

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

Институт цифрового развития  
Кафедра инфокоммуникаций

**«Наследование и полиморфизм в языке Python»**

**ОТЧЕТ**  
**по лабораторной работе №4.3**  
**дисциплины**  
**«Основы программной инженерии»**

Выполнила:

Кувшин Ирина Анатольевна  
2 курс, группа ПИЖ-б-о-21-1,  
011.03.04 «Программная инженерия»,  
направленность (профиль) «Разработка  
и сопровождение программного  
обеспечения», очная форма обучения

---

(подпись)

Проверил:

---

(подпись)

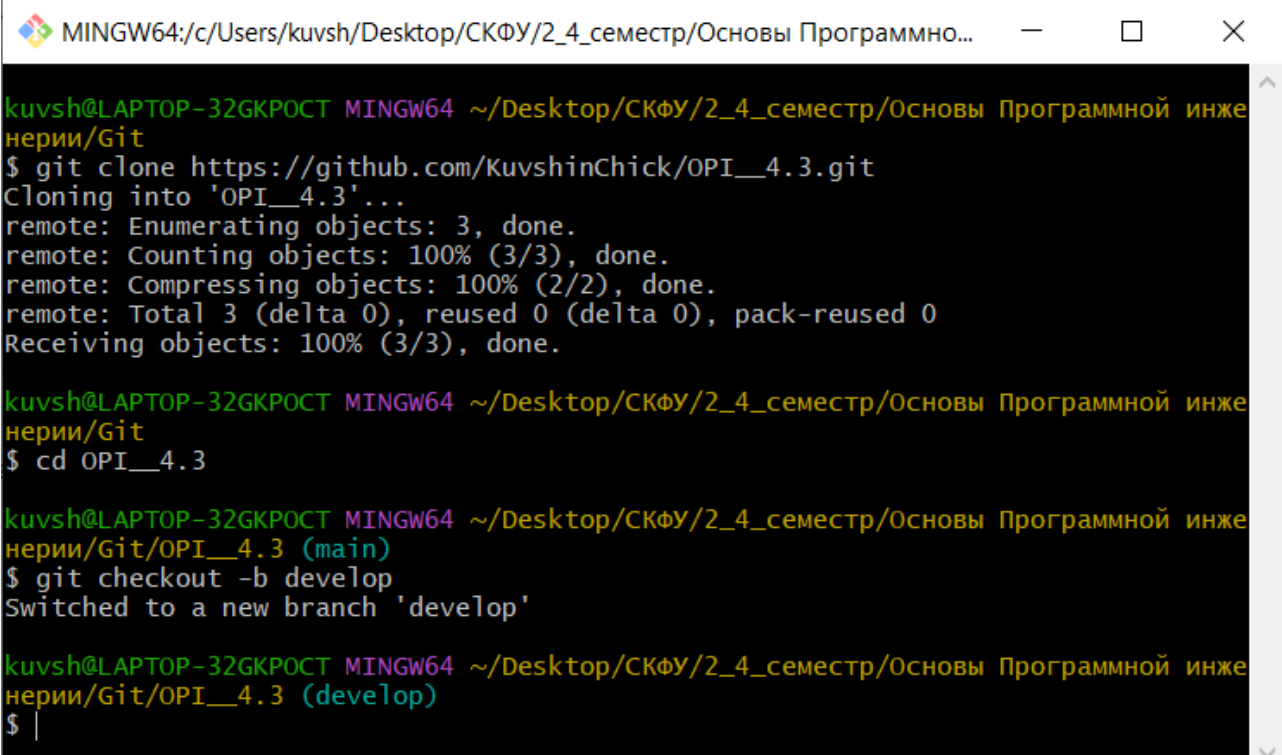
Отчет защищен с оценкой \_\_\_\_\_ Дата защиты \_\_\_\_\_

Ставрополь, 2023 г.

**Цель работы:** приобретение навыков по созданию иерархии классов при написании программ с помощью языка программирования Python версии 3.x.

