Конструктори

В c++ начинът за инициализиране на инстанция на клас е **конструкторът**. Той представлява *специален* метод на класа, който се различава от останалите методи на класа по следните начини:

- Името на конструктора съвпада с името на класа;
- Конструкторите са методи, които не връщат нищо;
- Когато се създава обект на класа, автоматично се извиква конструктор;
- Ако експлицитно не създадем конструктор, компилаторът ни създава конструктор по подразбиране, който няма параметри и инцилизира член-данните на класа с произволни стойности;

Има няколко вида конструктори:

- Конструктор по подразбиране (default constructor):

```
class Example {
private:
    int data;
public:
    Example();  //декларираме default конструктор
};

Example::Example() {  //дефинираме default конструктор
    data = 0;
}

Example a;  //създава се нова инстанция на класа Example и се инициализира посредством default конструктор
```

  Ако не създадем default-ен конструктор експлицитно, компилаторът ще ни създаде свой default-ен конструктор и ще присвои на член-данните произволни стойности. (Ще има допълнение!)

- Конструктор с параметри (parametrized constructor):

```
class Example {
private:
    int data;
public:
    Example();  //декларираме default конструктор
    Example(int data);  //декларираме parametrized конструктор
};

Example::Example() {  //дефинираме default конструктор
    data = 0;
}

Example::Example(int data) {  //дефинираме parametrized конструктор
    this->data = data;
}
```

```
Example a;  //създава се нова инстанция на класа Example и се инициализира посредством default конструктор
Example b(3213);  //създава се нова инстанция на класа Example и се инициализира посредством parametrized конструктор, като се присвоява стойноста 3213 на член-данната data
```

В c++ имаме възможност да приложим поговорката за "един удар, два заека" в контекста на конструкторите:

```
class Example {
private:
    int data;
public:
    Example(int data = 0);
};

Example::Example(int data = 0) {
    this->data = data;
}

Example a;  //data = 0;
Example b(3213); //data = 3213;
```

- Конструктор за копиране (copy constructor):

  Основно приложение - извиква се, когато за инициализация на обект от даден клас искаме да ползване вече създаден обект от същият клас. Ако не се създаде от програмиста конструктор за присвояване, компилатора създава собствен.

  Още кога се извиква:

  - Когато обект на клас се връща от функция по стойност. (компилаторът някой път оптимизира този случай, но идеята е, че се ползва в този случай copy конструкто)
  - Когато обект на клас се подава като аргумент на функция по стойност.

  Синтаксис

```
//В main функция
Example a(66666);
Example copy_of_a = a; //тук се извиква copy constructor
```

```
//В декларацията на класа
Example(const Example & other);
```

  Мално тълкуване на параметъра, че не съм сигурен дали добре ви го представих:

  - Example - означава, че приема като аргунмент обект от тип Example
  - *&* - означава, че подава обекта по референция,т.е. ще работи не с копие на данните му, а с лично неговите данни в паметта
  - *const* - означава, че този обект, който е подаден като параметър не може да бъде променян
  - *other_object* - име за именуване на обекта в обхвата на функцията

```
//Дефиниция на copy конструктор
Example::Example(const Example& other) {
    this->data = other.data;
}
```