Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт перспективной инженерии
Департамента цифровых, роботехнических систем и электроники

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ №1
дисциплины «Объектно-ориентированное программирование»

Выполнил:
Горбунов Данила Евгеньевич
3 курс, группа ИВТ-б-о-22-1,
09.03.01 «Информатика
и вычислительная техника»,
направленность (профиль)
«Программное обеспечение средств
вычислительной техники и
автоматизированных систем», очная
форма обучения

_____
(подпись)

Руководитель практики:
Воронкин Р.А.-доцент департамента
цифровых, роботехнических систем и
электроники института перспективной
инженерии

_____
(подпись)

Отчет защищен с оценкой _____ Дата защиты_____

Ставрополь, 2024 г.

Тема: «Элементы объектно-ориентированного программирования в языке Python»

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

Порядок выполнения работы

1. Необходимо создать репозиторий на Web-сервисе GitHub. (Рисунок 1)

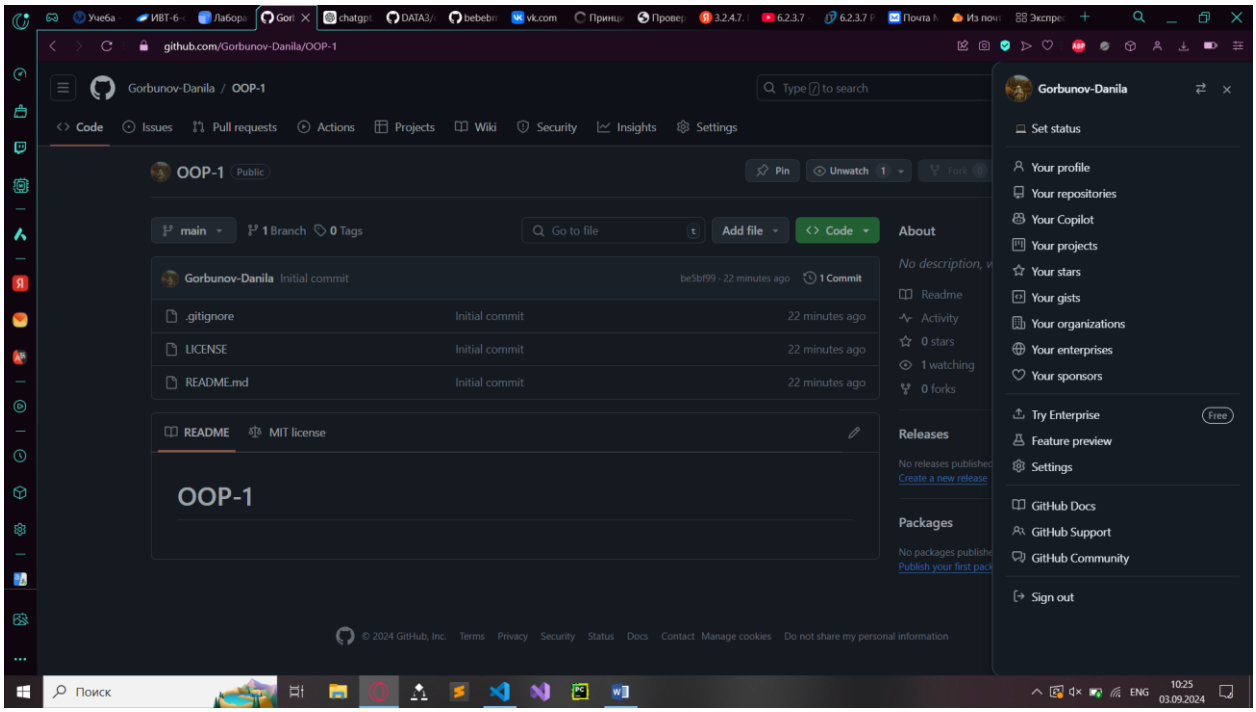

Рисунок 1. Создание репозитория

2. Необходимо выполнить пример 1. (Рисунок 2)

