

Практическая работа 10 - Основы ООП. Создание класса и объекта

ООП – объектно-ориентированное программирование. Является одной из наиболее важных концепций языка Python.

ООП, или объектно-ориентированное программирование, является парадигмой программирования, основанной на концепции "объектов". Объекты — это сущности, которые объединяют в себе как данные, так и методы для работы с этими данными.

Основные характеристики и принципы ООП включают:

Инкапсуляция:

Это сокрытие внутренней структуры объекта от внешнего мира. Данные объекта хранятся в полях (атрибутах), а доступ к этим данным осуществляется через методы. Это позволяет изолировать функционал объекта и уменьшить взаимозависимость компонентов системы.

Наследование:

Это механизм, позволяющий описывать новый класс на основе уже существующего (родительского), перенимая его свойства и функционал. Наследование позволяет расширять и модифицировать функционал уже существующих классов, способствует повторному использованию кода.

Полиморфизм:

Означает способность методов обрабатывать данные разных типов по-разному. Полиморфизм позволяет использовать один и тот же интерфейс для различных базовых форм данных (например, функции, которые могут принимать объекты разных классов).

Абстракция:

Это процесс выделения важных характеристик объекта и игнорирование его несущественных деталей. Абстракция позволяет сосредоточиться на том, что объект делает, а не на том, как он это делает.

ООП простыми словами

Поскольку на примере все усвоить гораздо проще, то давайте за пример возьмем робота, которого постараемся описать за счёт классов в ООП.

Класс в случае с роботом – это его чертёж. Экземпляр класса (объектом) называет целый робот, который создан точно по чертежу.

Наследование – это добавление полезных опций к чертежу робота. К примеру, берем стандартный чертёж робота и дорисуем к нему лазеры, крылья и броню. Все эти дорисовки мы сделаем в классе наследнике, основной функционал которого взят из родительского класса.

Полиморфизм – это общий функционал для всех роботов и не важно что каждый робот может очень сильно отличаться друг от друга. К примеру, в главном классе мы указываем возможность передвижения для всех последующих роботов. Далее в классе наследнике мы можем дополнительно указать возможность левитации для робота, в другом же классе укажем возможность передвижения по воде и так далее. Получается, что есть общий функционал что записан в главном чертеже, но его можно переписать для каждого последующего робота (для каждого наследника).

А **инкапсуляция** является для нас бронёй, защищающей робота. Под пластырем брони находятся уязвимые элементы, вроде проводов и микросхем. После прикрытия брешей с помощью брони (`protected` или `private`), робот полностью защищён от

внешних вмешательств. По сути, мы делаем доступ ко всем полям лишь за счёт методов, тем самым прямой доступ к полю будет закрыт.

У всех классов методы могут отличаться, как и поля с конструкторами. Каждый класс позволяет создавать любое количество разных объектов, все из них имеют собственные характеристики.

Создание классов

Для создания класса необходимо прописать ключевое слово `class` и далее название для класса. Общепринято начинать названия классов с буквы в верхнем регистре, но если этого не сделать, то ошибки не будет.

В любом классе можно создавать поля (переменные), методы (функции), а также конструкторы.

Создав новый класс и поместив туда какую-либо информацию мы можем создавать на основе него новые объекты. Объекты будут иметь доступ ко всем характеристикам класса.

Пример простого класса приведен ниже:

```
class Book:
```

```
    pass # Класс может ничего не возвращать
```

