МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»
ИНСТИТУТ ЦИФРОВОГО РАЗВИТИЯ

Отчет по лабораторной работе № 14
«Элементы объектно-ориентированного программирования в языке Python»

по дисциплине «Основы программной инженерии»

Выполнила:
Первых Дарья Александровна,
2 курс, группа ПИЖ-б-о-20-1

Проверил:
Доцент кафедры
инфокоммуникаций, Воронкин Р.А.

Ставрополь, 2022 г.

ВЫПОЛНЕНИЕ

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-


class Book:
    material = "paper"
    cover = "paperback"
    all_books = []


if __name__ == '__main__':
    print(Book.material)
    print(Book.cover)
    print(Book.all_books)
```

Book

main

```
C:\Users\podar\study\anaconda\envs\14LR\py
paper
paperback
[]
```

Рисунок 1 – Атрибут класса

```python
class River:
    # список всех рек
    all_rivers = []

    def __init__(self, name, length):
        self.name = name
        self.length = length
        # добавляем текущую реку в список всех рек
        River.all_rivers.append(self)


if __name__ == '__main__':
    volga = River("Волга", 3530)
```

main

```
C:\Users\podar\study\anaconda\envs\14LR\python.exe "C:/
Волга
Сена
Нил
```

Рисунок 2 – Элементы класса

```python
class River:
    all_rivers = []

    def __init__(self, name, length):
        self.name = name
        self.length = length
        River.all_rivers.append(self)

    def get_info(self):
        print("Длина {0} равна {1} км".format(self.name, self.length))


if __name__ == '__main__':
    volga = River("Волга", 3530)
    seine = River("Сена", 776)
    nile = River("Нил", 6852)
    volga.get_info()
```

```
main ×
C:\Users\podar\study\anaconda\envs\14LR\python.exe "C:/Users/podar/study/PyCh
Длина Волга равна 3530 км
Длина Сена равна 776 км
Длина Нил равна 6852 км
```

Рисунок 3 – Атрибуты экземпляра

```python
        self.name = name
        self.capacity = capacity
        self.cargo = 0

    def load_cargo(self, weight):
        if self.cargo + weight <= self.capacity:
            self.cargo += weight
            print("Loaded {} tons".format(weight))
        else:
            print("Cannot load that much")

    def unload_cargo(self, weight):
        if self.cargo - weight >= 0:
            self.cargo -= weight
            print("Unloaded {} tons".format(weight))
        else:
```

```
if __name__ == '__main__'
main ×
Jack Sparrow is the captain of the Black Pearl
Jack Sparrow
Loaded 600 tons
Unloaded 400 tons
Cannot load that much
Cannot unload that much
```

Рисунок 4 – Изменение атрибутов с помощью методов

```
            self.__height = h
        else:
            raise ValueError

    def area(self):
        return self.__width * self.__height


if __name__ == '__main__':
    rect = Rectangle(10, 20)
    print(rect.width)
    print(rect.height)
    rect.width = 50
    print(rect.width)
    rect.height = 70
    print(rect.height)
```

Rectangle › height() › else

main

```
C:\Users\podar\study\anaconda\envs\14LR\python.exe "C:/Users/podar
10
20
50
70
```

Рисунок 5 – Уровни доступа атрибута и метода

```
    def __reduce(self):
        # Функция для нахождения наибольшего общего делителя
        def gcd(a, b):
            if a == 0:
                return b
            elif b == 0:
                return a
            elif a >= b:
                return gcd(a % b, b)
            else:
                return gcd(a, b % a)

        c = gcd(self.__numerator, self.__denominator)
        self.__numerator //= c
        self.__denominator //= c
```

main

```
C:\Users\podar\study\anaconda\envs\14LR\python.exe "C:/Users/podar/study
3/4
Введите обыкновенную дробь: 4/5
4/5
31/20
1/20
3/5
16/15
```

Рисунок 6 – Свойства

Индивидуальное задание №1

Парой называется класс с двумя полями, которые обычно имеют имена first и second. Требуется реализовать тип данных с помощью такого класса.

Во всех заданиях обязательно должны присутствовать:

- метод инициализации__init__; метод должен контролировать значения аргументов на корректность;
- ввод с клавиатуры read ;
- вывод на экран display .

Реализовать внешнюю функцию с именем make_тип() , где тип — тип реализуемой структуры. Функция должна получать в качестве аргументов значения для полей структуры и возвращать структуру требуемого типа. При передаче ошибочных параметров следует выводить сообщение и заканчивать работу.

Поле first — дробное число; поле second — целое число, показатель степени. Реализовать метод power() — возведение числа first в степень second. Метод должен правильно работать при любых допустимых значениях first и second.

