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)

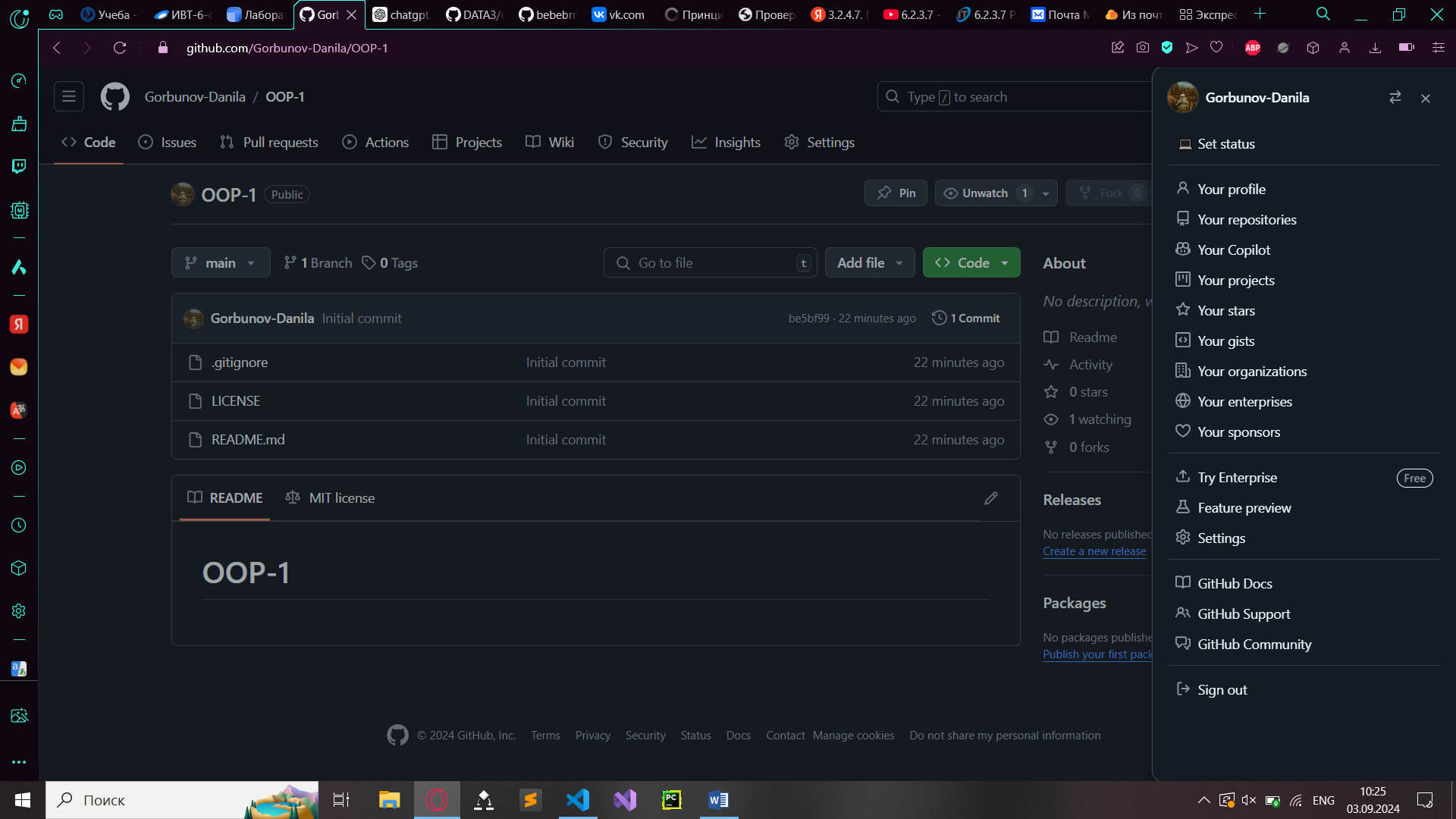


Рисунок 1. Создание репозитория

1. Необходимо выполнить пример 1. (Рисунок 2)

