




Cadenas de caracteres y Strings en C++

Programación 1 - 2do Cuatrimestre 2024

EQUIPO DOCENTE

Profesor:	Angel Simon
Profesor:	Brian Esteban Lara Campos
JTP:	Ariel Sebastian Tapia
JTP:	Verónica Carbonari
JTP:	Daniela F. Pinto (autora del contenido)

Caracteres en C++

VARIABLE SIMPLE	VARIABLES COMPLEJAS	
char	CADENA DE CARACTERES	STRING
 <p>Único carácter.</p>	 <p>Capacidad definida.</p>	 <p>Capacidad indefinida.</p>

Cadenas de caracteres

Cadena de caracteres

- Es un array unidimensional que guarda elementos de tipo char.
- Al igual que el resto de los arrays estáticos, tiene un tamaño constante, que se define al momento de declararse la variable.
- Es soportado tanto por C++ como por C de forma nativa.

String

- Es una clase proporcionada por el espacio de nombres std, que guarda elementos de tipo char en un array dinámico.
- Tiene un tamaño variable, con capacidad máxima indefinida (no depende intrínsecamente del lenguaje).
- Sólo es soportado por C++. C no soporta de forma nativa ni clases ni estructuras dinámicas.


Cadenas de caracteres

Algunas características de las cadenas de caracteres

Comunes a todos los vectores

- Capacidad predefinida y constante.
- Sus elementos se ordenan de forma consecutiva.
- Tienen indexación basada en 0 (cero).

Propias de las cadenas de caracteres

- Son arrays de caracteres finalizados con un carácter nulo ('\0').
- Es posible la manipulación de texto de longitud variable, dentro de la capacidad preestablecida según el tamaño del array.
- Es posible acceder al contenido y asignar valores sin necesidad de recorrer cada posición con un bucle.
-  La longitud del texto NO puede superar la capacidad del vector -1.

Cadenas de caracteres: Asignación de valores

Formas de asignar valores a una variable

1. Inicialización → Ej.: `int mi_variable = 2;`
 2. Asignación luego de la declaración:
 - 2.a - Ingreso por teclado → Ej.: `cin >> mi_variable;`
 - 2.b - Asignación desde el código
 - Asignación directa → Ej.: `mi_variable = 1;`
 - Asignación a partir de otra variable → Ej.: `mi_variable = mi_otra_variable;`
 - Uso de funciones específicas.
-

Cadenas de caracteres: Asignación de valores

1 - Inicialización

EJEMPLOS:

```
char ejemplo1[15] = { 'h', 'o', 'l', 'a', ' ', 'm', 'u', 'n', 'd', 'o' };
```

```
char ejemplo2[15] = { 'h', 'o', 'l', 'a', ' ', 'm', 'u', 'n', 'd', 'o', '\0' };
```

```
char ejemplo3[15] = "hola mundo";
```

SALIDA POR PANTALLA:

```
cout << ejemplo1; → hola mundo
```

```
cout << ejemplo2; → hola mundo
```

```
cout << ejemplo3; → hola mundo
```

DISTRIBUCIÓN DE CARACTERES EN EL VECTOR:

h	o	l	a		m	u	n	d	o	\0	... valores aleatorios (basura)
---	---	---	---	--	---	---	---	---	---	----	---------------------------------

Cadenas de caracteres: Asignación de valores

2.a - Ingreso por teclado

I) `cin >> ejemplo1;`

😬 Es una operación admitida por C++, pero no admite el ingreso de texto con espacios.

II) `cin.getline(ejemplo2 , 15);`

- Primer parámetro: mi cadena

- Segundo parámetro: capacidad máxima del texto a ingresar.

😊 Admite el ingreso de texto con espacios.

□ Tanto “`cin >> vector`” como “`cin.getline(vector, n);`” son funciones de la biblioteca `iostream`, y pertenecen al espacio de nombres `std`.

Cadenas de caracteres: Asignación de valores

2.b - Asignación desde el código

✗ `mi_cadena1 = { 'h', 'o', 'l', 'a', ' ', 'm', 'u', 'n', 'd', 'o' };`

✗ `mi_cadena2 = { 'h', 'o', 'l', 'a', ' ', 'm', 'u', 'n', 'd', 'o', '\0' };`

Al igual que para el resto de los arreglos, la asignación con llaves sólo es posible en la inicialización.

✗ `mi_cadena3 = "hola mundo";`

La asignación directa de texto entre comillas dobles, también es posible sólo en la inicialización.

✗ `mi_cadena1 = mi_cadena2;`

Tampoco es posible la asignación a partir de una variable con el operador "=".

Cadenas de caracteres: Asignación de valores

2.b - Asignación desde el código

Opciones de asignación (ambas válidas):



Recorrer el array con un bucle y asignar caracter por caracter a cada posición de la cadena.



Usar los métodos provistos por la biblioteca `string.h` (C/C++) o `cstring` (C++).

Cadenas de caracteres: Asignación de valores

2.b - Asignación desde el código: Uso de funciones específicas

Uso del método strcpy() de la biblioteca string.h:

- Inclusión de la biblioteca: `#include < string.h >`
- Llamado al método: `strcpy(cadena_de_destino, cadena_de_origen);`

→ Copia el contenido de `cadena_de_origen` en la variable `cadena_de_destino`.

