Differences Between Normal and Static Functions in C++

S.A. Abdulla

21/01/2024

Differences Between Normal and Static Functions in C++

Access to Members

Normal Function: Can access both static and non-static members directly. Requires an instance of the class to access non-static members.
Static Function: Can only access static members directly. Cannot access non-static members directly.

Usage Without Instance

Normal Function: Requires an instance of the class to be called.
Static Function: Can be called using the class name, without creating an instance.

`this` Pointer

Normal Function: Has access to the `this` pointer, which points to the instance of the class it is called on.
Static Function: Does not have access to the `this` pointer, as it is not associated with any particular instance.

Memory Allocation

Normal Function: Each instance of the class has its own set of non-static members, and these members are allocated memory for each instance.
Static Function: Shares the same set of static members among all instances of the class. Memory is allocated once for static members.

Visibility in Derived Classes

Normal Function: Can be overridden in derived classes.
Static Function: Cannot be overridden in derived classes. The function associated with the base class will be called even if it's called on a derived class object.

C++ Code Example

```
#include <iostream>

class Example {
public:
    int nonStaticVar = 42;

    void normalFunction() {
        std::cout << "Normal␣Function" << std::endl;
        std::cout << "Accessing␣non-staticVar:␣" << nonStaticVar << std::endl;
    }

    static void staticFunction() {
        std::cout << "Static␣Function" << std::endl;
        // Uncommenting the line below would result in a compilation error.
        // std::cout << "Accessing non-staticVar: " << nonStaticVar << std::endl;
        // Static functions do not have access to 'this'.
        // Uncommenting the line below would result in a compilation error.
        // std::cout << "this pointer value: " << this << std::endl;
    }
};

int main() {
    Example obj;

    obj.normalFunction();  // Accessing normal function
    Example::staticFunction();  // Accessing static function

    return 0;
}
```