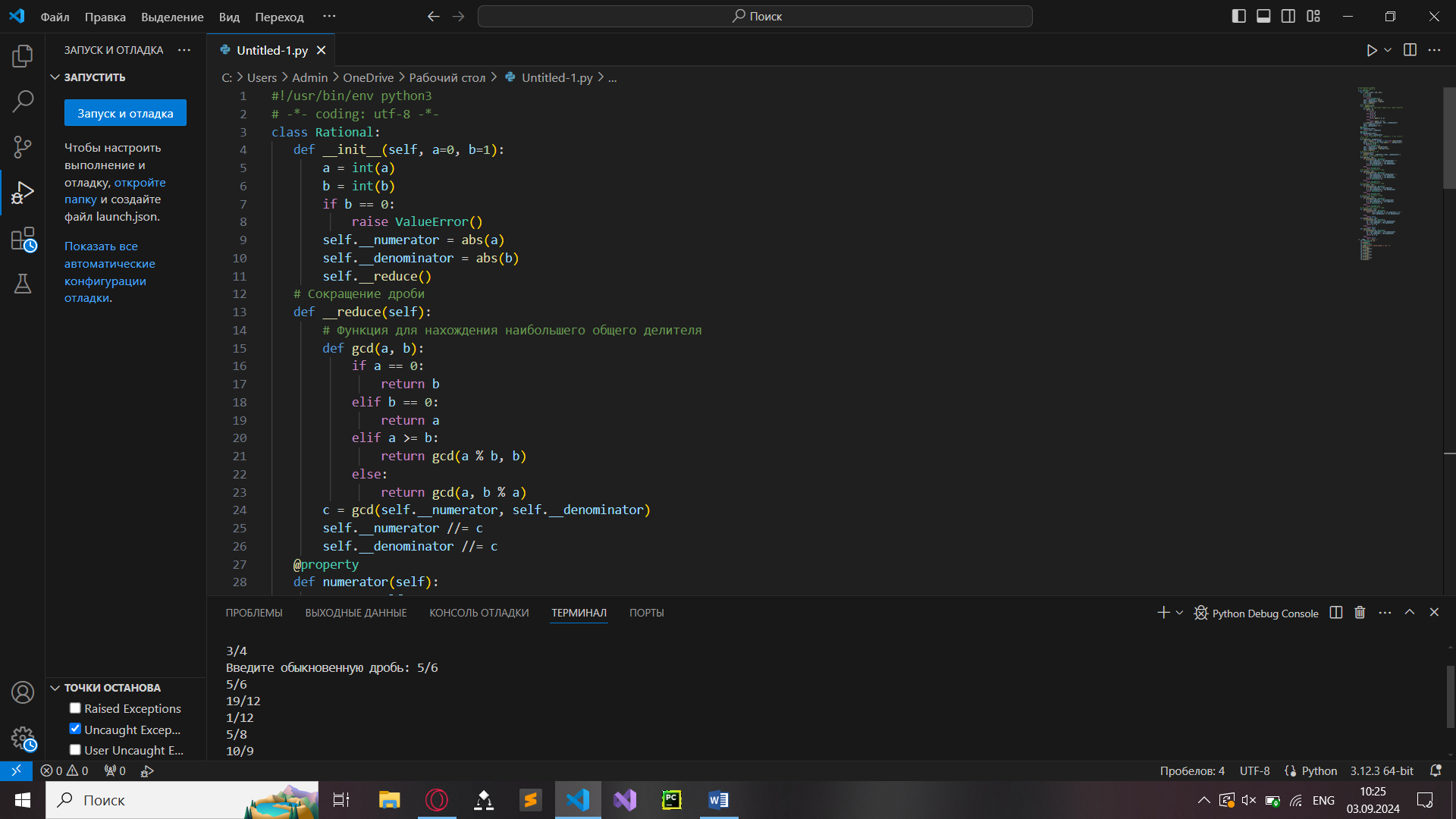


Рисунок 2. Выполнение примера 1

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

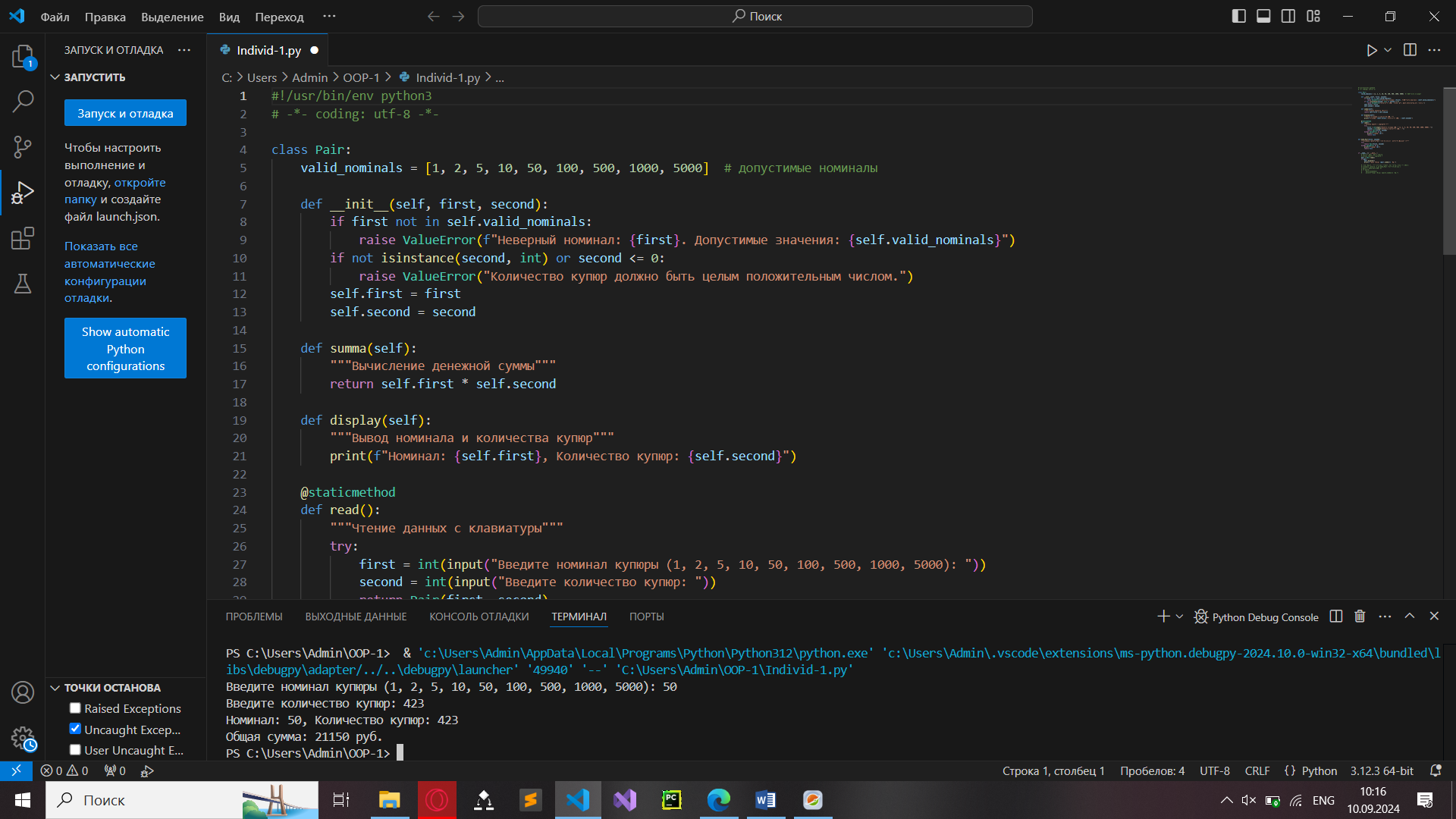


Рисунок 3. Выполнение индивидуального задания 1

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

