Grado en Ingeniería Información

PROGRAMACIÓN I

Sesión 10

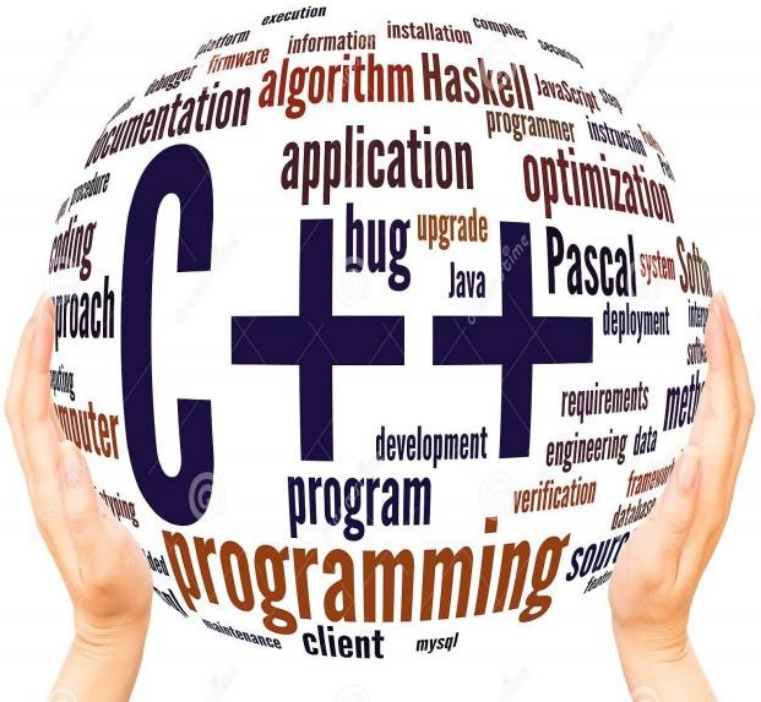
Curso 2022-2023

Marta N. Gómez (mgomezper@nebrija.es)

UNIVERSIDAD NEBRIJA

Tipos de datos estructurados.

- std::string
- std::array
- std::vector
- **std::set**



ÍNDICE

Tipo de datos set (std::set)

UNIVERSIDAD NEBRIJA

- El tipo ***set*** o **conjunto** es una secuencia **ordenada** (**de menor a mayor**) de datos **homogéneos** (mismo tipo) que **no se repiten**.
- Su **declaración** sólo necesita que se indique el **tipo de dato** que se guarda en el **conjunto** (***set***).

std::set<tipo> var_set;

- Su tamaño inicial será 0 (al declararlo) y modificará su tamaño según se vayan añadiendo o eliminando elementos en él.
- Necesita incluir la biblioteca de C++ ***set:***

#include <set>

Tipo de datos set (std::set)

UNIVERSIDAD NEBRIJA

- Declaración del tipo ***set***:

  Se crea un conjunto de números decimales **vacío**.

  ```
  std::set<float> set1;
  ```

- Inicialización del tipo ***set***:
  - Indicando sus valores entre **llaves** y separados por **comas**.

    ```
    std::set<int> un_set{1, 4, 3, 2, 5, 9, 6, 7, 8};
    ```

  - Utilizando el operador **=**

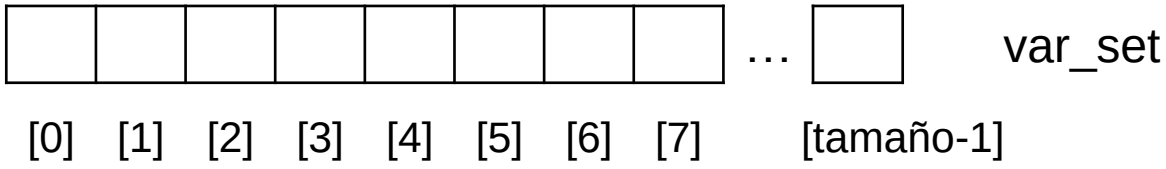
    ```
    std::set<char> vocales ={'o', 'i', 'e', 'a', 'u'};
    ```

El orden de los elementos en un ***set*** siempre en de **menor a mayor**, con independencia de su inicialización.
Por tanto, internamente los conjuntos anteriores serán:

```
{1, 2, 3, 4, 5, 6, 7, 8, 9};        {'a', 'e', 'i', 'o', 'u'};
```

Tipo de datos set (std::set)

