

# Функции

ОСНОВЫ C++



# Что будет на уроке

1. Узнаем всё, или почти всё о функциях. Аргументы, параметры, возвращаемые значения.
2. Научимся описывать указатели на функции и функции обратного вызова.
3. Рассмотрим inline функции и механизм перегрузки функций.
4. Изучим, зачем нужны пространства имён

# Функция - это обособленный, переиспользуемый участок кода

## Без функции

простой код 1

простой код 2

**сложный код 1**

**сложный код 2**

**сложный код 3**

**сложный код 4**

простой код 3

**сложный код 1**

**сложный код 2**

**сложный код 3**

**сложный код 4**

простой код 4

простой код 5

## С функцией

**функция**

**сложный код 1**

**сложный код 2**

**сложный код 3**

**сложный код 4**

простой код 1

простой код 2

**вызов функции**

простой код 3

**вызов функции**

простой код 4

простой код 5

**Объявление функции** - это указание программе на то, что функция существует и будет определена. Объявление иногда называют **прототипом функции**.

**Определение функции** - это описание тела функции, то есть непосредственно того кода, который будет исполняться при вызове функции. Часто определение совмещают с объявлением.

# Параметры функции

- Описание принимаемых в функцию значений;
- Значения помещаются в локальные переменные;
- С передачей по адресу всё чуть сложнее;
- Параметров может быть любое количество;
- Функция может быть без параметров;
- Параметры функции всегда lvalue;
- Параметры могут иметь значения по умолчанию;

**Вызов функции** - это указание программе выполнить код внутри **определённой** функции. Передаваемые в функцию при вызове значения называются **аргументами**.

- вызвать неопределённую функцию нельзя
- типы аргументов всегда повторяют типы параметров
- порядок указания аргументов всегда повторяет порядок указания параметров
- если есть параметр по умолчанию, аргумента может не быть
- аргументы функции всегда rvalue

# Переменное число параметров

Так называемый **vararg**, вариативный аргумент, используется когда неизвестно, сколько будет передано аргументов при вызове функции.

Например, функция `printf()`;

# Передача аргументов по ссылке и по указателю

- сами данные из аргумента не копируются внутрь функции;
- по ссылке обычно передают строки и объекты;
- по указателю обычно передают массивы и структуры;
- для доступа к данным указатель нужно разыменовывать;
- при передаче по указателю аргумент - адрес объекта;
- при передаче по ссылке аргумент - сам объект.



**Возвращаемые значения** - это значения, полученные в результате работы функции и подставляемые в качестве rvalue на месте её вызова.

- Тип функции определяют по типу возвращаемого значения
- Возвращаемое значение всегда строго типизировано
- В C++ нельзя не указывать тип возвращаемого значения
- Функция может не возвращать значений, тогда она имеет специальный тип **void** (англ. пустота)

# Перегрузка функций

Это описание функций с одинаковыми названиями;  
Главное, чтобы различались параметры;  
Тип функции может не отличаться (это не важно);  
Потом это будут использовать как часть полиморфизма в ООП;  
Какую перегрузку вызвать программа поймёт по аргументам.

# Указатели на функции

- вызов функции - это разыменование указателя на функцию
- имя функции - это тоже указатель на функцию
- указатель на функцию можно передать в качестве аргумента другой функции (дальше из этого придумали функциональное программирование)
- иногда удобно создать массив указателей на функции

**Пространство имён** - это явное создание общей области видимости для переменных и функций. Одно и то же пространство имён может быть объявлено в нескольких файлах, тогда видимость объектов внутри этого пространства сохранится.