UNIDAD 4
ACTIVIDAD PRÁCTICA
CADENAS

U-tad
CENTRO UNIVERSITARIO
DE TECNOLOGÍA Y ARTE DIGITAL

Centro adscrito a
Universidad
Camilo José Cela

ÍNDICE

ORGANIZACIÓN DE LA ACTIVIDAD PRÁCTICA

Cadenas	
Tipo de tarea	Individual
Entregables	INSD_IPR2_A1_U4_NombreAlumno_Apellidos.zip

DESCRIPCIÓN DE LA ACTIVIDAD PRÁCTICA

Bienvenidos al mundo del espionaje internacional…! Por motivos confidenciales de un cliente de la industria de la defensa, los alumnos deberán, si deciden aceptar esta misión, escribir un programa para encriptar mensajes en base a un diccionario. Para lograr la misión, deberán seguir un protocolo para transformar una cadena de caracteres en un mensaje encriptado que contiene tanto texto como datos complejos con un separador.

El mensaje a transmitirse será:

```
Mensaje recibido. Entre parentesis, se marcan los datos del agente secreto encubierto con clave, identificador, especialidad y estado: (cactus-123456-francotirador-ACTIVO)
```

Para ello, debemos implementar un protocolo con los siguientes pasos:

1. Se procederá a leer el mensaje ingresado por teclado desde el buffer mediante la función leeLineaDinamica(), la cual se encargará de reservar la memoria dinámica necesaria para la cadena que ingrese el analista de inteligencia. El tamaño de la cadena es desconocido a priori. Por ello, debemos tener cuidado de ajustar el tamaño de la memoria en todo momento.

2. A fin de normalizar los mensajes, se copiará la cadena ingresada por el analista de inteligencia en otra distinta, pero transformando todas las primeras letras de cada palabra por mayúsculas y dejando las demás en minúsculas a través de la función cambiaMayúsculas() a partir de la cadena leída, dejando el resto de caracteres que no sean letras sin modificarse (números y resto de caracteres).

3. Se procederá a implementar 2 diccionarios de transformaciones (uno para caracteres y otro para números) que encriptan el mensaje (ver tablas abajo) a través de la función encriptaMensaje() a partir de la cadena transformada. Los caracteres que no sean letras o números, quedan de la misma manera en que se encuentran. Se deberá imprimir la cadena encriptada. Sólo se podrán utilizar los caracteres de la tabla, por lo que no se permiten acentos ni la letra ñ.

Letra	Cambia por:	Letra	Cambia por:	Letra	Cambia por:	Número	Cambia por:
A	W	K	K	U	Y	0	7
B	X	L	B	V	U	1	8
C	A	M	J	W	T	2	3
D	H	N	D	X	C	3	5
E	Q	O	G	Y	Z	4	4
F	F	P	M	Z	S	5	6
G	P	Q	E			6	2
H	I	R	L			7	9
I	R	S	N			8	1
J	V	T	O			9	0

4. Para continuar, se deberá extraer la cadena con los datos complejos que se encuentran dentro de los paréntesis a través de la función encuentraDatos() a partir de la cadena encriptada, y guardará dicha cadena por separado. Se deberá imprimir la cadena encriptada con los datos.

5. Se deberán separar los datos complejos en los distintos campos a partir de la cadena encriptada con los datos. Aquí tenemos la libertad de usar cualquiera de los métodos y funciones del fichero de cabecera <string.h> y se deberá presentar los datos encriptados por separado.

6. Finalmente, se procederá a decodificar los campos del punto anterior a través de la función decodificaDatos() a partir de las cadenas separadas. Se presentarán los datos decodificados por separado (como se muestra el ejemplo de ejecución).

7. Todas las funciones y cadenas deberán estar implementadas con punteros o punteros a char, según corresponda.

8. Se deberá imprimir cada paso del protocolo para verificar que las funciones son correctas.

9. Los prototipos de las funciones a utilizar serán:

La lectura de la cadena del buffer se realizará con la función:

```
char* leeLineaDinamica();
```

La transformación en mayúsculas de la cadena leída se realizará con la función:

```
char * cambiaMayusculas(char * cadena_leida);
```

La codificación de la cadena en mayúsculas se realizará con la función:

```
char * encriptaMensaje(char * cadena_mayusculas);
```

La extracción de la cadena con los datos se realizará con la función:

```
char * encuentraDatos(char * cadena_encriptada);
```

La decodificación de la cadena con los datos (cada campo de datos) se realizará con la función:

```
char * decodificaDatos(char * cadena_encriptada_datos);
```

RÚBRICA DE CORRECCIÓN

La rúbrica para corregir el ejercicio seguirá los criterios listados a continuación, que tienen distintos pesos respecto al total de la nota.

Criterios	Excelente	Satisfactorio	No satisfactorio	Insuficiente
Conocimientos de cadenas en profundidad usando las funciones correctas. (Máximo 5 puntos)	Conoce perfectamente la gestión de cadenas avanzadas en C. (de 4 a 5 puntos)	Conoce correctamente la gestión de cadenas avanzadas en C. (de 3 a 4 puntos)	Conoce algunos aspectos de la gestión de cadenas avanzadas en C. (de 1,5 a 3 puntos)	No conoce la gestión de cadenas avanzadas en C. (de 0 a 1,5 puntos)
Conocimientos sobre las funciones de <string.h> en C, integrándolo en el código. (Máximo 3 puntos)	Conoce perfectamente cómo usar la funciones en C. (2 a 3 puntos)	Conoce correctamente cómo usar la funciones en C. (de 1 a 2 puntos)	Conoce algunos aspectos sobre cómo usar la funciones en C. (de 0.5 a 1.0 punto)	No tiene los conocimientos suficientes sobre cómo usar la funciones en C. (de 0 a 0.5 puntos)

Conocimientos sobre uso de memoria dinámica y punteros para cadenas en C, integrándolo en el código. (Máximo 2 puntos)	Conoce perfectamente cómo usar memoria dinámica y punteros para cadenas en C. (1,5 a 2 puntos)	Conoce correctamente cómo usar memoria dinámica y punteros para cadenas en C. (de 1 a 1,5 puntos)	Conoce algunos aspectos sobre cómo usar memoria dinámica y punteros para cadenas en C. (de 0.5 a 1.0 punto)	No tiene los conocimientos suficientes de cómo usar memoria dinámica y punteros para cadenas en C. (de 0 a 0.5 puntos)