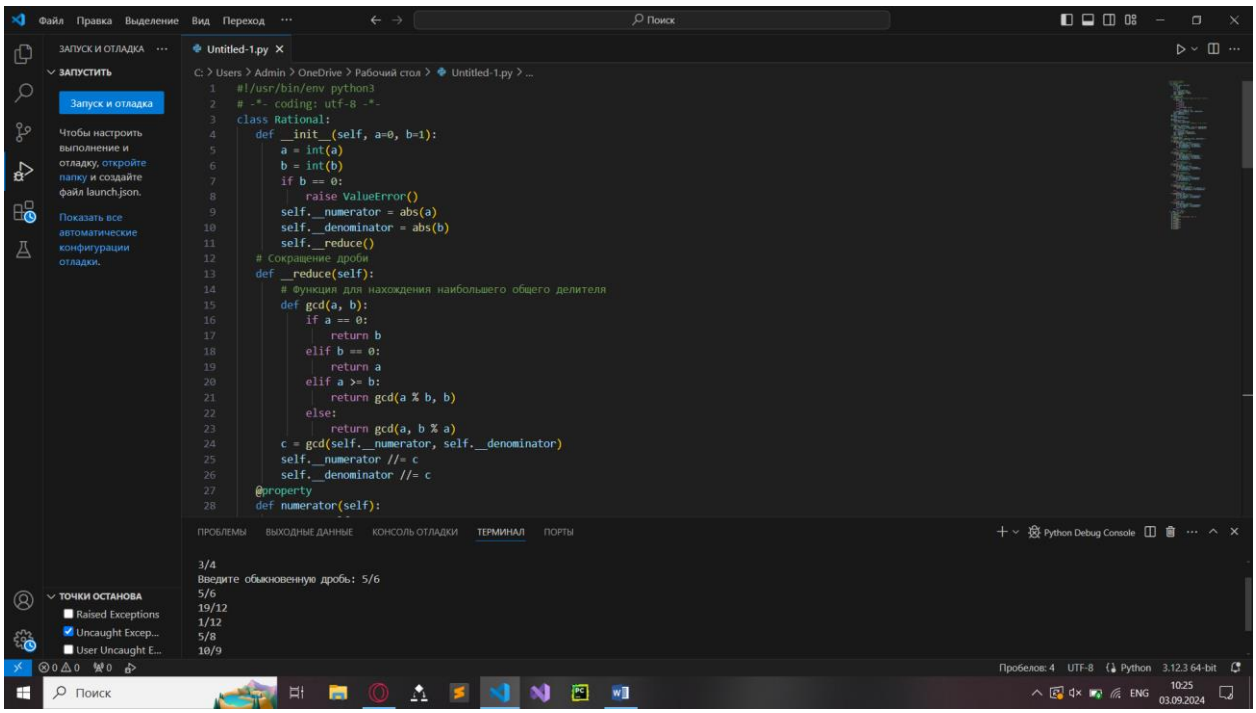

Рисунок 2. Выполнение примера 1

3. Необходимо выполнить индивидуального задание 1. (Рисунок 3)

