

Упражнение 6

Функции в математиката.

$$\begin{array}{lll} f(x) = x^2 & x \in \mathbb{N} & f: \mathbb{N} \rightarrow \mathbb{N} \\ f(x, y) = x + y & x, y \in \mathbb{R} & f: \mathbb{R}^2 \rightarrow \mathbb{R} \\ f(x, y, \varepsilon) = |x - y| < \varepsilon & x, y, \varepsilon \in \mathbb{R} & f: \mathbb{R}^3 \rightarrow B, \text{ където } B = \{0, 1\} \end{array}$$

Функции в C++.

Чрез функции разделяме кода на програмата ни на отделни части, които можем да **преизползваме** вместо да повтаряме код.

Синтаксис:

```
<тип> <идентификатор/име>(<параметри>)  
{  
    <тяло>  
}
```

// Забележка: Параметрите се изреждат със запетаи и за всеки от
// тях се упоменава тип и име на параметъра

Пример аналог с функциите в математиката:

```
int sum(int x, int y) {  
    return x + y;  
}
```

$$\begin{array}{ll} f(x, y) = x + y & \text{където} \\ x, y \in \mathbb{R} & f: \mathbb{R}^2 \rightarrow \mathbb{R} \end{array}$$

Прототип (декларация) на функция.

Казваме, че има такава функция и тя е дефинирана някъде в нашата програма.

```
<тип> <име на функцията>(<параметри>);
```

// Забележка: Тук параметрите са <тип> <име> или само <тип>

Дефиниция на функция.

Тук вече сме дефинирали функцията т.е. какво прави и какво връща като резултат.

```
<тип> <име на функцията>(<параметри>)  
{  
    <тяло на функцията>  
}
```

// Забележка: Всяка функция трябва да връща “нещо”.

// Изключение правят void функциите.

Параметри и аргументи.

Параметър - името на променливата, която се използва във функцията.

Аргумент - стойността на параметъра, подаден при извикването на функцията.

Стойности по подразбиране.

<тип> <име>(<параметър> = <стойност по подразбиране>);

Параметрите със стойност по подразбиране се пишат най-отзад на списъка с параметри. Стойност по подразбиране може да е всякакъв израз, който може да се изчисли по време на компилация.

Псевдоними като параметри.

Функция може да има псевдоним като параметър, което означава, че ще работи **директно** с клетката в паметта (вместо да прави копие на стойността), където е заделена променливата и всякаква промяна на тази променлива ще бъде отразена и извън функцията.

Пример:

```
void makeNegative(int& num)
{
    num = -num;
}
```