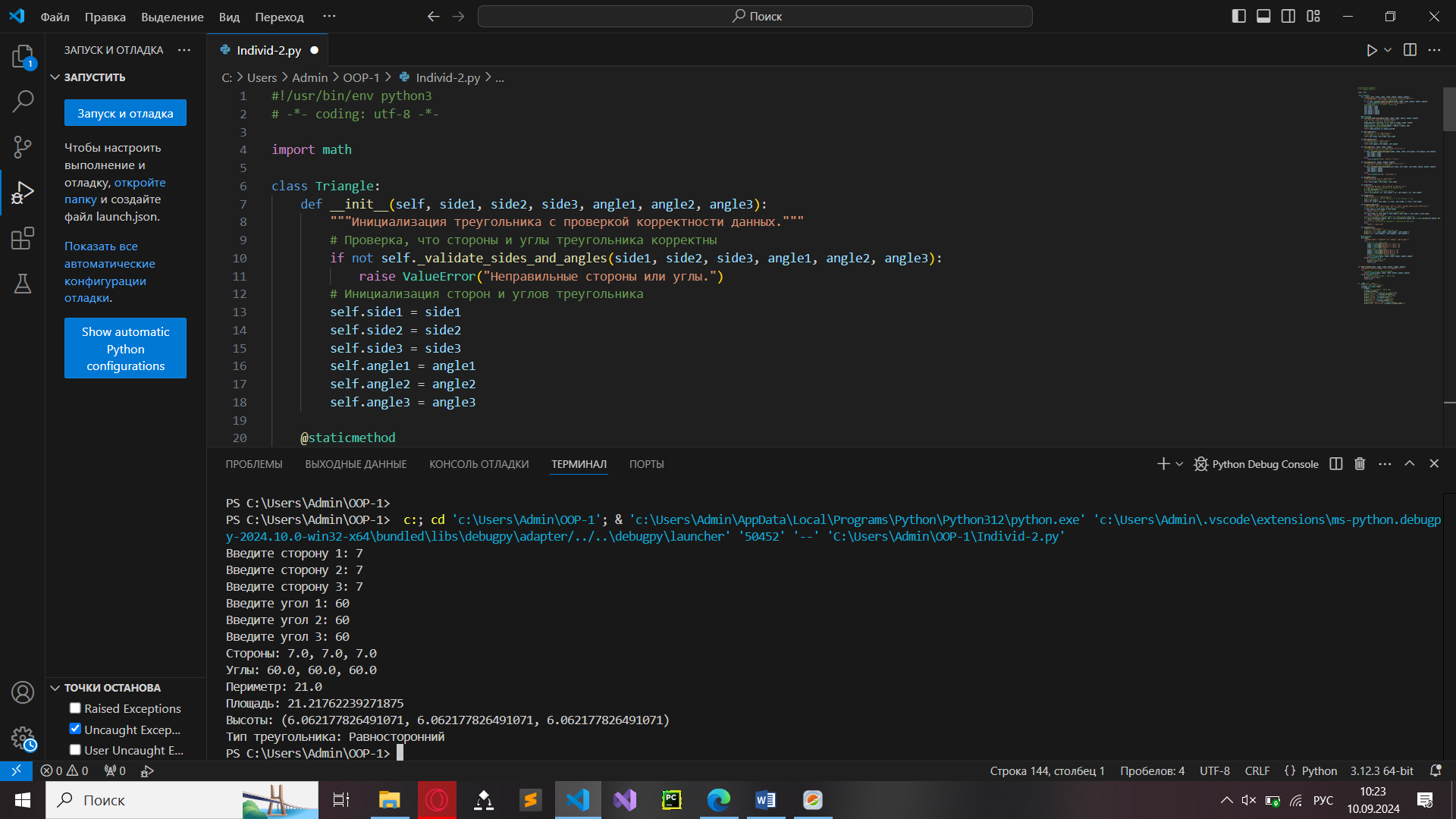


Рисунок 4. Выполнение индивидуального задания 2

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}!")

1. 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

1. 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

1. 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

1. 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

1. 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

1. 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"

1. 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