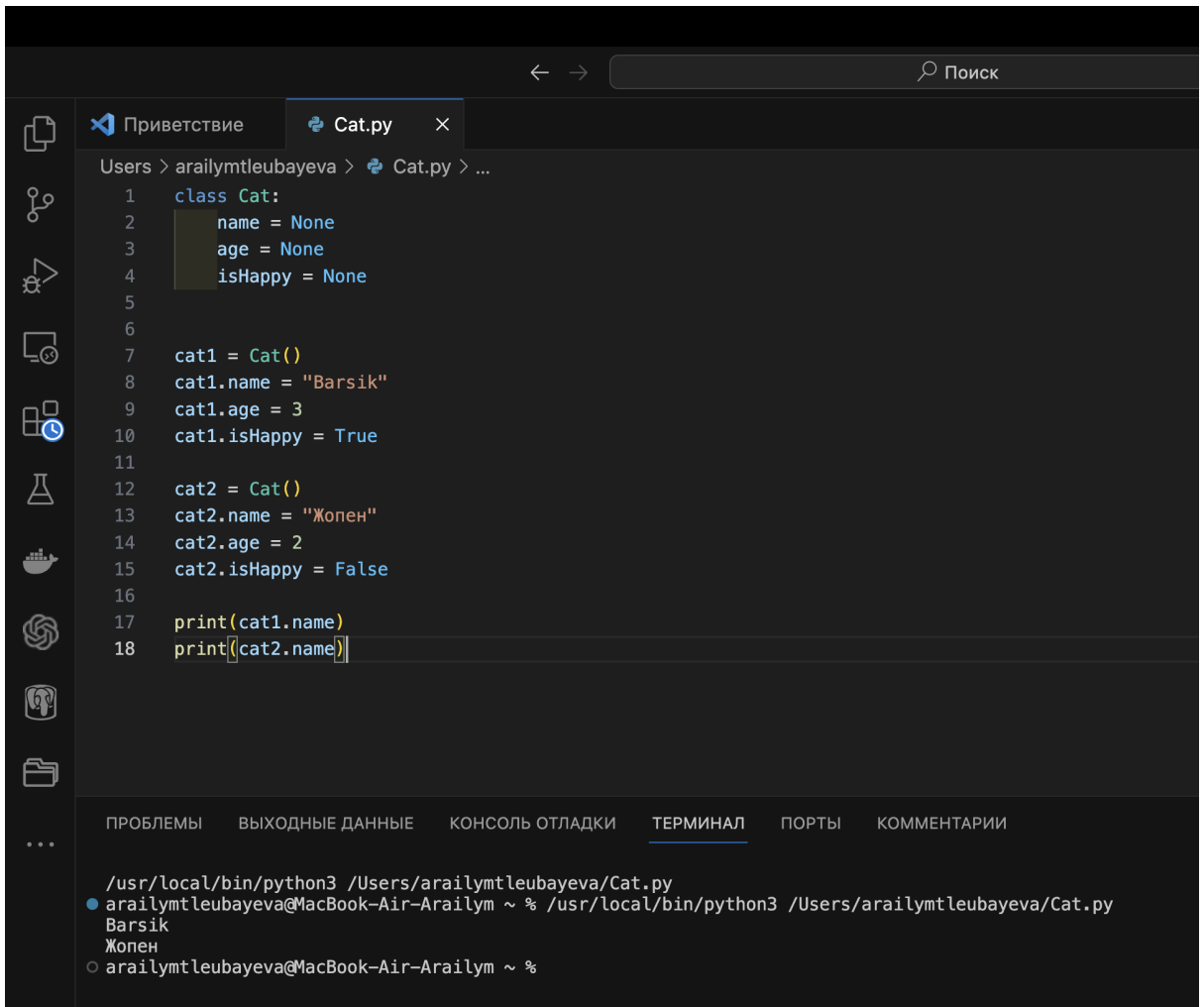
На основе такого класса мы можем создать множество объектов. Каждый объект в данном случае будет представлять из себя конкретную книжку. Для каждого объекта мы можем указать уникальные данные.

Чтобы создать объект нам потребуется следующий код:

```
obj_new = Some() # Создание объекта
```

```
obj_second = Some() # Создание 2 объекта
```