- El tipo ***set*** es un tipo de datos **contenedor**.
- Los elementos de un ***set*** o conjunto **no** son accesibles a través de su ***índice*** (posición).
- El acceso se hace a través del operador ***punto*** **(.)**



Tipo de datos set (std::set)

- El tipo ***set*** 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 set (std::set) - Funciones

UNIVERSIDAD NEBRIJA

insert Permite añadir elementos al ***set***. Si el elemento está repetido, se ignora

Ejemplo1 Set

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

using namespace std;

int main()
{
    set<string> setNombres;

    setNombres.insert("Sancho");
    setNombres.insert("Quijote");
    setNombres.insert("Marcelino");

    for (string elem:setNombres)
        cout << "\n\t" << elem;

    cout << "\n\n\t";
    return 0;
}
```

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

UNIVERSIDAD NEBRIJA

insert Permite añadir elementos al ***set***. Si el elemento está repetido, se ignora

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

using namespace std;

int main()
{
    set<string> setNombres;

    setNombres.insert("Sancho");
    setNombres.insert("Quijote");
    setNombres.insert("Marcelino");

    for (string elem:setNombres)
        cout << "\n\t" << elem;

    cout << "\n\n\t";
    return 0;
}
```

```
    Marcelino
    Quijote
    Sancho

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

Ejemplo2 Set

```cpp
#include <iostream>
#include <set>
#include <string>

using namespace std;

int main()
{
    set<string> setNombres;

    setNombres.insert("Sancho");
    setNombres.insert("Quijote");
    setNombres.insert("Marcelino");

    cout << "\n\tPRIMERO:";
    for (string elem:setNombres)
        cout << "\n\t" << elem;

    setNombres.insert("Quijote");

    cout << "\n\n\tDESPUES:";
    for (string elem:setNombres)
        cout << "\n\t" << elem;

    cout << "\n\n\t";
    return 0;
}
```

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

```cpp
#include <iostream>
#include <set>
#include <string>

using namespace std;

int main()
{
    set<string> setNombres;

    setNombres.insert("Sancho");
    setNombres.insert("Quijote");
    setNombres.insert("Marcelino");

    cout << "\n\tPRIMERO:";
    for (string elem:setNombres)
        cout << "\n\t" << elem;

    setNombres.insert("Quijote");

    cout << "\n\n\tDESPUES:";
    for (string elem:setNombres)
        cout << "\n\t" << elem;

    cout << "\n\n\t";
    return 0;
}
```

```
PRIMERO:
Marcelino
Quijote
Sancho

DESPUES:
Marcelino
Quijote
Sancho

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

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

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

clear Elimina todos los elementos del ***set***, dejando su tamaño a 0.

Ejemplo3 Set

```
#include <iostream>
#include <set>

using namespace std;

int main()
{
    set<int> setNum;

    setNum.insert(20);
    setNum.insert(10);
    setNum.insert(35);
    setNum.insert(15);
    setNum.insert(35);

    cout << "\n\tPRIMERO:";
    for (int elem:setNum)
        cout << "\n\t" << elem;

    setNum.clear();
    cout << "\n\n\tDESPUES:";
    for (int elem:setNum)
        cout << "\n\t" << elem;

    cout << "\n\n\t";
    return 0;
}
```

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

clear Elimina todos los elementos del ***set***, dejando su tamaño a 0.

```
#include <iostream>
#include <set>

using namespace std;

int main()
{
    set<int> setNum;

    setNum.insert(20);
    setNum.insert(10);
    setNum.insert(35);
    setNum.insert(15);
    setNum.insert(35);

    cout << "\n\tPRIMERO:";
    for (int elem:setNum)
        cout << "\n\t" << elem;

    setNum.clear();
    cout << "\n\n\tDESPUES:";
    for (int elem:setNum)
        cout << "\n\t" << elem;

    cout << "\n\n\t";
    return 0;
}
```

```
PRIMERO:
10
15
20
35

DESPUES:

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

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

empty Indica si el ***set*** contiene datos (***true***) o no (***false***).

Ejemplo4 Set

```
#include <iostream>
#include <set>

using namespace std;

int main()
{
    set<int> setNum;

    setNum.insert(20);
    setNum.insert(10);
    setNum.insert(35);
    setNum.insert(15);
    setNum.insert(35);

    cout << "\n\tANTES:";
    if(setNum.empty())  cout << "\n\n\tSET Vacio\n";
    else  cout << "\n\n\tSET Lleno\n";

    setNum.clear();
    cout << "\n\n\tDESPUES:";
      if(setNum.empty())  cout << "\n\n\tSET Vacio\n";
      else  cout << "\n\n\tSET Lleno\n";

    cout << "\n\n\t";
    return 0;
}
```

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

empty Indica si el *set* contiene datos (*true*) o no (*false*).

