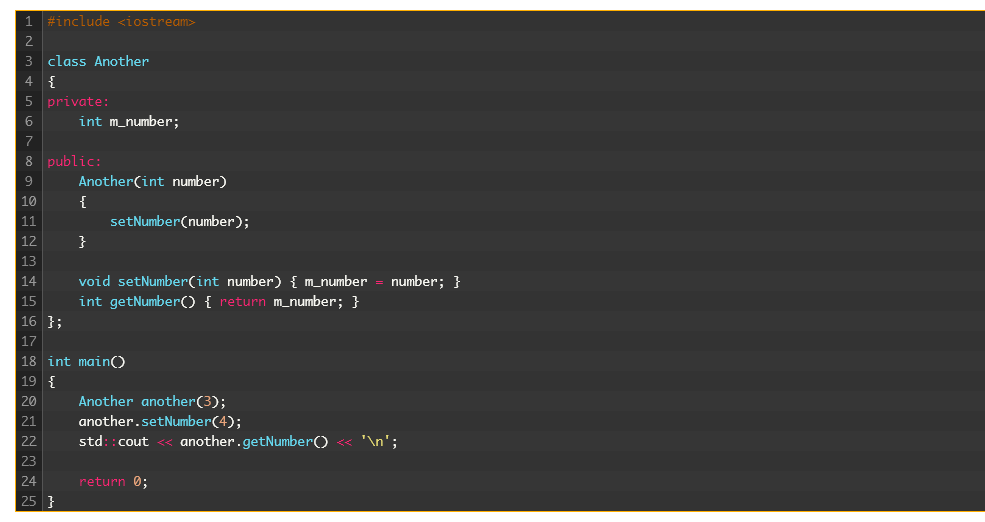
## Скрытый указатель \*this



При компиляции обычного метода, компилятор неявно добавляет к нему параметр \*this. Указатель \*this — это скрытый [константный указатель](https://ravesli.com/urok-87-ukazateli-i-const/), содержащий адрес объекта, который вызывает метод класса.

 При вызове another.setNumber(4) компилятор фактически вызывает setNumber(&another, 4).

   Внутри setNumber() указатель \*this содержит адрес объекта another.

   К любым переменным-членам внутри setNumber() добавляется префикс this->. Поэтому, когда мы говорим m\_number = number, компилятор фактически выполняет this->m\_number = number, который, в этом случае, обновляет another.m\_number на number.

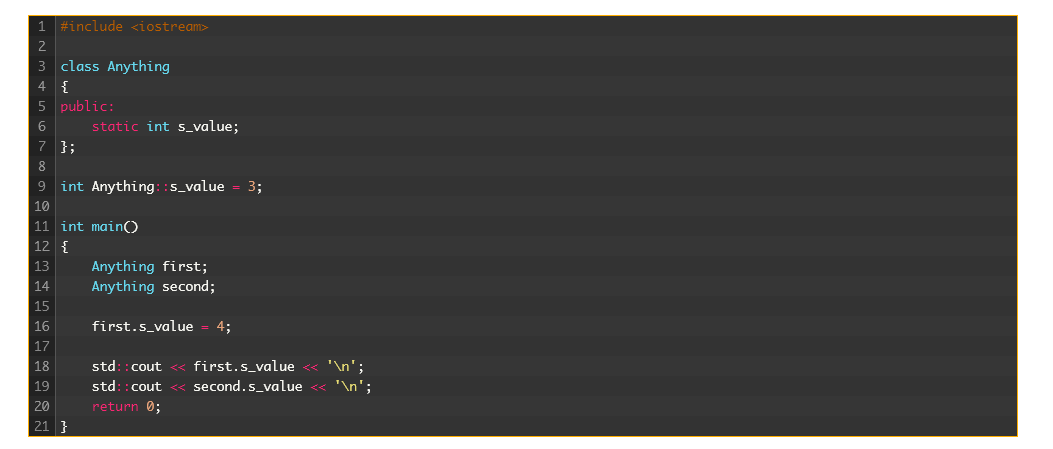
Иногда бывает полезно, чтобы метод класса возвращал объект, с которым работает, в виде возвращаемого значения. Основной смысл здесь — это позволить нескольким методам объединиться в «цепочку», работая при этом с одним объектом!

Если каждая функция будет возвращать указатель \*this, то мы сможем связать эти вызовы методов в одну цепочку :



## Статические переменные-члены класса

Переменные-члены класса можно сделать статическими, используя ключевое слово static. В отличие от обычных переменных-членов, статические переменные-члены являются общими для всех объектов класса. Рассмотрим следующую программу:

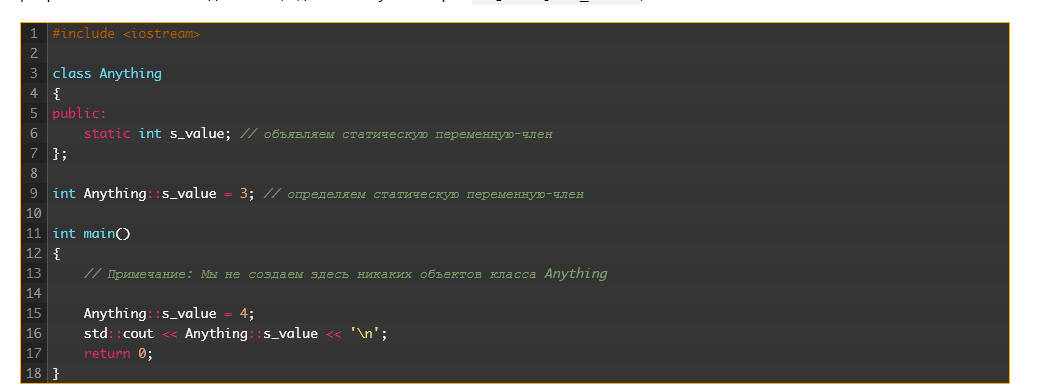


Поскольку s\_value является статической переменной-членом, то она является общей для всех объектов класса Anything. Следовательно, first.s\_value — это та же переменная, что и second.s\_value.

## Статические члены не связаны с объектами класса

Хотя вы можете получить доступ к статическим членам через разные объекты класса (как в примере, приведенном выше), но, оказывается, статические члены существуют, даже если объекты класса не созданы! Подобно глобальным переменным, они создаются при запуске программы и уничтожаются, когда программа завершает свое выполнение.

Следовательно, статические члены принадлежат классу, а не объектам этого класса. Поскольку s\_value существует независимо от любых объектов класса, то доступ к нему осуществляется напрямую через имя класса и оператор разрешения области видимости (в данном случае, через Anything::s\_value):



Обратите внимание, это определение статического члена не подпадает под действия [спецификаторов доступа](https://ravesli.com/urok-114-spetsifikatory-dostupa-public-i-private/): вы можете определить и инициализировать s\_value, даже если он будет private (или protected).

Зачем использовать статические переменные-члены внутри классов? Для присваивания уникального идентификатора каждому объекту класса (как вариант).

**+**