



Grado en Ingeniería Información

PROGRAMACIÓN I

Sesión 5

Curso 2022-2023

Marta N. Gómez (mgomezper@nebrija.es)

Tipos de datos estructurados.

- `std::string`
- `std::array`
- `std::vector`
- `std::set`



Tipo de datos string (std::string)

- El tipo *string* es una cadena de caracteres o cadena de texto formada por el tipo de dato simple *char*.
- Necesita incluir la biblioteca de C++ *string*:

#include <**string**>

- Ejemplo:

```
std::string mi_nom = "Marta";    // C++ “antiguo”
```

```
std::string mi_nom { "Marta" };  // C++ “moderno”
```

Tipo de datos **string** (std::string)

- El tipo *string* se maneja como cualquier otro tipo de datos:
 - Modificar su valor con el **operador de asignación**.
 - Hacer que sea una **constante**.
- Ejemplo:

```
std::string mi_nom1, mi_nom2 {"Vega"};
```

```
const std::string nom1 {Nuria}, nom2 = "Henar ";
```

```
mi_nom1 = "Marta";
```

Tipo de datos string (std::string)

- El tipo *string* es una **clase**:
 - Sus variables son **objetos**.
 - Contiene **miembros**: **atributos** (variables) y **métodos** (funciones).
 - El operador **punto** (.) permite el acceso a los **miembros** de la clase:

nom_var.miembro

Tipo de datos string (std::string) - Funciones

size Devuelve la longitud del *string*.

length Devuelve la longitud del *string*.

```
#include <iostream>
```

```
#include <string>
```

```
using namespace std;
```

```
int main(){
```

```
    string nombre{"Marta"};
```

```
    cout << "\n\n\tEl nombre es: " << nombre << endl;
```

```
    cout << "\tSu dimension es: " << ": " << nombre.size() << endl;
```

```
    cout << "\tSu longitud es: " << ": " << nombre.length() << endl;
```

```
    return 0;
```

```
}
```

```
El nombre es: Marta
Su dimension es: : 5
Su longitud es: : 5

Press <RETURN> to close this window...
```

Tipo de datos string (std::string) - Funciones

resize Permite modificar el tamaño del *string*.

- Si el tamaño es **mayor**, hay que indicar el **valor de relleno**.
- Si el tamaño es **menor**, se **trunca la cadena**.

```
#include <iostream>
```

```
#include <string>
```

```
using namespace std;
```

```
int main(){
```

```
    string nombre{"Marta"};
```

```
    cout << "\n\n\tEl nombre es: " << nombre << endl;
```

```
    nombre.resize(10, '*');           // Tamaño mayor
```

```
    cout << "\tAhora es: " << ": " << nombre << endl;
```

```
    nombre.resize(4);                 // Tamaño menor
```

```
    cout << "\tY ahora es: " << ": " << nombre << endl;
```

```
    return 0;
```

```
}
```

```
El nombre es: Marta
Ahora es: : Marta*****
Y ahora es: : Mart

Press <RETURN> to close this window...
```

Tipo de datos string (std::string) - Funciones

empty Comprueba si el *string* está vacío. Devuelve *true* si la cadena está vacía y *false* en otro caso.

```
#include <iostream>
```

```
#include <string>
```

```
using namespace std;
```

```
int main(){
```

```
    string nombre{"Marta"};
```

```
    bool esVacia = nombre.empty();
```

```
    cout << "\n\n\tEl nombre esta: " << esVacia;
```

```
    nombre.resize(0);
```

```
    esVacia = nombre.empty();
```

```
    cout << "\n\n\tEl nombre esta: " << esVacia << endl;
```

```
    return 0;
```

```
}
```

```
El nombre esta: 0

El nombre esta: 1

Press <RETURN> to close this window...
```

// false (0)

// true (1)

Tipo de datos string (std::string) - Funciones

clear Borra el contenido del *string*, queda vacía.

```
#include <iostream>
```

```
#include <string>
```

```
using namespace std;
```

```
int main(){
```

```
    string nombre{"Marta"};
```

```
    bool esVacia = nombre.empty();
```

```
    cout << "\n\n\tEl nombre esta: " << esVacia;
```

```
    nombre.clear();
```

```
    esVacia = nombre.empty();
```

```
    cout << "\n\n\tEl nombre esta: " << esVacia << endl;
```

```
    return 0;
```

```
}
```

```
El nombre esta: 0

El nombre esta: 1

Press <RETURN> to close this window...
```

// false (0)

// true (1)