```
#include <iostream>
#include <set>

using namespace std;

int main()
{
    set<int> setNum;

    setNum.insert(20);
    setNum.insert(10);
    setNum.insert(35);
    setNum.insert(15);
    setNum.insert(35);

    cout << "\n\tANTES:";
    if(setNum.empty())  cout << "\n\n\tSET Vacio\n";
    else  cout << "\n\n\tSET Lleno\n";

    setNum.clear();
    cout << "\n\n\tDESPUES:";
      if(setNum.empty())  cout << "\n\n\tSET Vacio\n";
      else  cout << "\n\n\tSET Lleno\n";

    cout << "\n\n\t";
    return 0;
}
```

```
ANTES:

SET Lleno


DESPUES:

SET Vacio


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

UNIVERSIDAD NEBRIJA

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

size Devuelve el **número de elementos** que tiene el ***set***.

Ejemplo5 Set

```
#include <iostream>
#include <set>

using namespace std;

int main()
{
    set<int> setNum1, setNum2;
    const int total{5};

    for (int i{0}; i < total; i++)
        setNum1.insert(total-i);

    cout << "\n\n\tDatos de setNum1 ";
    for (int elem:setNum1)
        cout << "  " << elem;

    cout << "\n\n\tsetNum1 tiene: "<< setNum1.size() << " elementos.";
    cout << "\n\n\tsetNum2 tiene: "<< setNum2.size() << " elementos.";

    cout << "\n\n\t";
    return 0;
}
```

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

size Devuelve el **número de elementos** que tiene el ***set***.

```cpp
#include <iostream>
#include <set>

using namespace std;

int main()
{
    set<int> setNum1, setNum2;
    const int total{5};

    for (int i{0}; i < total; i++)
        setNum1.insert(total-i);

    cout << "\n\n\tDatos de setNum1 ";
    for (int elem:setNum1)
        cout << "  " << elem;

    cout << "\n\n\tsetNum1 tiene: "<< setNum1.size() << " elementos.";
    cout << "\n\n\tsetNum2 tiene: "<< setNum2.size() << " elementos.";

    cout << "\n\n\t";
    return 0;
}
```

```
Datos de setNum1   1  2  3  4  5

setNum1 tiene: 5 elementos.

setNum2 tiene: 0 elementos.

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

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

find Entre otras cosas, permite saber si en un conjunto (***set***) existe determinado elemento o no.

La función ***find*** devuelve lo que se llama un ***iterador*** a la posición donde se ha encontrado el elemento buscado. Si dicho ***iterador*** apunta al final del conjunto ***set.end*** quiere decir que el elemento no ha sido encontrado.

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

find Permite saber si un determinado elemento está en el ***set***.

Ejemplo6 Set

```
#include <iostream>
#include <set>

using namespace std;

int main()
{
    set<int> setNum;
    setNum.insert(20);
    setNum.insert(10);
    setNum.insert(35);
    setNum.insert(15);
    setNum.insert(35);

    if(setNum.find(15) == setNum.end())
        cout << "\n\tEl " << 15 << " NO existe en setNum";
    else   cout << "\n\tEl " << 15 << " SI existe en setNum";

    if(setNum.find(25) == setNum.end())
        cout << "\n\tEl " << 25 << " NO existe en setNum";
    else   cout << "\n\tEl " << 25 << " SI existe en setNum";

    cout << "\n\n\t";
    return 0;
}
```

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

find Permite saber si un determinado elemento está en el ***set***.

```
#include <iostream>
#include <set>

using namespace std;

int main()
{
    set<int> setNum;
    setNum.insert(20);
    setNum.insert(10);
    setNum.insert(35);
    setNum.insert(15);
    setNum.insert(35);

    if(setNum.find(15) == setNum.end())
        cout << "\n\tEl " << 15 << " NO existe en setNum";
    else   cout << "\n\tEl " << 15 << " SI existe en setNum";

    if(setNum.find(25) == setNum.end())
        cout << "\n\tEl " << 25 << " NO existe en setNum";
    else   cout << "\n\tEl " << 25 << " SI existe en setNum";

    cout << "\n\n\t";
    return 0;
}
```

```
El 15 SI existe en setNum
El 25 NO existe en setNum

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

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

erase Elimina un determinado elemento del ***set***, si existe. En otro caso, no hace nada.

