Объектно-ориентированное программирование (ООП)

Основы алгоритмизации и программирования

Что такое ООП?

Объе́ктно-ориенти́рованное программи́рование (сокр. ООП) — методология программирования, основанная на представлении программы в виде совокупности взаимодействующих объектов, каждый из которых является экземпляром определённого класса, а классы образуют иерархию наследования.

Класс - это абстрактный тип данных. С помощью класса описывается некоторая сущность (характеристики и возможные действия).

Объект - это уже конкретный экземляр класса и того, что он представляет.

Основные принципы ООП

- Инкапсуляция
- Наследование
- Полиморфизм
- Абстракция

Инкапсуляция

Инкапсуляция - это свойство системы, позволяющее объединить данные и методы, работающие с ними, в классе и скрыть детали реализации от пользователя.

В классе могут быть реализованы внутренние вспомогательные методы и поля, устройство которых пользователю знать не нужно.

Наследование

Наследование - это свойство системы, позволяющее описать новый класс на основе уже существующего с частично или полностью заимствующейся функциональностью.

Класс, на базе которого создаётся новый класс называется **базовым,** а базирующийся новый класс - **наследником**.

Благодаря наследованию нам не обязательно переписывать весь набор свойств для нескольких объектов если они похожи: достаточно описать один класс а затем создавать его наследников, убирая, модифицируя и/или добавляя те свойства, которые необходимы нам.

Полиморфизм

Полиморфизм - это способность объектов с одним интерфейсом иметь различную реализацию.

Например, есть 2 класса, **круг** и **квадрат**. У обоих классов есть метод **getsquare(),** который считает и возвращает площадь. Но площадь круга и квадрата рассчитываются по разным формулам, соответственно, реализация одного и того же метода различная.

Абстракция

Абстракция - позволяет выделять из некоторой сущности только необходимые характеристики и методы, которые в полной мере (для поставленной задачи) описывают объект.

Например, создавая класс для описания студента, мы выделяем только необходимые его характеристики, такие как ФИО, номер зачётной книжки, номер группы. Здесь нет смысла добавлять поле “вес” или “имя кота/собаки” и т.п.

Понятие и описание класса

Класс - это абстрактный тип данных; - некоторый шаблон, на основе которого будут создаваться его экземпляры - **объекты**.

```
class имя_класса {
    модификатор_доступа:
        <тут уже всякие методы>
}
```

P.S. Классы описываются также как и функции - **вне** main().

Что может включаться в класс?

- поля;
- константы;
- свойства;
- конструкторы;
- методы;
- события;
- операторы;
- индексаторы;
- вложенные типы.

Модификаторы доступа

В языке C++ уровень доступности определяется с помощью специальных модификаторов private, protected, public. Применение этих модификаторов может быть использовано в двух случаях:

- в случае объявления элементов класса.
- в случае наследования классов.

Описание модификаторов

- Модификатор **private**

  Элементы доступны только методам, реализованным в классе, а также дружественным функциям и их методам.

- Модификатор **protected**

  Элементы доступны только методам, реализованным в классе, также дружественным функциям, методам и наследникам базового класса.

- Модификатор **public**

  Элементы доступны всем методам в программе.