

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ ФЕДЕРАЦИИ  
Федеральное государственное автономное  
образовательное учреждение высшего образования  
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Кафедра инфокоммуникаций

**Отчет по лабораторной работе №1**  
**Элементы объектно-ориентированного**  
**программирования в языке Python**  
**По дисциплине «Технологии программирования и алгоритмизация»**

Выполнил студент группы ИВТ-б-о-20-1

Галяс Д. И. « » \_\_\_\_\_ 20\_\_ г.

Подпись студента \_\_\_\_\_

Работа защищена « » \_\_\_\_\_ 20\_\_ г.

Проверил Воронкин Р. А. \_\_\_\_\_

(подпись)

Ставрополь 2022

**Цель работы:** приобретение навыков по работе с классами и объектами при написании программ с помощью языка программирования Python версии 3.x.

**Ссылка на репозиторий:** <https://github.com/DIMITRY-GALYAS1/Laba-4.1.git>

**Ход работы:**

1. Создал новый репозиторий на github, после клонировал его и создал в папке репозитория новый проект PyCharm.
2. Выполнил первый пример.

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

class Rational:

    def __init__(self, a=0, b=1):
        a = int(a)
        b = int(b)

        if b == 0:
            raise ValueError()

        self.__numerator = abs(a)
        self.__denominator = abs(b)

        self.__reduce()

    # Сокращение дроби
    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)

        g = gcd(self.__numerator, self.__denominator)
```

Рисунок 1. Пример 1

3. Выполнил первое индивидуальное задание.

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

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

class MyClass:

    def __init__(self, first=0, second=0):
        self.first = float(first)
        self.second = float(second)

    def read(self):
        first = input('Введите дробное число: ')
        second = input('Введите целое число, показатель степени: ')

        self.first = float(first)
        self.second = float(second)

    def display(self):
        print("Вы хотите получить число {0} в степени {1}".format(self.first, self.second))

    def make_power(self):
        if self.first == 0:
            raise ValueError()
        else:
```

Рисунок 2. Первое индивидуальное задание

#### 4. Выполнил второе индивидуальное задание.

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

"""
Создать класс ModelWindow для работы с моделями экранных окон. В качестве полей
задаются: заголовок окна, координаты левого верхнего угла, размер по горизонтали, размер
по вертикали, цвет окна, состояние «видимое/невидимое», состояние «с рамкой/без рамки».
Координаты и размеры указываются в целых числах. Реализовать операции: передвижение
окна по горизонтали, по вертикали; изменение высоты и/или ширины окна изменение цвета;
изменение состояния, опрос состояния. Операции передвижения и изменения размера
должны осуществлять проверку на пересечение границ экрана. Функция вывода на экран
должна индуцировать состояние полей объекта.
"""

class ModelWindow:

    def __init__(self, zagolovok, x, y, gorizontal, vertical, color, condition, frame_status):
        self.zagolovok = str(zagolovok)
        self.x = int(x)
        self.y = int(y)
        self.gorizontal = int(gorizontal)
        self.vertical = int(vertical)
        self.color = str(color).lower()
        self.condition = str(condition).lower()
        self.frame_status = str(frame_status).lower()

    def read(self):
        zagolovok = input("Введите заголовок: ")
        self.zagolovok = str(zagolovok)
```

Рисунок 3. Второе индивидуальное задание

### Контрольные вопросы:

1. Как осуществляется объявление класса в языке Python?

Классы объявляются с помощью ключевого слова `class` и имени класса.

2. Чем атрибуты класса отличаются от атрибутов экземпляра?

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. Что касается переменных экземпляра, они хранят данные, уникальные для каждого объекта класса.

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.

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.

6. Как добавить атрибуты в класс?

Атрибут класса - это атрибут, общий для всех экземпляров класса.

Атрибуты экземпляра - это как раз те, которые мы определяем в методах, поэтому по определению мы можем создавать новые атрибуты внутри наших пользовательских методов.

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

В Python таких возможностей нет, и любой может обратиться к атрибутам и методам вашего класса, если возникнет такая необходимость. Это существенный недостаток этого языка, т.к. нарушается один из ключевых принципов ООП – инкапсуляция. Хорошим тоном считается, что для чтения/изменения какого-то атрибута должны использоваться специальные методы, которые называются getter/setter, их можно реализовать, но ничего не мешает 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).

#### 8. Каково назначение функции `isinstance`?

В Python есть встроенная функция `instance ()`, которая сравнивает значение с указанным типом. Если данное значение и тип соответствуют, он вернет `true`, иначе `false`. Используя `isinstance ()`, вы можете проверить строку, число с плавающей точкой, `int`, список, кортеж, `dict`, `set`, `class` и т.д.

**Вывод:** в ходе выполнения лабораторной работы приобрел навыки по работе с классами и объектами при написании программ с помощью языка программирования Python версии 3.x.