Ejemplo7 Set

```
#include <iostream>
#include <set>

using namespace std;

int main()
{
    set<int> setNum1;
    const int total{5};

    for (int i{0}; i < total; i++)
        setNum1.insert(total-i);
    cout << "\n\n\tDatos de setNum1 ANTES:";
    for (int elem:setNum1)
        cout << "  " << elem;
    setNum1.erase(5);
    setNum1.erase(100);
    cout << "\n\n\tDatos de setNum1 DESPUES:";
    for (int elem:setNum1)
        cout << "  " << elem;

    cout << "\n\n\t";
    return 0;
}
```

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

erase Elimina un determinado elemento del ***set***, si existe. En otro caso, no hace nada.

```
Datos de setNum1 ANTES:  1  2  3  4  5

Datos de setNum1 DESPUES:  1  2  3  4

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

```cpp
#include <iostream>
#include <set>

using namespace std;

int main()
{
    set<int> setNum1;
    const int total{5};

    for (int i{0}; i < total; i++)
        setNum1.insert(total-i);
    cout << "\n\n\tDatos de setNum1 ANTES:";
    for (int elem:setNum1)
        cout << "  " << elem;
    setNum1.erase(5);
    setNum1.erase(100);
    cout << "\n\n\tDatos de setNum1 DESPUES:";
    for (int elem:setNum1)
        cout << "  " << elem;

    cout << "\n\n\t";
    return 0;
}
```

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

erase Permite saber si un elemento ha sido borrado del ***set***

Ejemplo8 Set

```
#include <iostream>
#include <set>

using namespace std;

int main()
{    set<int> setNum;
     setNum.insert(20);
     setNum.insert(10);
     setNum.insert(35);
     setNum.insert(15);

     cout << "\n\n\tEl elemento " << 35;
     if(setNum.erase(35)) cout << " ha sido Borrado.";
     else  cout << " NO se ha encontrado.";

     cout << "\n\n\tEl elemento " << 100;
     if(setNum.erase(100)) cout << " ha sido Borrado.";
     else  cout << " NO se ha encontrado.";

    cout << "\n\n\t";
    return 0;
}
```

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

erase Permite saber si un elemento ha sido borrado del *set*

```
#include <iostream>
#include <set>

using namespace std;

int main()
{    set<int> setNum;
     setNum.insert(20);
     setNum.insert(10);
     setNum.insert(35);
     setNum.insert(15);

     cout << "\n\n\tEl elemento " << 35;
     if(setNum.erase(35)) cout << " ha sido Borrado.";
     else  cout << " NO se ha encontrado.";

     cout << "\n\n\tEl elemento " << 100;
     if(setNum.erase(100)) cout << " ha sido Borrado.";
     else  cout << " NO se ha encontrado.";

    cout << "\n\n\t";
    return 0;
}
```

```
El elemento 35 ha sido Borrado.

El elemento 100 NO se ha encontrado.

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

Tipo de datos set (std::set)

- El tipo de elementos de un ***set*** pueden ser cualquiera: ***int, char, string, bool, struct, array, vector.***

```
set <set<tipo>> var_set;
```

Tipo de datos

- Ejmplo

```
set<set<int>> set1;
```

```
set<set<int>> set1 {{0 1}, {2}, {3, 4, 5}};
```

Tipo de datos set (std::set)

UNIVERSIDAD NEBRIJA

Ejemplo9 Set

```
#include <iostream>
#include <set>

using namespace std;

int main()
{
    set<set<int>> setNum;

    setNum.insert({7,4});
    setNum.insert({4});
    setNum.insert({8,3,6});
    for (set<int> conjunto:setNum)
    {   cout << "\n\n\t{";
        for (int elem:conjunto)
            cout << " " << elem;
        cout << " }";
    }

    cout << "\n\n\t";
    return 0;
}
```

Tipo de datos set (std::set)

UNIVERSIDAD NEBRIJA

```cpp
#include <iostream>
#include <set>

using namespace std;

int main()
{
    set<set<int>> setNum;

    setNum.insert({7,4});
    setNum.insert({4});
    setNum.insert({8,3,6});
    for (set<int> conjunto:setNum)
    {   cout << "\n\n\t{";
        for (int elem:conjunto)
            cout << " " << elem;
        cout << " }";
    }

    cout << "\n\n\t";
    return 0;
}
```

```
{ 3 6 8 }

{ 4 }

{ 4 7 }

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