Простой класс Cat



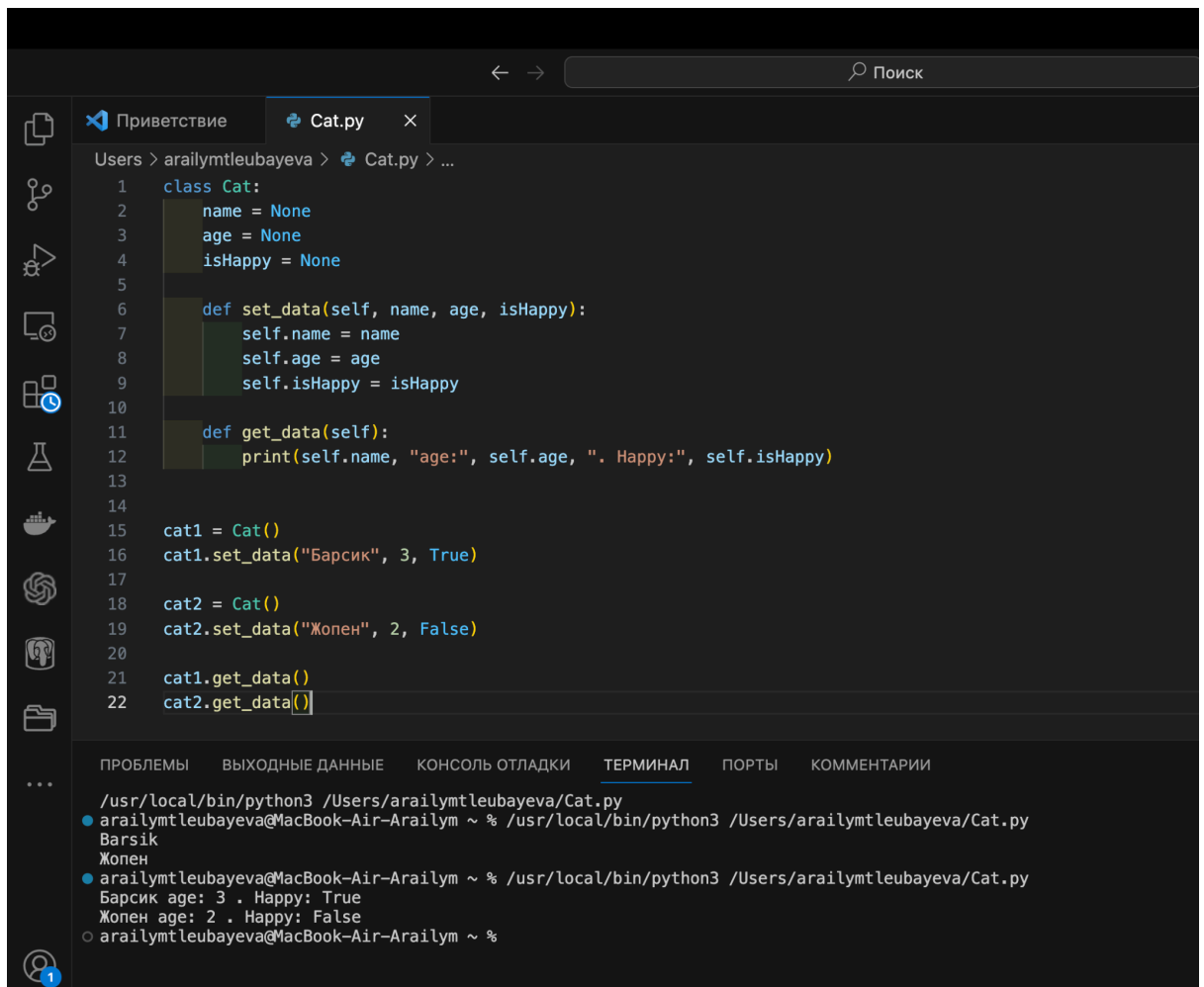
The image shows a screenshot of an IDE interface. At the top, there is a search bar with the text "Поиск". Below it, the file explorer shows a project named "Приветствие" and a file named "Cat.py". The main editor displays the following Python code:

```
1 class Cat:
2     name = None
3     age = None
4     isHappy = None
5
6
7 cat1 = Cat()
8 cat1.name = "Barsik"
9 cat1.age = 3
10 cat1.isHappy = True
11
12 cat2 = Cat()
13 cat2.name = "Жопен"
14 cat2.age = 2
15 cat2.isHappy = False
16
17 print(cat1.name)
18 print(cat2.name)
```

At the bottom, the "ТЕРМИНАЛ" (Terminal) tab is active, showing the execution of the script:

```
/usr/local/bin/python3 /Users/arailymtleubayeva/Cat.py
arailymtleubayeva@MacBook-Air-Arilym ~ % /usr/local/bin/python3 /Users/arailymtleubayeva/Cat.py
Barsik
Жопен
arailymtleubayeva@MacBook-Air-Arilym ~ %
```

