

Лекция 13

Объектно-Ориентированное Программирование

Основные понятия. Объект

Объект - конкретная сущность предметной области.

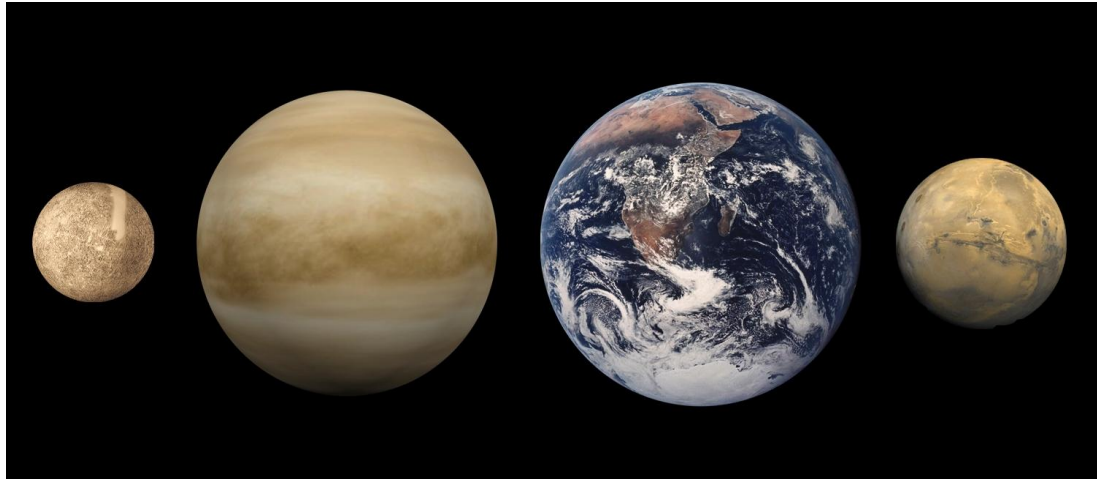


Основные понятия. Класс

Класс - это тип объекта. Или говорят, что объект - экземпляр класса.

Класс: **Планета.**

Объекты: **Меркурий, Венера, Земля, Марс.**



Основные понятия. Методы класса

- Метод класса - функция, которая принадлежит классу.
- В языке Python первый аргумент метода - `self`, экземпляр класса. При описании метода пишется явно.
- Методы имеют доступ к полям экземпляра класса.
- Пример:

class Student:

```
def __init__(self): # метод класса Student; self - текущий экземпляр класса Student  
    pass
```

Основные понятия. Поля объекта

- Поле (атрибут) объекта - некоторая переменная, которая лежит в области видимости объекта и доступна во внешней программе через синтаксис:

`<имя_объекта>.<поле>`

- Поля объекта устанавливаются в методах класса через обращение к экземпляру `self`, например:

```
def __init__(self):
```

```
    self.name = 'Fedor' # поле экземпляра класса Student
```

Основные понятия. Создание простого класса

```
class Student:
```

```
    def __init__(self): # конструктор; self - текущий экземпляр класса Student
```

```
        self.name = 'Fedor' # поле экземпляра класса Student
```

```
        name = 'Petr' # локальная переменная в конструкторе
```

```
new_student = Student() # создание экземпляра класса Student
```

```
print(new_student.name)
```

Основные понятия. Конструктор

- Конструктор - метод, который вызывается при создании экземпляра класса.
- Конструктор ничего не возвращает.
- Конструктор может быть не описан, тогда создастся пустой объект.
- Конструктор может быть только один, но может иметь переменные по умолчанию.

Основные понятия. Поля класса

```
class Counter:
```

```
    counter = 0 # Поле класса
```

```
    def __init__(self):
```

```
        self.data = 'some data' # Поле экземпляра класса
```

```
a = Counter()
```

```
b = Counter()
```

```
print(a.counter, b.counter, Counter.counter)
```


Основные понятия. Наследование

- Повторное использование и последующее расширение одним классом атрибутов другого класса.
- Класс, определённый через наследование от другого класса, называется: производным классом, классом-потомком, подклассом, классом-наследником. Класс, от которого новый класс наследуется, называется: предком, базовым классом или суперклассом.
- Пример в Python:

```
class Super: # Родитель
```

```
    def __init__(self):  
        self.data = 'some data'
```

```
class Subclass(Super): # Потомок
```

```
    pass
```

isinstance() и isinstance()

➤ isinstance(obj1, class1)

Возвращает True, если **obj1** является экземпляром класса **class1** или суперкласса класса **class1**

➤ isinstance(class1, class2)

Возвращает True, если **class1** является наследником класса **class2**.

super()

- Иногда в процессе написания метода в классе наследнике может понадобиться вызвать метод суперкласса. Это можно сделать через имя суперкласса или через функцию `super()`.

[illegible]

Основные понятия. Перегрузка операторов

- В python существует возможность переопределения не только методов класса, но и операторов.
- Вы можете создать свой тип данных и определить для его экземпляров операции сложения/умножения/извлечения среза и т.д.

Источники и полезные ссылки

- Классы в Python, документация

<https://docs.python.org/3/tutorial/classes.html>

- Операторы языка python, которые вы можете переопределить:

<https://docs.python.org/3/reference/datamodel.html>

- Курс “Python: основы и применение” <https://stepik.org/course/512/syllabus>