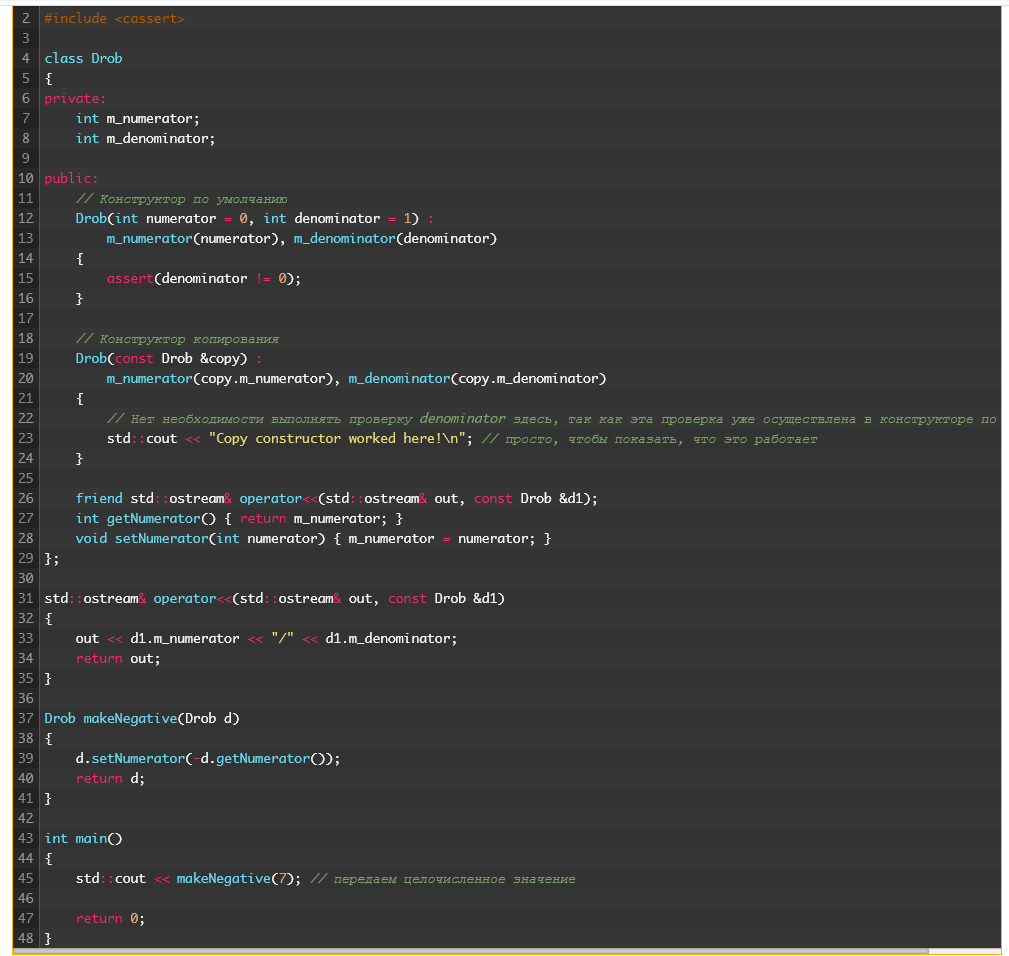
## Конструкторы преобразования, ключевые слова explicit и delete

По умолчанию язык C++ обрабатывает любой [конструктор](https://ravesli.com/urok-116-konstruktory/), как оператор [неявного преобразования](https://ravesli.com/urok-55-neyavnoe-preobrazovanie-tipov-dannyh/). Рассмотрим следующую программу:

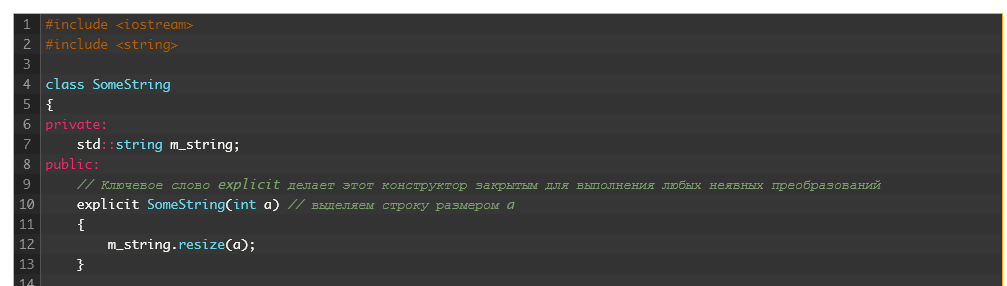


Хотя функция makeNegative() ожидает объект [класса](https://ravesli.com/urok-113-klassy-obekty-i-metody-klassov/) Drob, мы передаем ей целочисленный [литерал](https://ravesli.com/urok-36-literaly-magicheskie-chisla/) 7. Поскольку у класса Drob есть конструктор, который может принимать одно целочисленное значение (конструктор по умолчанию), то компилятор выполнит неявную конвертацию литерала 7 в объект класса Drob. Это делается путем выполнения [копирующей инициализации](https://ravesli.com/urok-142-kopiruyushhaya-initsializatsiya/) параметра d функции makeNegative() с помощью конструктора Drob(int, int).

**Ключевое слово explicit**

Иногда выполнение неявных преобразований может иметь смысл, а иногда может быть крайне нежелательным и генерировать неожиданные результаты

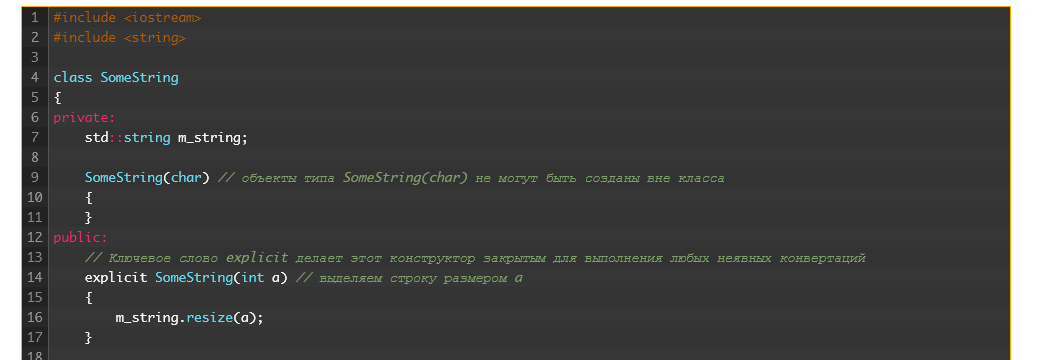
Один из способов решения этой проблемы — сделать конструктор явным, используя ключевое слово explicit (которое пишется перед именем конструктора). Явные конструкторы (с ключевым словом explicit) не используются для неявных конвертаций:



Однако использование явного конструктора только предотвращает выполнение неявных преобразований. [Явные конвертации](https://ravesli.com/urok-56-yavnoe-preobrazovanie-tipov-dannyh-operatory-casts/) (через операторы явного преобразования) по-прежнему разрешены.

## **Ключевое слово delete**

Еще одним способом запретить конвертацию 'a' в SomeString (неявным или явным способом) является добавление [закрытого](https://ravesli.com/urok-114-spetsifikatory-dostupa-public-i-private/) конструктора SomeString(char):



Тем не менее, этот конструктор все еще может использоваться внутри класса (private закрывает доступ к данным только для объектов вне тела класса).

Лучшее решение — использовать ключевое слово delete (добавленное в C++11) для удаления этого конструктора:

