1. **Конструктор** — это особый метод, инициализирующий

экземпляр своего класса.

**-конструктор умолчания** (default constructor)

В отсутствие явно заданного конструктора по умолчанию его код генерируется компилятором (что на исходном тексте, естественно, не отражается).

**-конструктор преобразования** (conversion constructor)

Конструктор, принимающий один аргумент. Задаёт преобразование типа своего аргумента в тип конструктора. Такое преобразование типа неявно применяется только если оно уникально.

**-конструктор с двумя и более параметрами** (parameterized

constructors)

Параметризованный конструктор может быть вызван явно или неявно, например:

Example e = Example(0, 50); *// явный вызов*

Example e(0, 50); *// неявный вызов*

**-конструктор копирования** (copy constructor)

Конструктор, аргументом которого является **ссылка** на объект того же класса. Применяется в [C++](https://ru.wikipedia.org/wiki/C%2B%2B) для передачи объектов в функции [по значению](https://ru.wikipedia.org/wiki/%D0%92%D1%8B%D0%B7%D0%BE%D0%B2_%D0%BF%D0%BE_%D0%B7%D0%BD%D0%B0%D1%87%D0%B5%D0%BD%D0%B8%D1%8E).

Конструктор копирования в основном необходим, когда объект имеет указатели на объекты, выделенные в [куче](https://ru.wikipedia.org/wiki/%D0%9A%D1%83%D1%87%D0%B0_(%D0%BD%D0%B5%D1%80%D0%B0%D1%81%D0%BF%D1%80%D0%B5%D0%B4%D0%B5%D0%BB%D1%91%D0%BD%D0%BD%D0%B0%D1%8F_%D0%BF%D0%B0%D0%BC%D1%8F%D1%82%D1%8C)). Если программист не создаёт конструктор копирования, то компилятор создаст **неявный** конструктор копирования, который копирует указатели **как есть**, то есть фактическое копирование данных не происходит и два объекта ссылаются на одни и те же данные в куче.

**-конструктор перемещения** (move constructor)

В [C++11](https://ru.wikipedia.org/wiki/C%2B%2B11) появился новый тип неконстантных [ссылок](https://ru.wikipedia.org/wiki/%D0%A1%D1%81%D1%8B%D0%BB%D0%BA%D0%B0_(C%2B%2B)) обозначаемый как T&&, и новый вид конструкторов — конструкторы перемещения ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *move constructors*). Конструктор перемещения принимает на входе значение неконстантной ссылки на объект класса, и используется для передачи владения ресурсами этого объекта. Конструкторы перемещения были придуманы для решения проблемы потери эффективности, связанной с созданием временных объектов.

**Правило пяти -** правило в [C++](https://ru.wikipedia.org/wiki/C%2B%2B), гласящее, что если [класс](https://ru.wikipedia.org/wiki/%D0%9A%D0%BB%D0%B0%D1%81%D1%81_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)) или структура [определяет](https://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D1%8A%D1%8F%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D0%B5_(%D0%B8%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)) один из следующих методов, то они должны явным образом определить все пять метода.

* Деструктор
* Конструктор копирования
* Оператор присваивания копированием
* Конструктор перемещения
* Оператор присваивания перемещением

Всегда стоит избегать дублирования одного и того же кода, так как при изменении или исправлении одного участка, придётся не забыть исправить остальные. Идиома копирования и обмена позволяет этого избежать, используя повторно код конструктора копирования.

1. **Операция присваивания копированием** - особый вид операции присваивания, используемый для присваивания объектов одного класса друг другу. Является одним из [особых членов-функций](https://ru.wikipedia.org/w/index.php?title=%D0%9E%D1%81%D0%BE%D0%B1%D1%8B%D0%B5_%D1%87%D0%BB%D0%B5%D0%BD%D1%8B-%D1%84%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D0%B8&action=edit&redlink=1) и генерируется автоматически [компилятором](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D0%B8%D0%BB%D1%8F%D1%82%D0%BE%D1%80) в случае, если нет явного объявления программистом. Код, сгенерированный компилятором, выполняет побитовое копирование.

Операция присваивания копированием отличается от [конструктора копирования](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BD%D1%81%D1%82%D1%80%D1%83%D0%BA%D1%82%D0%BE%D1%80_%D0%BA%D0%BE%D0%BF%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F) тем, что должна очищать члены-данные цели присваивания (и правильно обрабатывать самоприсваивание), тогда как конструктор копирования присваивает значения неинициализированным членам-данным.

1. **Оператор** в **С++** - это некоторое действие или функция обозначенная специальным символом.
2. **+, -, \*, =, >, <, <=, >=, +=, -=, \*=**
3. Бинарный оператор - это функция от двух параметров, параметрами которой являются левый и правый операнды оператора.

Когда перегруженный оператор является методом класса, тип первого операнда должен быть указателем на данный класс (всегда this), а второй должен быть объявлен в списке параметров.

Matrix operator + (const Matrix &matr) const; // Сложение матрицы

1. Если оператор определяется как отдельная функция (часто

такие операторы определяются дружественными функциями

Число параметров функции должно соответствовать арности

оператора. Например для бинарных операторов (+, -, \* и др)

параметра два.

// этот оператор дружественный для класса Matrix

// поэтому имеет доступ к его закрытым членам

**friend Matrix operator + (const Matrix &a, const Matrix &b);**

1. Унарные операторы и бинарные операторы типа “X=”

рекомендуется реализовывать в виде методов класса, а прочие

бинарные операторы — в виде дружественных функций.

Так стоит делать потому, что оператор-метод всегда

вызывается для левого операнда.

1. Вектор - контейнер с произвольным доступом, лист - с последовательным. Если нужно хаотично выбирать элементы из контейнера, или как-то хитро его сортировать, нужен вектор.

Расположение данных в памяти у вектора последовательное, а значит вы сможете получить доступ к любому элементу вектора очень быстро (с помощью []). Зато добавление элемента, например, в середину вектора - очень затратно. Ведь нужно сдвинуть остальные данные.