```
class Num:
    def __init__(self, first=0, second=0):
        self.first = first
        self.second = second

    def read(self):
        self.first = float(input("Введите дробное число: "))
        self.second = int(input("Введите целое число: "))

    def display(self):
        print(f"Результат: ", power(self.first, self.second))

def power(first, second):
    if first == 0:
        raise ValueError
    else:
        return first ** second
```

```
Num
main
Введите дробное число: 3.2
Введите целое число: 2
Результат:  10.240000000000002
```

Рисунок 7 – Индивидуальное задание №1

Индивидуальное задание №2

Составить программу с использованием классов и объектов для решения задачи. Во всех заданиях, помимо указанных в задании операций, обязательно должны быть реализованы следующие методы:

- метод инициализации__init__;
- ввод с клавиатуры read ;
- вывод на экран display .

Создать класс Vector3D, задаваемый тройкой координат. Обязательно должны быть реализованы: сложение и вычитание векторов, скалярное произведение векторов, умножение на скаляр, сравнение векторов, вычисление длины вектора, сравнение длины векторов.

```
class Vector3D:

    def __init__(self, x=0, y=0, z=0):
        self.x = x
        self.y = y
        self.z = z


    def read(self, prompt=None):
        line = input() if prompt is None else input(prompt)
        parts = list(map(int, line.split(' ', maxsplit=2)))
        if parts[2] == 0:
            raise ValueError()

        self.x = parts[0]
        self.y = parts[1]
        self.z = parts[2]
```

```
if __name__ == "__main__"
main
Координаты:  7, 5, 2
Введите координаты: 5 5 5
Координаты:  5, 5, 5
Координаты:  12, 10, 7
Координаты:  2, 0, -3
Скалярное произведение векторов: 70
```

Рисунок 8 – Индивидуальное задание №2

ВОПРОСЫ

1. Как осуществляется объявление класса в языке Python?

Классы объявляются с помощью ключевого слова class и имени класса:

class MyClass:

var = ... # некоторая переменная

def do_smt(self):

какой-то метод

2. Чем атрибуты класса отличаются от атрибутов экземпляра?

Атрибут класса - это атрибут, общий для всех экземпляров класса. Атрибуты класса определены внутри класса, но вне каких-либо методов. Их значения одинаковы для всех экземпляров этого класса. Так что вы можете рассматривать их как тип значений по умолчанию для всех наших объектов.

Атрибуты экземпляра определяются в методах и хранят информацию, специфичную для экземпляра.

3. Каково назначение методов класса?

Методы определяют функциональность объектов, принадлежащих конкретному классу.

4. Для чего предназначен метод__init__() класса?

Метод__init__является конструктором. Конструкторы - это концепция объектно- ориентированного программирования. Класс может иметь один и только один конструктор. Если__init__определен внутри класса, он автоматически вызывается при создании нового экземпляра класса. Метод __init__указывает, какие атрибуты будут у экземпляров нашего класса.

5. Каково назначение self ?

Аргумент self представляет конкретный экземпляр класса и позволяет нам получить доступ к его атрибутам и методам. В примере с__init__мы создаем атрибуты для конкретного экземпляра и присваиваем им значения аргументов метода. Важно использовать параметр self внутри метода, если мы хотим сохранить значения экземпляра для последующего использования.

В большинстве случаев нам также необходимо использовать параметр self в других методах, потому что при вызове метода первым аргументом, который ему передается, является сам объект.

6. Как добавить атрибуты в класс?

Новый атрибут класса указывается через точку после названия класса, затем ему присваивается определенное значение.

7. Как осуществляется управление доступом к методам и атрибутам в языке Python?

Хорошим тоном считается, что для чтения/изменения какого-то атрибута должны использоваться специальные методы, которые называются getter/setter, их можно реализовать, но ничего не помешает изменить атрибут напрямую. При этом есть соглашение, что метод или атрибут, который начинается с нижнего подчеркивания, является скрытым, и снаружи класса трогать его не нужно (хотя сделать это можно).

Если же атрибут или метод начинается с двух подчеркиваний, то тут напрямую вы к нему уже не обратитесь (простым образом).

8. Каково назначение функции isinstance ?

Встроенная функция isinstance(obj, Cls) , используемая при реализации методов арифметических операций и операций отношения, позволяет узнать что некоторый объект obj является либо экземпляром класса Cls либо экземпляром одного из потомков класса Cls.