

Investigación de palabra clave **Static**

Definición

En C++, 'static' es una palabra clave que se utiliza para indicar que una variable o una función tiene alcance local y duración de almacenamiento estática. La palabra clave 'static' en C++ se utiliza para controlar el alcance y la duración de almacenamiento de las variables y funciones.

Ventajas y desventajas

Ventajas de usar variables estáticas:

- Almacenamiento de datos duradero: las variables estáticas se inicializan solo una vez durante la ejecución del programa y su valor se mantiene incluso después de que la función en la que se declara haya terminado.
- Control del ámbito: las variables estáticas sólo son visibles dentro del ámbito en el que se declaran.
- Variables compartidas: las variables estáticas en una clase son compartidas por todas las instancias de la clase.

Desventajas de usar variables estáticas:

- Uso de memoria: las variables estáticas pueden ocupar memoria adicional en tiempo de ejecución, especialmente si se utilizan en grandes cantidades.
- Pérdida de modularidad: las variables estáticas pueden dificultar la modularidad del código y hacer que sea más difícil de entender y mantener.
- Dificultad para realizar pruebas unitarias: las variables estáticas pueden dificultar la realización de pruebas unitarias debido a su duración de almacenamiento y su alcance.

Ventajas de usar funciones estáticas:

- Uso sin instancia: las funciones estáticas se pueden llamar sin tener una instancia de la clase, lo que puede ser útil en algunas situaciones.
- Control del ámbito: las funciones estáticas solo son visibles dentro del archivo de origen en el que se declaran.
- Compartir código: las funciones estáticas se pueden utilizar para compartir código común entre varias funciones en el mismo archivo de origen.

Desventajas de usar funciones estáticas:

- Limitación de acceso a datos de instancia: las funciones estáticas no tienen acceso a los datos de instancia de una clase y solo pueden utilizar datos estáticos.
- Dificultad para realizar pruebas unitarias: las funciones estáticas pueden dificultar la realización de pruebas unitarias debido a su duración de almacenamiento y su alcance.

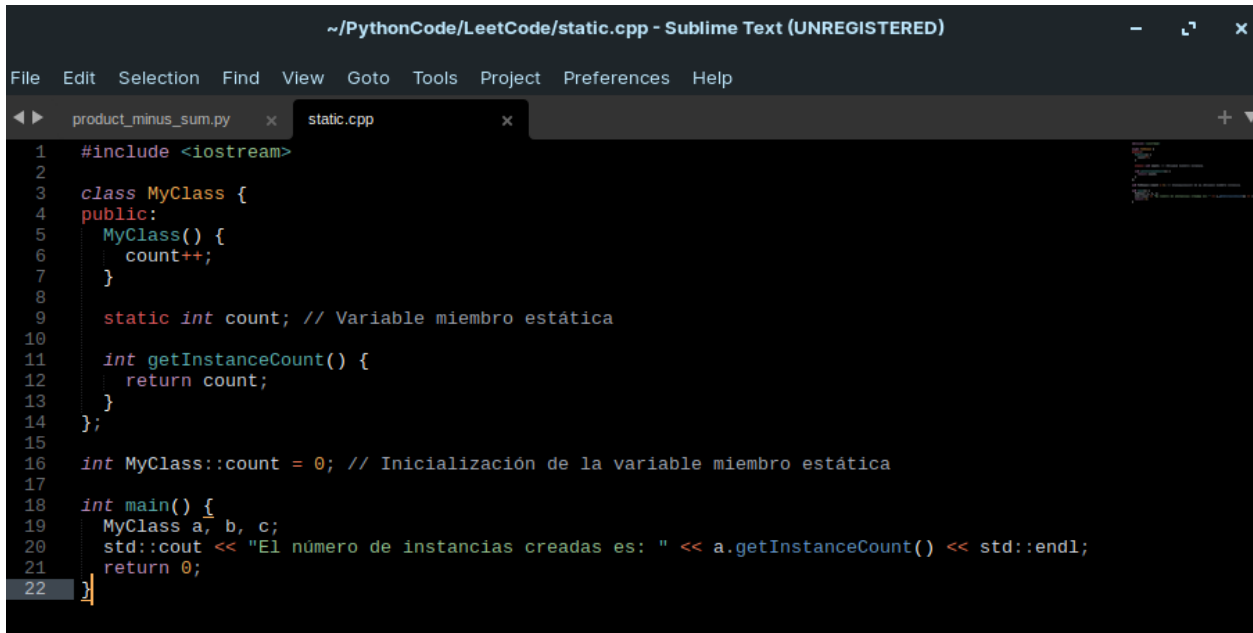
- Limitaciones de flexibilidad: las funciones estáticas no se pueden sobrescribir en las subclases.

En variables miembro

Cuando se utiliza con una variable, 'static' se utiliza para indicar que la variable se inicializa solo una vez durante la ejecución del programa y que su valor se mantiene incluso después de que la función en la que se declara ha terminado. Además, la variable solo es visible dentro del ámbito en el que se declara.