Ejemplo 1:

```
#include < string.h >

strcpy( cadena_de_destino, "hola mundo" );
```

Ejemplo 2:

```
#include < string.h >

char cadena_de_destino[20];
char cadena_de_origen[20] = "Hola mundo";
strcpy( cadena_de_destino, cadena_de_origen );
```

Cadenas de caracteres: Asignación de valores

Otros métodos de string.h

char strcat(): *strcat(cadena_destino, cadena_fuente);*

→ Concatena *cadena_destino* + *cadena_fuente*.

int strcmp(): *strcmp(cadena1, cadena2);*

→ Compara alfabéticamente la cadena *cadena1* contra la *cadena2* y devuelve:

0 si *cadena1* = *cadena2*

< 0 si *cadena1* < *cadena2*

> 0 si *cadena1* > *cadena2*

int strcmpi(): *strcmpi(cadena1, cadena2);*

→ Igual que *strcmp()*, pero sin distinguir entre mayúsculas y minúsculas.

int strlen(): *strlen(mi_cadena);*

→ Devuelve la longitud de la cadena hasta el carácter terminador, sin incluirlo.

Clase String

Strings en C++

- String es una clase proporcionada por el espacio de nombres std, que guarda elementos de tipo char en un array dinámico.
- Tiene un tamaño variable, con capacidad máxima indefinida (no depende intrínsecamente del lenguaje).
- Sólo es soportado por C++, ya que C no soporta clases ni estructuras dinámicas de forma nativa.
- Para poder operar con strings en C++, es necesario importar la biblioteca *string*:

```
#include <string>
```

Clase String

Strings en C++

Un string es un array de caracteres con comportamiento especializado, gracias a una serie de métodos propios, provistos por la biblioteca estándar de C++ con la finalidad de facilitar la manipulación de texto.

Para poder hacer uso de las propiedades de la clase string, es necesario incluir la biblioteca <string> y trabajar con el espacio de nombre std:

```
#include < string >
```

```
using namespace std;
```

Clase String

Asignación de valores a una variable string: Inicialización

EJEMPLO:

`string` mi_texto = "hola mundo"; → uso las comillas dobles para hacer la asignación.

- Puedo declarar mi variable string de forma similar a cualquier variable simple (aunque no lo sea).
- No necesito predefinir el tamaño al momento de declararla.

REPRESENTACIÓN DE MI VARIABLE STRING EN UN ARRAY DE CHAR:

h	o	l	a		m	u	n	d	o	\0
---	---	---	---	--	---	---	---	---	---	----

Clase String

Asignación de valores a una variable string: Ingreso por teclado

EJEMPLOS:

`string mi_texto;`

Ejemplo I) `cin >> mi_texto;`

😬 Es una operación admitida por C++, pero no admite el ingreso de texto con espacios, al igual que en el caso de las cadenas de caracteres .

Ejemplo II) `getline(cin, mi_texto);`

- Primer parámetro: operador de extracción `cin`.
- Segundo parámetro: mi variable string.

😊 Admite el ingreso de texto con espacios.

Clase String

Asignación de valores a una variable string: Con operador de asignación

EJEMPLOS:

```
string mi_texto;
```

```
string mi_otro_texto;
```

Ejemplo I)

```
mi_texto = "mi texto asignado de forma directa.";
```



¡Haciendo uso de la variable string, es posible asignar texto de forma directa!

Ejemplo II)

```
mi_otro_texto = mi_texto;
```



También es posible la asignación de valor a partir de otra variable con el operador de asignación.

Clase String

Manipulación de variables string

```
string mi_variable = "Hola";  
string mi_variable_2 = "Mundo";
```

→ Concatenación:

```
string mi_variable_3 = mi_variable + mi_variable_2 ; // genera una nueva cadena: HolaMundo
```

→ Comparación alfabética:

```
bool resultado1 = (mi_variable == mi_variable_2);  
bool resultado2 = (mi_variable > mi_variable_2);  
bool resultado3 = (mi_variable < mi_variable_2);
```

→ Consultar la longitud del texto:

```
int longitud = mi_variable.length();
```

String a cadena de caracteres

conversión de string a cadena de caracteres

TRASPASO DEL VALOR DE UNA VARIABLE STRING A UNA CADENA DE CARACTERES:

- Se puede lograr con el método `std::strcpy(mi_cadena, fuente_de_texto);`.
- Recordemos que este método nos permite copiar en una variable de tipo cadena de caracteres, una cadena de texto literal u otra cadena de caracteres.
- ⚠ No podemos pasar como segundo parámetro una variable de tipo string. Sin embargo, disponemos del método `c_str()` de la biblioteca `<string>`, que nos permite convertir una variable string en una cadena de caracteres.

EJEMPLPO:

```
string mi_variable_string = "Hola Mundo";  
char mi_variable_cadena [15];
```

```
strcpy(mi_variable_cadena, mi_variable_string.c_str());
```

String a cadena de caracteres

conversión de string a cadena de caracteres

... VEAMOS EL CÓDIGO COMPLETO:

```
#include <iostream>
#include <cstring>
#include <string>
using namespace std;

int main (){
    string mi_variable_string = "Hola Mundo";
    char mi_variable_cadena [15];

    strcpy(mi_variable_cadena, mi_variable_string.c_str());
    cout<<mi_variable_cadena;
    return 0;
}
```

SALIDA POR PANTALLA:

>> Hola Mundo

Cadenas de caracteres en C/C++

Bibliografía

- Joyanes Aguilar, Sánchez García, Zahonero Martínez (2007). ESTRUCTURA DE DATOS EN C++. McGRAW-HILL/INTERAMERICANA DE ESPAÑA, S. A. U. España.
- Joyanes Aguilar, L. (2006). Programación en C++: algoritmos, estructuras de datos y objetos: (2 ed.). Madrid etc, Spain: McGraw-Hill España.