

Практическая работа № 2

Базовые понятия объектно-ориентированного программирования

Цель работы: получить представление о базовых понятиях объектно-ориентированного программирования, получить навыки проектирования и создания классов.

Теоретические сведения

Python является мультипарадигмальным языком программирования, так как поддерживает различные подходы к программированию. Одним из таких подходов к решению задач является объектно-ориентированное программирование (далее – ООП).

В отличие от функционального программирования, где основной упор делается на функции, ООП делает акцент на объектах.

Объект – это набор данных и методов, которые управляют этими данными.

Класс и его атрибуты. Экземпляр класса

Представим себе в качестве объекта многоэтажный дом.

Класс – это схема объекта. Мы можем представить себе класс как эскиз (прототип) дома, который содержит все подробности об этажах, дверях, окнах и т.д. На основе этих описаний мы сможем построить реальный дом.

Как из чертежа дома можно сделать много домов, так и из класса можно создать много объектов из одного класса. Объект также называется экземпляром класса, а процесс его создания – инициализацией.

Определения классов начинаются с ключевого слова `class`.

Далее создадим пустой класс.

Листинг 1 – Пустой класс

```
class MyFirstClass:  
    pass
```

Как только мы определяем класс, создается новый объект класса с таким же именем. Этот объект класса позволяет нам получать доступ к различным атрибутам, а также инициализировать новые объекты этого класса.

Класс создает новое локальное пространство имен, в котором определяются все его атрибуты. Атрибуты могут быть данными (в таком случае их называют полями) или методами.

Методы – это функции, определяемые внутри тела класса. Они используются для определения поведения объекта.

Существуют также специальные атрибуты, которые начинаются с двойного подчеркивания `__`.

Особый интерес представляет метод `__init__`. Эта специальный метод вызывается всякий раз, когда создается новый объект данного класса.

В ООП методы такого типа называются конструкторами и обычно используются для инициализации начальных значений.

Ниже рассмотрим класс круга.

Листинг 2 – Класс круга

```
class Circle:

    def __init__(self, radius):
        self.radius = radius

    def get_area(self):
        return 3.14 * self.radius * self.radius
```

Процедура создания экземпляра класса похожа на вызов функции:

Листинг 3 – Создание экземпляра класса

```
circle = Circle(5)
```

После создания экземпляра класса мы можем получить доступ к его атрибутам:

Листинг 4 – Получение доступа к атрибутам класса через его экземпляр

```
print(circle.radius)
print(circle.get_area())
```

Практическая часть

После ознакомления с теоретической частью необходимо выполнить представленные ниже практические задания в соответствии с Вашим вариантом (вариант определяется Вашим порядковым номером в списке группы).

Практическое задание

Спроектировать и реализовать класс, который описывает объект, заданный Вашим вариантом.

Таблица 1 – Варианты для практического задания

| Вариант | Название объекта |
|---------|-------------------------------------|
| 1 | Обучающийся |
| 2 | Геометрическая фигура (тело) |
| 3 | Автомобиль |
| 4 | Город |
| 5 | Книга |
| 6 | Животное |
| 7 | Мяч |
| 8 | Река |
| 9 | Матч по какому-либо виду спорта |
| 10 | Учебный предмет |
| 11 | Пользователь какого-либо приложения |

Продолжение таблицы 1

| Вариант | Название объекта |
|----------------|-------------------------|
| 12 | Квартира |

Примечание – Класс должен содержать минимум два поля, являющихся разными типами данных, и иметь минимум два метода.