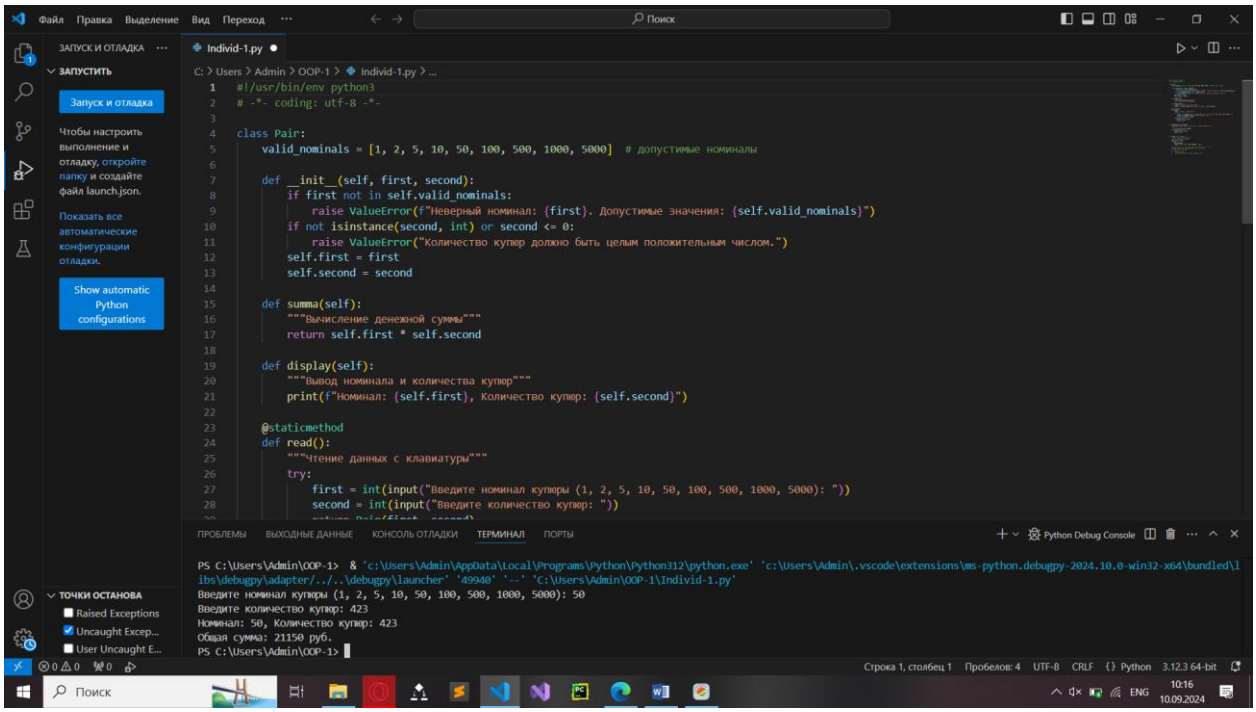

Рисунок 3. Выполнение индивидуального задания 1

4. Необходимо выполнить индивидуальное задание 2. (Рисунок 4)