Класс «Cat» с методами



```
1 class Cat:
2     name = None
3     age = None
4     isHappy = None
5
6     def set_data(self, name, age, isHappy):
7         self.name = name
8         self.age = age
9         self.isHappy = isHappy
10
11     def get_data(self):
12         print(self.name, "age:", self.age, ". Happy:", self.isHappy)
13
14
15 cat1 = Cat()
16 cat1.set_data("Барсик", 3, True)
17
18 cat2 = Cat()
19 cat2.set_data("Жопен", 2, False)
20
21 cat1.get_data()
22 cat2.get_data()
```

PROБЛЕМЫ ВЫХОДНЫЕ ДАННЫЕ КОНСОЛЬ ОТЛАДКИ ТЕРМИНАЛ ПОРТЫ КОММЕНТАРИИ

```
/usr/local/bin/python3 /Users/arailymtleubayeva/Cat.py
● arailymtleubayeva@MacBook-Air-Arailym ~ % /usr/local/bin/python3 /Users/arailymtleubayeva/Cat.py
Барсик
Жопен
● arailymtleubayeva@MacBook-Air-Arailym ~ % /usr/local/bin/python3 /Users/arailymtleubayeva/Cat.py
Барсик age: 3 . Happy: True
Жопен age: 2 . Happy: False
○ arailymtleubayeva@MacBook-Air-Arailym ~ %
```

Задание 1. Класс автомобиль

Создайте класс Автомобиль. В класс добавьте несколько полей:

- количество колес
- модель автомобиля
- скорость

Также добавьте методы:

- метод что получает три параметра и устанавливает их в поля класса
- метод, который выводит информацию про объект на экран
-

Создайте два объекта и используйте оба метода для каждого из объектов.

Задание 2. Создание полей класса

- Создайте пустой класс без методов и полей.

На основе класса создайте два объекта. Обратитесь ко второму объекту и создайте для него поле «newValue» со значением 5.

Задание 3. Небольшой класс

Создайте класс с одним текстовым полем, а также методом, что будет писать на экран значение поля + параметр переданный в него.

Создайте объект на основе класса и вызовите метод из класса.

Самостоятельная работа

Простые задачи(30 б)

1. Класс Точка:

Создайте класс Точка, который инициализируется с двумя атрибутами: x и y. Напишите метод, который выводит координаты точки на экран.

2. Класс Прямоугольник:

Создайте класс Прямоугольник, который принимает ширину и высоту в качестве параметров. Добавьте метод для расчета площади прямоугольника.

3. Класс Студент:

Создайте класс Студент с атрибутами для имени, фамилии и года обучения. Добавьте метод для вывода полной информации о студенте.

Средние задачи(40б)

1. Класс Автомобиль с наследованием:

Создайте класс Автомобиль с базовыми атрибутами. Затем создайте классы Грузовик и Легковой, которые наследуются от Автомобиль. Добавьте уникальные атрибуты для каждого подкласса.

2. Класс Счетчик:

Разработайте класс Счетчик, который инициализируется с начальным значением. Добавьте методы для увеличения, уменьшения счетчика и получения текущего значения.

3. Класс Телефонная книга:

Создайте класс ТелефоннаяКнига, который хранит имена и номера телефонов. Реализуйте методы для добавления нового контакта, удаления контакта по имени и поиска номера телефона по имени.

Сложные задачи (каждая задача по 10б)

1. Класс Банковский счет с инкапсуляцией:

Создайте класс БанковскийСчет, который скрывает детали баланса (сделайте его приватным). Реализуйте методы для внесения и снятия средств, а также проверки баланса.

2. Класс Книжная библиотека:

Создайте систему классов для управления книжной библиотекой. Включите классы для книг, читателей и самой библиотеки. Реализуйте функционал для регистрации книг, их выдачи и возврата.

3. Класс Геометрические фигуры с полиморфизмом:

Создайте абстрактный класс Фигура с методом для расчета площади. От него должны наследоваться классы Круг, Прямоугольник и Треугольник, каждый с собственной реализацией расчета площади.