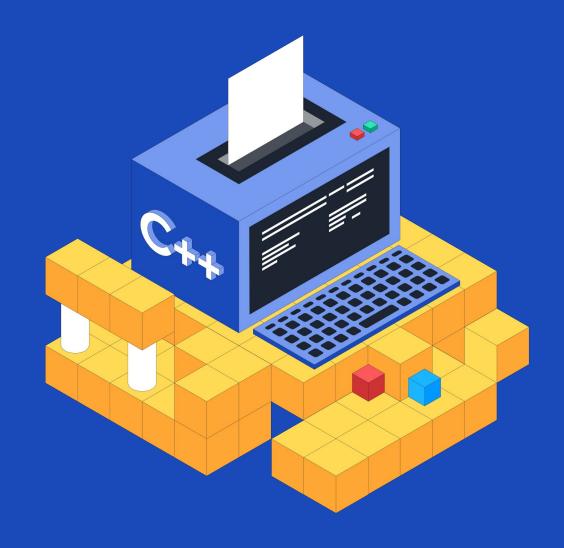




ПРОГРАММИРОВАНИЕ НА ЯЗЫКЕ C++

Материалы подготовлены отделом методической разработки

Начальный уровень









С+.1.12. Классы / ООП









Классы

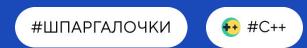
Классы - это пользовательский тип данных. Чтобы создать класс, нужно написать ключевое слово **class** и имя класса. Имя класса всегда пишется с большой буквы:

class Animal {

};



В классе (в фигурных скобках) можно создавать переменные (свойства класса) и функции (методы класса). Но прежде нужно указать спецификатор доступа.





Спецификаторы доступа

Спецификатор или уровень доступа определяет, кто может использовать свойства и методы класса. Всего есть 3 спецификатора:

```
public - доступ к членам класса не ограничен;
private - доступ к членам класса извне закрыт;
protected - доступ к членам класса открыт только дружественным и дочерним классам.
Лучше всего делать свойства класса закрытыми, а методы - открытыми:
class Animal {
       private:
             string name;
      public:
             void setName(string n) {
                    name = n;
```







Объекты

Объект - это переменная пользовательского типа данных (созданного класса). Чтобы создать объект, нужно написать имя класса и имя переменной:

Animal cat;

А чтобы получить доступ к члену класса, его нужно указать через точку после имени объекта:

<имя_объекта>.<свойство или метод>;

Например:

cout << cat.name; // сработает, только если свойство name открытое (public)

cat.setName("Kitty");







ООП

ООП или **объектно-ориентированное программирование** - подход в программировании, основанный на разделении программы на объекты и описании (определении) этих объектов.

Объектно-ориентированное программирование позволяет сделать код более понятным и структурированным. В ООП есть 3 основных принципа: **инкапсуляция**, **наследование** и **полиморфизм**.







Инкапсуляция - это свойство, позволяющее объединить в классе свойства и методы, работающие с ними. А также скрыть детали реализации от пользователя. В С++ инкапсуляция реализована с помощью спецификаторов доступа.









Наследование - это свойство, позволяющее создать новый класс-потомок на основе уже существующего, при этом все характеристики класса-родителя присваиваются классупотомку. Чтобы создать класс-потомок, после его имени нужно поставить двоеточие и имя класса-родителя:

class Cat: public Animal







Полиморфизм - это свойство классов, позволяющее использовать объекты классов с одинаковым интерфейсом, но разным поведением для разных классов. Например, полиморфизм позволяет переопределить поведение метода в классе-потомке:

```
class Animal {
      ---
     void say() {
           cout << "My name is " << name;</pre>
class Cat: public Animal {
     void say() {
           cout << "Meow";
```







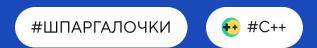
Объявление и реализация метода

В C++ можно разделить объявление метода и его реализацию, это помогает сделать класс короче. Пример объявления метода **set** без реализации:

```
class Company {
    ...
    public:
        void set(string companyName);
};
```

Реализация в таком случае прописывается уже после класса. Помимо типа и имени метода, указывается имя класса, в котором он был объявлен:

```
void Company::set(string companyName) {
    name = companyName;
```





Конструктор класса

Конструктор класса - это специальный метод, который запускается в момент создания класса и имеет точно такое же название, как и класс. С помощью конструктора можно задать начальные значения свойствам класса. Важно: при создании конструктора не нужно указывать тип данных:

```
class Company {
    ...
    Company();
};

Company::Company() {
    money = 0;
    cin >> name;
}
```