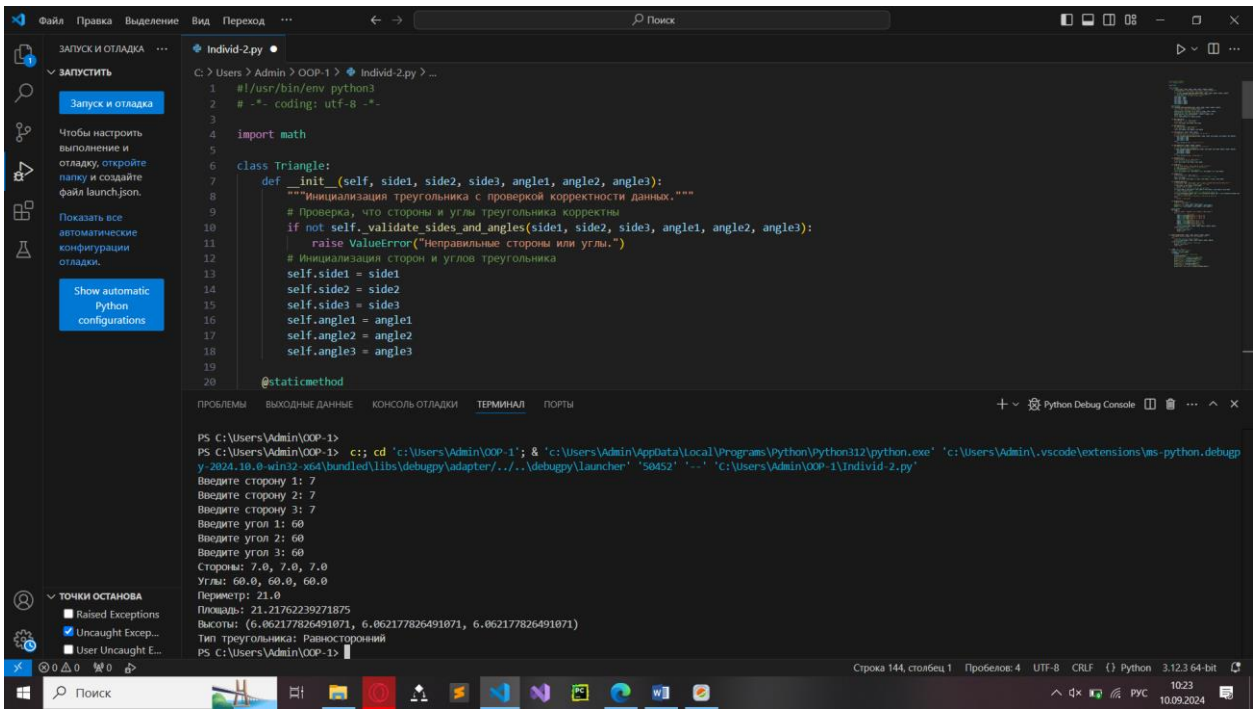

Рисунок 4. Выполнение индивидуального задания 2

Контрольные вопросы

1. Как осуществляется объявление класса в языке Python?

В Python класс объявляется с помощью ключевого слова `class`. Пример:

```
class MyClass:
    def __init__(self, name):
        self.name = name
    def say_hello(self):
        print(f"Hello, {self.name}!")
```

2. Чем атрибуты класса отличаются от атрибутов экземпляра?

Атрибуты класса — это переменные, общие для всех экземпляров (объектов) класса. Они объявляются внутри класса, но вне методов.

Атрибуты экземпляра — это переменные, которые принадлежат конкретному экземпляру класса и могут иметь разные значения для каждого экземпляра. Они обычно определяются в методе __init__ с использованием self.

```
class MyClass:
    class_attribute = 10  # Атрибут класса
```

```
    def __init__(self, instance_value):
        self.instance_attribute = instance_value  # Атрибут экземпляра
obj1 = MyClass(5)
obj2 = MyClass(15)
print(obj1.class_attribute)  # 10 (атрибут класса общий)
print(obj1.instance_attribute)  # 5 (разные значения для разных объектов)
print(obj2.instance_attribute)  # 15
```

3. Каково назначение методов класса?

Методы класса — это функции, которые принадлежат классу и выполняют действия с его атрибутами. Они могут изменять состояние объекта, работать с данными класса и взаимодействовать с другими объектами.

Пример:

```
class MyClass:
    def __init__(self, value):
        self.value = value
    def double(self):
        return self.value * 2
```

4. Для чего предназначен метод __init__() класса?

Метод __init__ — это инициализатор, который вызывается при создании нового объекта класса. Он используется для задания начальных значений атрибутов экземпляра.

Пример:

```
class MyClass:
    def __init__(self, value):
        self.value = value
obj = MyClass(10)
print(obj.value)  # 10
```

5. Каково назначение self ?

self — это ссылка на текущий экземпляр класса, который используется для доступа к атрибутам и методам объекта. Он обязателен как первый параметр методов экземпляра.

Пример:

```
class MyClass:
    def __init__(self, value):
        self.value = value
    def get_value(self):
        return self.value  # Используем self для доступа к атрибуту
```

6. Как добавить атрибуты в класс?

Атрибуты могут быть добавлены через метод __init__ или напрямую в класс. Для экземпляра атрибуты добавляются с помощью self:

```
class MyClass:
    class_attribute = 0  # Атрибут класса
    def __init__(self, value):
        self.instance_attribute = value  # Атрибут экземпляра
```

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

Python не имеет строгих модификаторов доступа (таких как private, public в других языках), но есть соглашения:

Атрибуты и методы с одним подчеркиванием (_attribute) считаются "защищенными", и их не рекомендуется использовать вне класса.

Атрибуты и методы с двумя подчеркиваниями (__attribute) считаются "частными", и Python применяет манглинг имен, что затрудняет доступ к ним из внешнего кода.

Пример:

```
class MyClass:
    def __init__(self):
        self.public = "Это публично"
        self._protected = "Это защищено"
```

```
        self.__private = "Это приватно"
```

8. Каково назначение функции isinstance ?

Функция isinstance() проверяет, принадлежит ли объект к определенному классу (или его подклассу). Возвращает True, если объект принадлежит указанному классу, иначе — False.

Пример:

```
class MyClass:
    pass
obj = MyClass()
print(isinstance(obj, MyClass))  # True
print(isinstance(obj, int))  # False
```