

Práctica 7. Cadenas.

Objetivos de la práctica:

- Adquirir la capacidad de programar en lenguaje C.
- Adquirir la capacidad de resolución de problemas sencillos.
- Comprender el uso y funcionamiento de los punteros.
- Comprender cómo se trabaja con cadenas.

1. Cadenas

Las cadenas de caracteres en C son vectores de datos de tipo `char`. Para declarar una cadena podemos hacer:

```
char cadena[100];
```

O bien, usando punteros:

```
char *cadena;
```

Y luego reservaremos espacio con `malloc` como con el resto de tipos de datos:

```
cadena=(char*)malloc(sizeof(char)*longitud);
```

Todas las cadenas acaban en el carácter fin de cadena `'\0'`

Para leer una cadena podemos usar:

- Si queremos leer solo una palabra (lee hasta el primer espacio en blanco):

```
scanf("%s", cadena);
```
- Si queremos leer frases con espacios en blancos: `fgets(cadena, 100, stdin);`

Para mostrar una cadena por pantalla:

- Se podría hacer con un bucle que imprima cada letra mientras no se llegue al fin de la cadena (carácter `'\0'`):

```
for(i=0;cadena[i]!='\0';i++)  
    printf("%c", cadena[i]);
```

- Si no queremos hacer un bucle y queremos imprimir toda la cadena, podemos usar: `printf("%s", cadena);`

Hay una librería que es la `string.h` que proporciona muchas funciones interesantes cuando se trabaja con cadenas (más información en las transparencias de Cadenas del moodle):

- `strcpy`: Copia una cadena en otra

- **strncpy**: Copia n caracteres de una cadena en otra
- **strcmp**: Dice si dos cadenas son iguales
- **strcat**: Concatena una cadena con otra (la añade al final)
- **strlen**: Da el número de caracteres de una cadena, sin contar el `\0`
- **strstr**: Busca si una cadena está en otra
- **strtok**: Trocea una cadena

2. Ejercicios propuestos

2.1. Ejercicio 1

Realice un programa `copia.c` en lenguaje C que lea una cadena de máximo 100 caracteres e imprima por pantalla:

- su longitud. Para ello usar la función **strlen**.
- si es igual a la cadena **compara**. Para ello usa la función **strcmp**.
- concaténale la cadena **añadida**. Para ello usa la función **strcat**.

2.2. Ejercicio 2

Realice un programa `cadena.c` en lenguaje C que lea una cadena de tantos caracteres como indique el usuario y llame a una función **quitaespacios** que elimine los espacios en blanco de la cadena y la guarde en otra cadena. La función **main** imprimirá la cadena resultante. La definición de la función será así: *`void quitaespacios(char *cadena, char *cadenasin)`*

- Cuando trabajamos con punteros, podemos acceder a sus posiciones como `puntero[i]`, al igual que hemos hecho con los vectores.

2.3. Ejercicio 3

Realice un programa `cadena.c` en lenguaje C que lea una cadena de máximo 100 caracteres y llame a una función **invierte** que invierte la cadena, guardándola en otra cadena que también se le pasa como argumento. La función **main** imprimirá la cadena resultante. La definición de la función será así: *`void invierte(char *cadenaOrigen, char *cadenaDestino)`*.