

VECTOR DE CARACTERES



Vector de caracteres

- En C, los vectores de caracteres tienen características únicas en comparación a vectores de otro tipo.
- A los vectores de caracteres que terminan con el caracter nulo ('\\0') se los llaman *cadena* o *strings*.
- El carácter '\\0' es un valor especial que se utiliza para indicar el fin de una cadena.

Vector de caracteres

- Un vector de caracteres puede ser inicializado utilizando un string.

- Por ejemplo

```
char palabra[ ] = "Cadena";
```

inicializa el vector **palabra** con las letras de la palabra "Cadena".

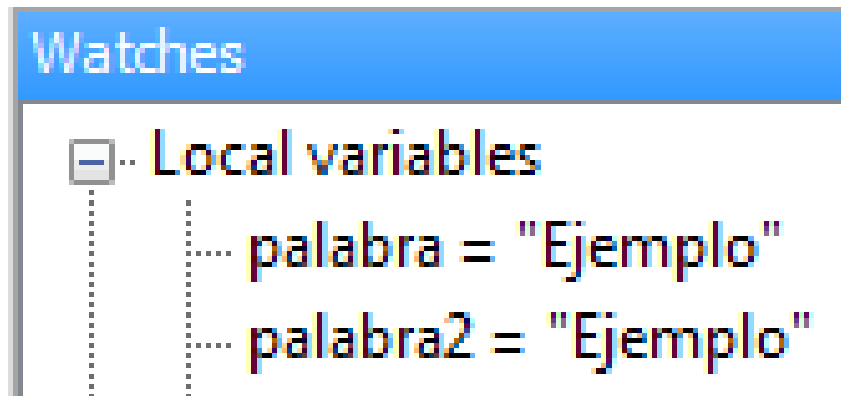
El compilador automáticamente determina la dimensión física del vector. En este caso es 7, ya que se incluye '\0'.

Vector de caracteres

- Las siguientes declaraciones son equivalentes

```
char palabra[ ] = "Ejemplo";
```

```
char palabra2[ ] = { 'E', 'j', 'e', 'm', 'p', 'l', 'o', '\0' } ;
```



Como si fueran
dos strings !

Vector de caracteres

- También es posible indicar la dimensión física del vector en la inicialización.
- Por ejemplo
 - `char palabra[10] = "Cadena";`
- ¿Qué sucede con las celdas no inicializadas?
 - ▣ Se inicializan en 0 (en ASCII: 0 es el carácter nulo).
- ¿Qué sucede si la cantidad de caracteres del string supera la dimensión física especificada?
 - ▣ El compilador nos avisa.

Imprimiendo un vector de caracteres

```
char texto[]="Texto de 23 caracteres.";
printf("%s\n", texto);
printf("Si! Tiene %c%c caracteres!",
      texto[9], texto[10]);
```

"%s" despliega el contenido de un vector de caracteres comenzando por el elemento 0 hasta llegar al carácter nulo.

Imprimiendo un vector de caracteres

```
char texto[]="Texto de 23 caracteres.";

printf("%s\n", texto);

printf("Si! Tiene %c%c caracteres!",
      texto[9], texto[10]);
```

Utilice **%c** para mostrar los caracteres en forma individual

Leyendo un vector de caracteres

- Podemos utilizar **scanf** para ingresar un string por teclado

```
char palabra3[20];  
scanf("%s", palabra3);
```

¿Cuál es la longitud máxima que puede tener el string leído? nulo.

Note que no se utiliza &

Se utiliza %s para leer una secuencia de caracteres **hasta encontrar el primer blanco** (o el fin de línea)


```
/*Strings como vectores de caracteres */
```

```
#include <stdio.h>
```

```
int main()
```

¿Cuántos elementos tiene cada arreglo?

```
{    char string1[20],  
        string2[] = "Primer ejemplo";
```

```
    int i;
```

```
    printf("Ingrese un string : ");
```

```
    scanf("%s", string1);
```

```
    printf("string1 es %s \n"
```

```
           "string2 es %s \n"
```

```
           "String1 con blancos es ",
```

```
           string1, string2);
```

```
    for (i=0; string1[i]!='\0'; i++)
```

```
        printf("%c ", string1[i]);
```

```
    return 0;
```

```
}
```

Strings.c

```

/*Strings como vectores de caracteres */
#include <stdio.h>
int main()
{
    char string1[20];
    string2[] ¿Está bien escrito o tiene errores de
    int i;          sintaxis ?

    printf("Ingrese un string : ");
    scanf("%s", string1);

    printf("string1 es %s \n"
           "string2 es %s \n"
           "String1 con blancos es ",
           string1, string2);

    for (i=0; string1[i]!='\0'; i++)
        printf("%c ", string1[i]);

    return 0;
}

```

Strings.c

```
/*Strings como vectores de caracteres */
```

```
#include <stdio.h>
```

Strings.c

```
int main()
```

```
{    char string1[20],
```

```
        string2[] = "Primer ejemplo";
```

```
    int i;
```

```
    printf("Ingrese un string : ");
```

```
    scanf("%s", string1);
```

```
    printf("string1 es %s \n"
```

```
        "string2 es %s \n"
```

```
        "String1 con blancos es ",
```

```
        string1, string2);
```

```
    for (i=0; string1[i]!='\0'; i++)
```

```
        printf("%c ", string1[i]);
```

```
    return 0;
```

Recorre el string (vector de caracteres)
hasta alcanzar el '\0'

```
}
```

Ejecución del programa anterior

```
Ingrese un string : caminando por el parque
string1 es caminando
string2 es Primer ejemplo
String1 con blancos es c a m i n a n d o
```

- Note que el `scanf` procesa los caracteres hasta encontrar el primer blanco (o el fin de línea).
En la salida anterior se observa que sólo guardó en `string1` la primera palabra ingresada.
- También es responsabilidad del programador que haya lugar suficiente para almacenar el `'\0'`.
- ¿Qué pasa con el programa anterior si el `'\0'` no está?