Tipo de datos string (std::string) - Funciones

at Permite acceder al carácter en la posición *n-ésima* del *string*. La primera posición es *0* y la última es *tamaño – 1*.

```
#include <iostream>
```

```
#include <string>
```

```
using namespace std;
```

```
int main(){
```

```
    string ciudad{"Santiago de Compostela"};
```

```
    char letra = ciudad.at(7);
```

```
    cout << "\n\n\tLa letra es: " << letra;
```

```
    letra = ciudad.at(0);
```

```
    cout << "\n\n\tLa letra es: " << letra;
```

```
    letra = ciudad.at(ciudad.size() - 1);
```

```
    cout << "\n\n\tLa letra es: " << letra << endl;
```

```
    return 0;
```

```
}
```

```
La letra es: o
```

```
La letra es: S
```

```
La letra es: a
```

```
Press <RETURN> to close this window...
```

```
// 'g'
```

```
// 'S'
```

```
// 'a'
```

Tipo de datos string (std::string) - Funciones



front Devuelve el primer carácter del *string*.

back Devuelve el último carácter del *string*.

```
#include <iostream>
```

```
#include <string>
```

```
using namespace std;
```

```
int main(){
```

```
    string ciudad{"Santiago de Compostela"};
```

```
    char letra = ciudad.front();
```

```
    cout << "\n\n\tLa letra es: " << letra;
```

```
    letra = ciudad.back();
```

```
    cout << "\n\n\tLa letra es: " << letra << endl;
```

```
    return 0;
```

```
}
```

```
La letra es: S
La letra es: a
Press <RETURN> to close this window...
```

// 'S'

// 'a'

Tipo de datos string (std::string) - Funciones

append Añade una cadena de texto al **final** de un *string* determinado.

```
#include <iostream>
#include <string>
```

```
using namespace std;
```

```
int main(){
    string refran{"No por mucho madrugar"};
    refran.append(" amanece mas temprano");
    cout << "\n\n\tEl refran es:"\n\n\t\t" << refran << endl;

    return 0;
}
```

```
El refran es:
        No por mucho madrugar amanece mas temprano
Press <RETURN> to close this window...
```

Tipo de datos string (std::string) - Funciones

operador + Concatena dos cadenas de texto en un *string*.

```
#include <iostream>
#include <string>
```

```
using namespace std;
```

```
int main(){
```

```
    string refran, parte1 {"No por mucho madrugar"},
           parte2 {" amanecer mas temprano."};
```

```
    refran = parte1 + parte2;
```

```
    cout << "\n\n\tEl refran es:\n\n\t\t" << refran << endl;
```

```
    return 0;
```

```
}
```

```
El refran es:
           No por mucho madrugar amanecer mas temprano.

Press <RETURN> to close this window...
```

Tipo de datos string (std::string) - Funciones

push_back Añade un carácter al **final** del *string*.

```
#include <iostream>
```

```
#include <string>
```

```
using namespace std;
```

```
int main(){
```

```
    string refran{"El que tiene boca se equivoca"};
```

```
    refran.push_back('.');
```

```
    cout << "\n\n\tEl refran es:\n\n\t\t" << refran << endl;
```

```
    return 0;
```

```
}
```

```
El refran es:
                El que tiene boca se equivoca.
Press <RETURN> to close this window...
```

Tipo de datos string (std::string) - Funciones

pop_back Elimina el último carácter del *string*.

```
#include <iostream>
#include <string>
```

```
using namespace std;
```

```
int main(){
```

```
    string refran{"El que tiene boca se equivoca.*"};
```

```
    refran.pop_back();
```

```
    cout << "\n\n\tEl refran es :\n\n\t\t" << refran << endl;
```

```
    return 0;
```

```
}
```

```
El refran es :
                El que tiene boca se equivoca.
Press <RETURN> to close this window...
```

Tipo de datos string (std::string) - Funciones

insert Introduce una cadena de texto antes de la posición indicada por el índice.

```
#include <iostream>
```

```
#include <string>
```

```
using namespace std;
```

```
int main(){
```

