Оглавление

[Общее для классов: 2](#_Toc147524955)

[Объект: 2](#_Toc147524956)

[Класс: 2](#_Toc147524957)

[Экземпляр: 2](#_Toc147524958)

[Атрибут: 2](#_Toc147524959)

[Метод: 2](#_Toc147524960)

[Инкапсуляция: 2](#_Toc147524961)

[Организация работы между классами: 3](#_Toc147524962)

[Наследование: 3](#_Toc147524963)

[Полиморфизм: 3](#_Toc147524964)

[Полиморфизм типа Ad hoc: 3](#_Toc147524965)

[Полиморфизм типа Параметрический: 3](#_Toc147524966)

[Апкаст: 3](#_Toc147524967)

[Даункаст: 3](#_Toc147524968)

[Композиция: 3](#_Toc147524969)

[Вложенные классы: 3](#_Toc147524970)

[Мета классы 3](#_Toc147524971)

[Агрегация: 3](#_Toc147524972)

# Общее для классов:

Здесь описано общие для классов

## Объект:

Объект (объект класса) – единица информации в памяти. Хранит конкретные значения свойств и информацию о принадлежности к классу.

1. Может выполнять методы (методы – функции класса).

## Класс:

Класс - Описывает модель объекта, его свойства и поведение.

1. Говоря языком программиста, класс — такой тип данных, который создается для описания сложных объектов.

## Экземпляр:

Экземпляр – Конкретный объект каво-то класса.

1. Для краткости вместо «Объект, порожденный классом "Стул", говорят «экземпляр класса "Стул"».

## Атрибут:

Атрибут – Все имена в классе: переменные и методы (добавляются с помощью конструкции: self.”имя переменной”). Свойство, присущее объекту.

Если атрибут определен в самом классе (имеется с самого начала не считая методов) то он является атрибутом класса. При изменении атрибута класса, атрибут изменится у всех экземпляров

При создании атрибута в методе или вне класса, считается атрибутом экземпляра

1. Класс объекта определяет, какие атрибуты есть у объекта.
2. Конкретные значения атрибутов — характеристика уже не класса, а конкретного экземпляра этого класса (то есть объекта).

## Метод:

Метод – Функция в классе для воздействия с экземпляром класса. Действие, которое объект может выполнять над самим собой или другими объектами.

## Инкапсуляция:

Инкапсуляция – Позволяет сделать класс единым целым. Класс превращается в капсулу со всем необходимым для его функционирования. Работать из вне с классом можно только через разрешённые методы и атрибуты (ибо иначе не позволит инкапсуляция)

# Организация работы между классами:

Описывает то, как можно организовать работу между классами

## Наследование:

Наследование – позволяет вынести общее (методы и атрибуты) для нескольких классов в другой класс что позволяет избавится от дублирования кода.

## Полиморфизм:

Возможность через единый интерфейс (класс) работать с объектами разных классов.

Позволяет исполнять один и тот же код для данных разных классов

### Полиморфизм типа Ad hoc:

Реализуется через перегрузку функций

### Полиморфизм типа Параметрический:

Реализуется через метод класса родителя к классу наследнику

## Апкаст:

Апкаст - приведение дочернего типа к родительскому по иерархии наследования.

## Даункаст:

Даункаст - приведение родительского типа к дочернему.

## Композиция:

Композиция – запуск одного класса в другом

## Вложенные классы:

Такие классы, которые реализованы в другом классе

## Мета классы

Мета классы – это классы, которые создают классы. В Python, классы сами являются объектами, и мета классы отвечают за их создание. Они определяют правила и поведение, которыми должны обладать классы, создаваемые с их помощью.

Данные мета классы нельзя создавать динамически (в коде инициализируются один раз когда происходит запуск программы)

Для создания мета класса необходимо обычному классу Python наследовать класс type.

СТАРАТСЯ ИСПОЛЬЗОВАТЬ КАК МОЖНО МЕНЬШЕ.

## Агрегация:

Хранение объекта класса в качестве атрибута другого объекта класса