Funciones para cadenas de caracteres

<string.h>

- **strlen(c1):** Retorna el número de caracteres de la cadena **c1** hasta el carácter nulo (el cual no se incluye).

- **Ejemplo**

```
char palabra[50]="strlen";  
printf("Longitud: %d",strlen(palabra));
```

Funciones para cadenas de caracteres

<string.h>

- **strcpy(c1, c2):** Copia la cadena **c2** en la cadena **c1**. La cadena **c1** debe ser lo suficientemente grande como para almacenar la cadena **c2** y su carácter de terminación NULL (que también se copia).
- **Ejemplo**

```
char copia[50];  
char linea[] = "Esto es un string";  
  
strcpy(copia, linea);  
printf("%s", copia);
```

Esto es un string

Funciones para cadenas de caracteres

<string.h>

- **strcat(c1, c2)** : Agrega la cadena **c2** al arreglo **c1**.
El primer carácter de **c2** sobrescribe el carácter de terminación NULL de **c1**.

- **Ejemplo**

```
char linea1[50] = "Esto es un string";  
char linea2[] = ". Preguntas?";  
  
strcat(linea1, linea2);  
printf("%s", linea1);
```

Esto es un string. Preguntas?

Funciones para cadenas de caracteres

<string.h>

- **strcmp(c1, c2)** : Compara **c1** con **c2** y devuelve

$$\text{strcmp}(c1, c2) = \begin{cases} < 0 & \text{si } c1 < c2 \\ 0 & \text{si } c1 = c2 \\ > 0 & \text{si } c1 > c2 \end{cases}$$

- **Ejemplo**

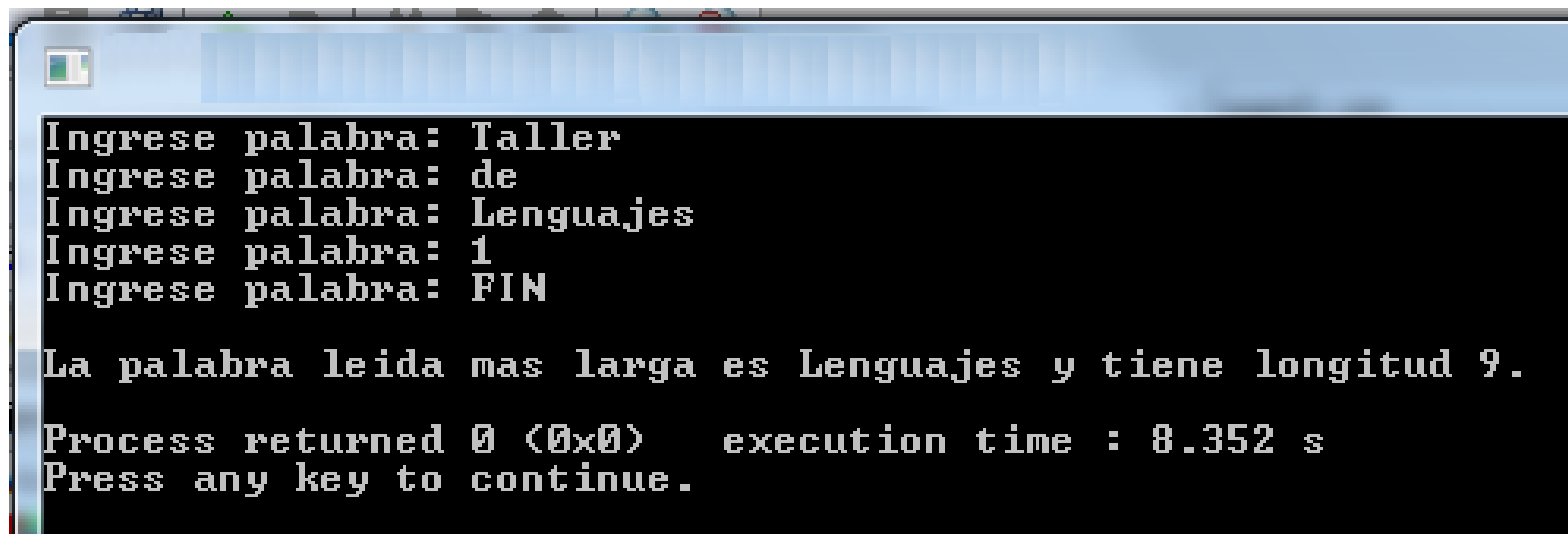
```
char texto1[] = "Función strcmp",  
      texto2[] = "Preguntas?",  
      texto3[] = "función strcmp",  
      texto4[] = "Esta terminado!";
```

Imprime
-1 -1 1

```
printf("%d %d %d\n", strcmp(texto1, texto2),  
        strcmp(texto1, texto3),  
        strcmp(texto1, texto4));
```


Ejercicio a resolver

- Escriba un programa que lea palabras hasta encontrar la palabra 'FIN' e informe cual es la palabra de longitud máxima.



```
Ingrese palabra: Taller
Ingrese palabra: de
Ingrese palabra: Lenguajes
Ingrese palabra: 1
Ingrese palabra: FIN

La palabra leida mas larga es Lenguajes y tiene longitud 9.

Process returned 0 (0x0)    execution time : 8.352 s
Press any key to continue.
```