs elaborados anteriormente.

Tercera entrega: jueves 26 de noviembre.

2.5 Elaboración del manual de usuario

Este apartado consiste en elaborar un tutorial o manual de usuario que explique el uso de la aplicación software desarrollada.

En esta fase se debe entregar un documento que describa el uso de la aplicación software.
Nombre del documento: NumEquipo_Manual_Usuario.docx

FECHA DE ENTREGA DEL PROYECTO FINAL: LUNES 30 DE NOVIEMBRE.