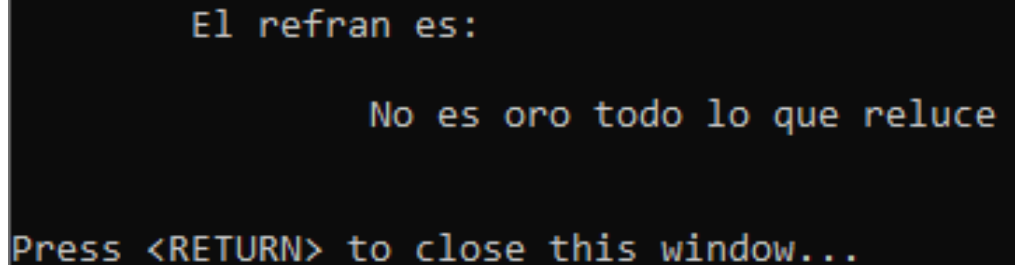
```
    string refran{"es oro todo lo que reluce"};
```

```
    refran.insert (0, "No ");
```

```
    cout << "\n\n\tEl refran es:\n\n\t\t" << refran << endl;
```

```
    return 0;
```

```
}
```



```
El refran es:  
    No es oro todo lo que reluce  
  
Press <RETURN> to close this window...  
-
```


Tipo de datos string (std::string) - Funciones

erase Elimina una parte de una cadena de texto, para ello hay que indicar la **posición inicial** y el **número de caracteres a eliminar**.

```
#include <iostream>
```

```
#include <string>
```

```
using namespace std;
```

```
int main(){
```

```
    string refran{"El saber no ocupa ningun lugar."};
```

```
    refran.erase(18, 7);
```

```
    cout << "\n\n\tEl refran es:\n\n\t\t" << refran << endl;
```

```
    return 0;
```

```
}
```

```
El refran es:
           El saber no ocupa lugar.
Press <RETURN> to close this window...
```

Tipo de datos `string` (`std::string`) - Funciones

find Busca una cadena de texto dentro de otra, desde el inicio o a partir de la posición que se indique.

La función *find* devuelve la posición donde empieza la cadena buscada. Si no se encuentra dicha cadena dentro del *string*, la función devuelve **-1**.

Tipo de datos string (std::string) - Funciones

find Buscar desde el principio del *string*.

```
#include <iostream>
#include <string>

using namespace std;
```

```
int main(){
    string refran{"A grandes males, grandes remedios."};
    int posicion = refran.find("grandes");
    cout << "\n\n\tEl refran es: " << refran;
    cout << "\n\n\tLa primera posicion de la palabra dentro del refran es: "
    << posicion + 1 << endl;

    return 0;
}
```

```
El refran es: A grandes males, grandes remedios.
```

```
La primera posicion de la palabra dentro del refran es: 3
```

```
Press <RETURN> to close this window...
```

Tipo de datos string (std::string) - Funciones

find Buscar desde una **determinada posición** del *string*.

```
#include <iostream>
#include <string>
```

```
using namespace std;
```

```
int main(){
```

```
    string refran{"A grandes males, grandes remedios."};
```

```
    int posicion = refran.find("grandes", 9);
```

```
    cout << "\n\n\tEl refran es: " << refran;
```

```
    cout << "\n\n\tLa segunda posicion de la palabra dentro del refran es: "
    << posicion + 1 << endl;
```

```
    return 0;
```

```
}
```

```
El refran es: A grandes males, grandes remedios.
```

```
La segunda posicion de la palabra dentro del refran es: 18
```

```
Press <RETURN> to close this window...
```

Tipo de datos string (std::string) - Funciones

find si la cadena **no se encuentra**, la función **devuelve -1**.

```
#include <iostream>
#include <string>

using namespace std;
```

```
La posicion de la palabra dentro del refran es: -1

Press <RETURN> to close this window...
```

```
int main(){
    string refran{"A grandes males, grandes remedios."};
    int posicion = refran.find("sol");
    cout << "\n\n\tLa posicion de la palabra dentro del refran es: " <<
    posicion << endl;

    return 0;
}
```

Tipo de datos string (std::string) - Funciones



substr Extrae una subcadena de otra cadena. Hay que indicar la **posición inicial** y el **número de caracteres** a extraer.

```
#include <iostream>
#include <string>
```

```
using namespace std;
```

```
int main(){
```

```
    string refran{"A grandes males, grandes remedios."};
```

```
    string parte_refran = refran.substr(2, 13);
```

```
    cout << "\n\n\tLa subcadena del refran es: " << parte_refran << endl;
```

```
    return 0;
```

```
}
```

```
La subcadena del refran es: grandes males
Press <RETURN> to close this window...
```

