LABORATORIO 9
CADENAS DE CARACTERES - ARRAYS

1 OBJETIVOS

Al finalizar esta actividad, serás capaz de:

- Trabajar adecuadamente con las cadenas de caracteres, también llamadas *strings*.
- Utilizar funciones que manejen *strings*.

2 MOTIVACIÓN

- Existen multitud de problemas que no se pueden solucionar si no es con el uso de estructuras de datos como las cadenas de caracteres.
- En particular, nos referimos a problemas en los que es necesario almacenar, manejar y escribir series de caracteres (frases, listas de palabras, códigos, etc.).

2.1 Las cadenas de caracteres en C

- Almacenan en una misma estructura un número determinado de valores de tipo carácter. En general su uso es como el de los *arrays*.
- Para trabajar con ellos se utiliza el carácter especial de fin de cadena ('\0') que se ubica en la última posición ocupada del *string*.

3 EJERCICIOS

3.1 Ejercicio 1

3.1.1 Objetivo:

El objetivo del ejercicio 1 es leer un *string* y recorrerlo para contar el número de palabras.

3.1.2 Ayuda para el enunciado 1:

Para contar palabras nos vale con identificar dónde empieza cada una de ellas, de modo que al detectar un espacio en blanco seguido de una letra podemos asegurar que hemos encontrado una nueva palabra. Atención a la primera palabra del *string*, pues puede ser un caso especial.
De modo análogo, podrían contarse los finales de palabra, es decir, el número de letras seguidas por un blanco. En este caso, la última palabra de la cadena sería el caso especial.

3.1.3 Enunciado 1:

Realizar un programa que lea una cadena de como máximo 250 caracteres e indique de cuántas palabras está compuesta. Cada palabra puede estar separada de la siguiente por uno o más espacios en blanco; asimismo, puede ocurrir que antes de la primera palabra, o después de la última, haya uno o más blancos.

Ejemplo:

Cad		'H'	'o'	'l'	'a'					'g'	'e'	'n'	't'	'e'				'\0'	...

```
La cadena de caracteres tiene 2 palabras.
```

3.2 Ejercicio 2

3.2.1 Objetivo:

Este ejercicio ayudará a trabajar diferentes aspectos de los *strings*. Requiere la realización de búsquedas y filtrados (lo que exige un adecuado uso de los índices para el acceso a las posiciones del *string*) y, opcionalmente, la definición de funciones.

3.2.2 Ayuda para el enunciado 2:

Utilizar como punto de partida el ejercicio 1, incluir si se desea una función que, dados un *string* y un carácter, determine el número de apariciones del carácter en la cadena.

3.2.3 Enunciado 2:

Hacer un programa que pida una cadena de caracteres (*cad1*) formada exclusivamente por letras y espacios en blanco (250 caracteres como máximo), y visualice las iniciales de cada palabra de dicha cadena, sabiendo que las palabras pueden estar separadas por uno o más blancos. A continuación deberá generar otra cadena (*cad2*), que resulte de eliminar en cad1 todos los blancos y convertir todas sus letras a minúsculas. Por último, indicar cuál es la letra que más aparece en cad2.

Ejemplo:

Cad1		'H'	'o'	'l'	'a'					'g'	'e'	'n'	't'	'e'				'\0'	...

```
Las iniciales de las palabras son: H g.
```

Cad2	'h'	'o'	'l'	'a'	'g'	'e'	'n'	't'	'e'	'\0'	...

```
La letra que más aparece en cad2 es la 'e', que aparece 2
veces.
```

3.3 Ejercicio 3

3.3.1 Objetivo:

El objetivo del ejercicio 3 es seguir trabajando con *strings*, en este caso realizando algunas modificaciones sobre una cadena de caracteres.

3.3.2 Ayuda para el enunciado 3:

Aplicar los cambios sobre la cadena de uno en uno: haz uno y comprueba que funciona. Debes tener cuidado con las últimas posiciones del *array*; por ejemplo, en el caso de que la posición 249 tenga una 'g' no tiene sentido preguntar si la siguiente posición tiene una 'u', ya que la posición 250 no está definida.

3.3.3 Enunciado 3:

Escribir un programa que pida una frase y produzca ciertas faltas de ortografía. Para ello vamos a realizar los siguientes cambios:

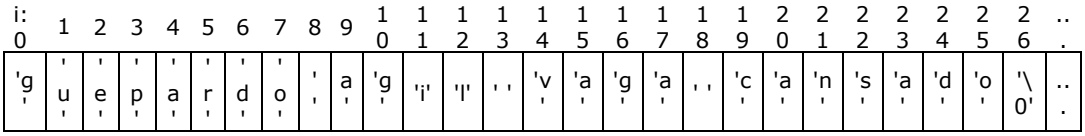
- Cambiar la 'c' por 'k' siempre que esté seguida por 'a', 'o', ó 'u'.
- Cambiar 'v' por 'b', y 'b' por 'v'.
- Eliminar la 'u' en los casos "gue" y "gui".
- Insertar 'h' si la palabra empieza por vocal.

Se supone que la frase inicial está en minúsculas y sin acentos (no hay que comprobar nada), y que puede contener un máximo de 250 caracteres, incluyendo el fin de cadena.

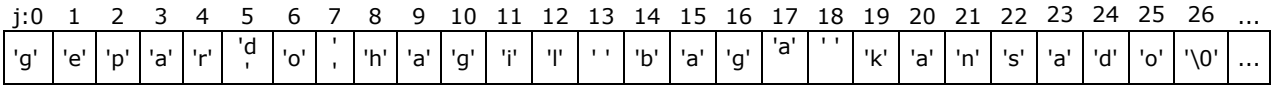
Ejemplo:

```
DIME UNA FRASE: guepardo agil vaga cansado
TRANSFORMADA: gepardo hagil baga kansado
```

Si la frase inicial es **cad1** :



La transformación produce una nueva cadena **cad2** :



3.4 Ejercicio 4

3.4.1 Objetivo:

El objetivo del ejercicio 4 es abordar un problema relativamente complejo de *strings*, simplificándolo mediante el uso de funciones.

3.4.2 Ayuda para el enunciado 4:

Para resolver este problema se sugiere seguir estos pasos:

- Convertir la cadena de entrada a otro *string* en el que se han eliminado los espacios en blanco. Usar una función para ello.
- Convertir el *string* sin blancos a otra cadena en la que las mayúsculas se han convertido en minúsculas. Usar una función para ello.
- Decidir si la cadena resultante (que sólo tendrá letras minúsculas hasta la posición del carácter '\0') es la misma leída de izquierda a derecha y viceversa. Usar una función para ello.

3.4.3 Enunciado 4

Hacer un programa que lea por teclado una cadena de caracteres y decida si se trata de un palíndromo o no.

NOTA: Un palíndromo es una cadena que se lee igual de izquierda a derecha que de derecha a izquierda.

Ejemplo: "Dabale arroz a la zorra el abad".