En C++, cuando se utiliza la palabra clave 'static' en una variable miembro de una clase, se crea una variable compartida por todas las instancias de la clase, en lugar de crear una variable separada para cada instancia.

Un ejemplo de uso de 'static' en una variable miembro de una clase sería el siguiente:



```
~/PythonCode/LeetCode/static.cpp - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
product_minus_sum.py x static.cpp x
1 #include <iostream>
2
3 class MyClass {
4 public:
5     MyClass() {
6         count++;
7     }
8
9     static int count; // Variable miembro estática
10
11     int getInstanceCount() {
12         return count;
13     }
14 };
15
16 int MyClass::count = 0; // Inicialización de la variable miembro estática
17
18 int main() {
19     MyClass a, b, c;
20     std::cout << "El número de instancias creadas es: " << a.getInstanceCount() << std::endl;
21     return 0;
22 }
```

En este ejemplo, la clase MyClass tiene una variable miembro estática llamada count, que se utiliza para contar el número de instancias creadas de la clase. La variable se inicializa en 0 y se incrementa en 1 cada vez que se crea una instancia de la clase.

La función miembro getInstanceCount() devuelve el valor de count, lo que nos permite saber cuántas instancias de la clase se han creado hasta el momento.

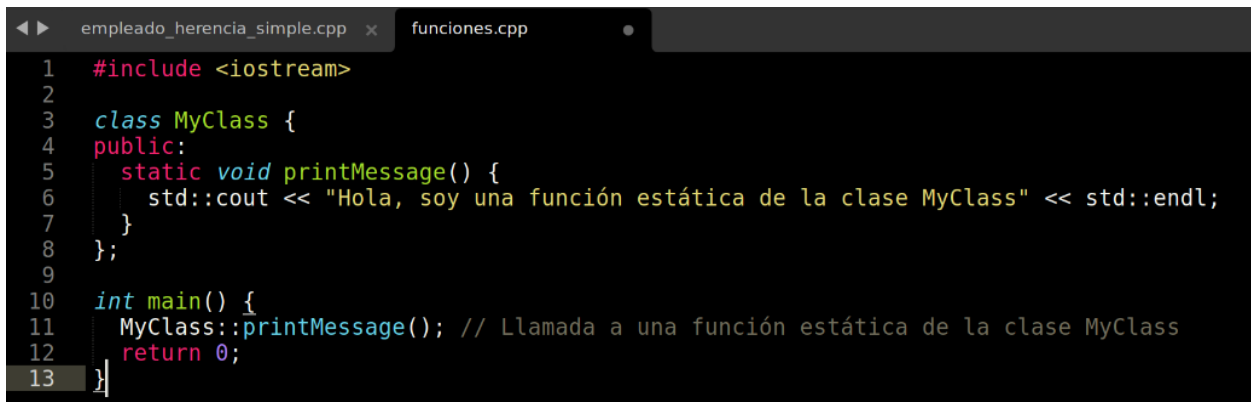
Como la variable miembro count es estática, todas las instancias de la clase comparten la misma variable, y el valor de count es el mismo para todas las instancias.

En funciones miembro

Cuando se utiliza con una función, 'static' se utiliza para indicar que la función tiene alcance local y que solo se puede llamar desde dentro del archivo de origen en el que se declara.

En C++, cuando se utiliza la palabra clave 'static' en una función miembro de una clase, la función se convierte en una función estática de la clase, lo que significa que se puede llamar sin tener una instancia de la clase.

Un ejemplo de uso de 'static' en una función miembro de una clase sería el siguiente:

A screenshot of a code editor with two tabs: 'empleado_herencia_simple.cpp' and 'funciones.cpp'. The 'funciones.cpp' tab is active, showing C++ code. The code defines a class 'MyClass' with a public static member function 'printMessage()' that prints a message to the console. The 'main()' function calls 'MyClass::printMessage()' to demonstrate the static function.

```
1  #include <iostream>
2
3  class MyClass {
4  public:
5      static void printMessage() {
6          std::cout << "Hola, soy una función estática de la clase MyClass" << std::endl;
7      }
8  };
9
10 int main() {
11     MyClass::printMessage(); // Llamada a una función estática de la clase MyClass
12     return 0;
13 }
```

En este ejemplo, la clase 'MyClass' tiene una función miembro estática llamada 'printMessage()'. La función se define dentro de la clase utilizando la palabra clave 'static'.

En la función 'main()', la función estática se llama utilizando el nombre de la clase y el operador de resolución de ámbito '::'. Como la función es estática, no es necesario tener una instancia de la clase para llamarla.

Referencias

cplusplus.com. (2022). Static keyword. <http://www.cplusplus.com/doc/oldtutorial/static/>

Microsoft. (2022). Static keyword (C++).

<https://docs.microsoft.com/en-us/cpp/cpp/static-cpp?view=msvc-170>

GeeksforGeeks. (2022). Static keyword in C++ | Set 1 (Data Member).

<https://www.geeksforgeeks.org/static-keyword-cpp/>