Tipo de datos string (std::string) – E/S

Si el *string* solo contiene una palabra, sin espacios en blanco, la lectura por teclado es igual que otra variable simple.

```
#include <iostream>
#include <string>

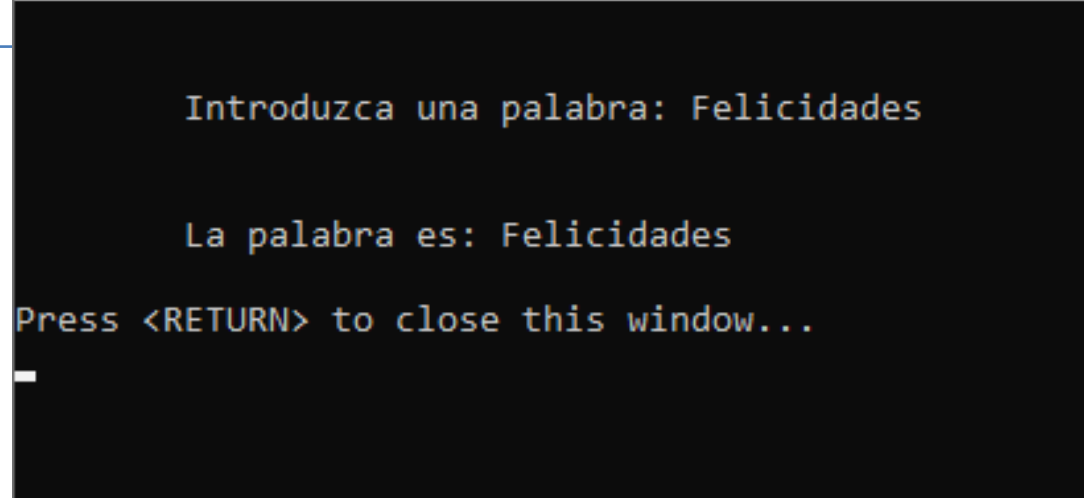
using namespace std;

int main(){
    string palabra;

    cout << "\n\n\tIntroduzca una palabra: ";
    cin >> palabra;

    cout << "\n\n\tLa palabra es: " << palabra;

    return 0;
}
```



```
Introduzca una palabra: Felicidades

La palabra es: Felicidades

Press <RETURN> to close this window...
_
```

Tipo de datos string (std::string) – E/S

Si el *string* contiene una **frase**, una línea de texto, la lectura por teclado se debe hacer a través de la función *getline*.

```
#include <iostream>
#include <string>

using namespace std;

int main(){
    string frase;

    cout << "\n\n\tIntroduzca una frase: ";
    getline(cin, frase);

    cout << "\n\n\tLa frase es: " << frase;

    return 0;
}
```

```
Introduzca una frase: El tiempo es oro

La frase es: El tiempo es oro
Press <RETURN> to close this window...
```


Control de errores - (std::cin.fail())

La función *fail* permite detectar si el valor tecleado es adecuado para asociarlo a la variable indicada en la lectura.

```
#include <iostream>
#include <string>

using namespace std;
```

```
int main(){
    int numero;
    cout << "\n\n\tIntroduzca numero: ";
    cin >> numero;
    if(cin.fail())
        cout << "\n\n\tERROR, no se ha tecleado un numero\n";
    else cout << "\n\n\tEl numero tecleado es " << numero << "\n";
    return 0;
}
```

```
Introduzca numero: Hace sol

ERROR, no se ha tecleado un numero

Press <RETURN> to close this window...
```

Control de errores - (std::cin.fail())

```
#include <iostream>
#include <string>

using namespace std;
```

```
int main(){
    double numero;
    cout << "\n\n\tIntroduzca numero: ";
    cin >> numero;
```

```
    if(cin.fail())
        cout << "\n\n\tERROR, no se ha tecleado un numero\n";
    else cout << "\n\n\tEl numero tecleado es " << numero << "\n";
    return 0;
```

```
}
```

```
Introduzca un numero: 3.1416

El numero tecleado es 3.1416

Press <RETURN> to close this window...
```

```
Introduzca numero: Hace sol

ERROR, no se ha tecleado un numero

Press <RETURN> to close this window...
```

Control de errores - (std::cin.fail())

```
Introduzca una frase: 234
```

```
La frase tecleada es 234
```

```
Press <RETURN> to close t
```

```
Introduzca una frase: Hoy es domingo
```

```
La frase tecleada es Hoy es domingo
```

```
Press <RETURN> to close this window...
```

```
#include <iostream>
#include <string>
```

```
using namespace std;
```

```
int main(){
    string frase;
    cout << "\n\n\tIntroduzca una frase: ";
    getline(cin, frase);
    if (cin.fail())
        cout << "\n\n\tERROR, no se ha tecleado una frase\n";
    else cout << "\n\n\tLa frase tecleada es " << frase << "\n";
    return 0;
```

```
}
```