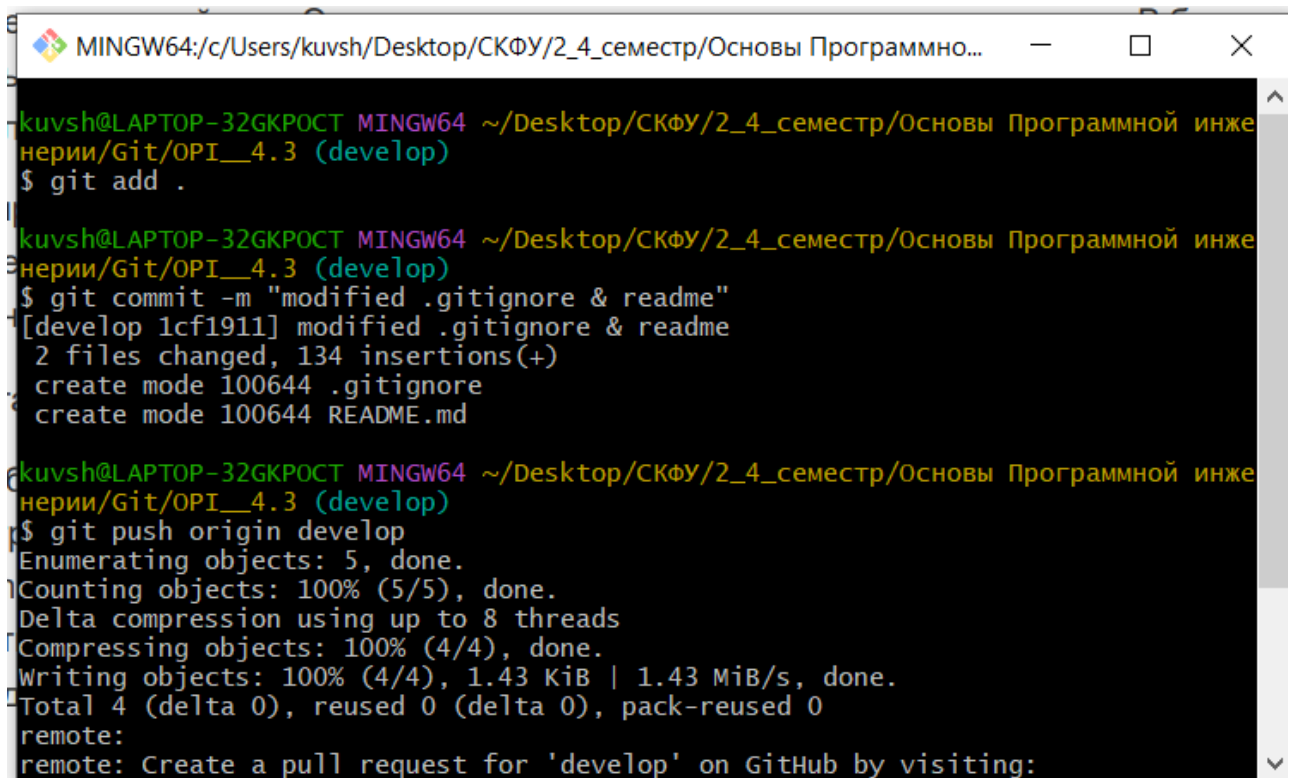
**Ход работы:**

1. Изучить теоретический материал работы.
2. Создать общедоступный репозиторий на GitHub, в котором будет использована лицензия MIT и язык программирования Python.
3. Выполните клонирование созданного репозитория.
4. Дополните файл .gitignore необходимыми правилами для работы с IDE PyCharm.
5. Организуйте свой репозиторий в соответствии с моделью ветвления git-flow.
6. Создайте проект PyCharm в папке репозитория.
7. Проработайте примеры лабораторной работы.



```
MINGW64:/c:/Users/kuvsh/Desktop/СКФУ/2_4_семестр/Основы Программно...
kuvsh@LAPTOP-32GKPOCT MINGW64 ~/Desktop/СКФУ/2_4_семестр/Основы Программно...
$ git clone https://github.com/KuvshinChick/OPI__4.3.git
Cloning into 'OPI__4.3'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (3/3), done.
kuvsh@LAPTOP-32GKPOCT MINGW64 ~/Desktop/СКФУ/2_4_семестр/Основы Программно...
$ cd OPI__4.3
kuvsh@LAPTOP-32GKPOCT MINGW64 ~/Desktop/СКФУ/2_4_семестр/Основы Программно...
$ git checkout -b develop
Switched to a new branch 'develop'
kuvsh@LAPTOP-32GKPOCT MINGW64 ~/Desktop/СКФУ/2_4_семестр/Основы Программно...
$ |
```

Рисунок 4.1.1 – Клонирование репозитория и создание ветки develop



```
MINGW64:/c/Users/kuvsh/Desktop/СКФУ/2_4_семестр/Основы Программно...
kuvsh@LAPTOP-32GKPOCT MINGW64 ~/Desktop/СКФУ/2_4_семестр/Основы Программно...
$ git add .
kuvsh@LAPTOP-32GKPOCT MINGW64 ~/Desktop/СКФУ/2_4_семестр/Основы Программно...
$ git commit -m "modified .gitignore & readme"
[develop 1cf1911] modified .gitignore & readme
2 files changed, 134 insertions(+)
create mode 100644 .gitignore
create mode 100644 README.md
kuvsh@LAPTOP-32GKPOCT MINGW64 ~/Desktop/СКФУ/2_4_семестр/Основы Программно...
$ git push origin develop
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 8 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 1.43 KiB | 1.43 MiB/s, done.
Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'develop' on GitHub by visiting:
```

Рисунок 4.1.2 – Обновление .gitignore и readme

8. Решите задачу:

9. Разработайте программу по следующему описанию.

В некой игре-стратегии есть солдаты и герои. У всех есть свойство, содержащее уникальный номер объекта, и свойство, в котором хранится принадлежность команде. У солдат есть метод "иду за героем", который в качестве аргумента принимает объект типа "герой". У героев есть метод увеличения собственного уровня. В основной ветке программы создается по одному герою для каждой команды. В цикле генерируются объекты-солдаты. Их принадлежность команде определяется случайно. Солдаты разных команд добавляются в разные списки. Измеряется длина списков солдат противоборствующих команд и выводится на экран. У героя, принадлежащего команде с более длинным списком, увеличивается уровень. Отправьте одного из солдат первого героя следовать за ним. Выведите на экран идентификационные номера этих двух юнитов.

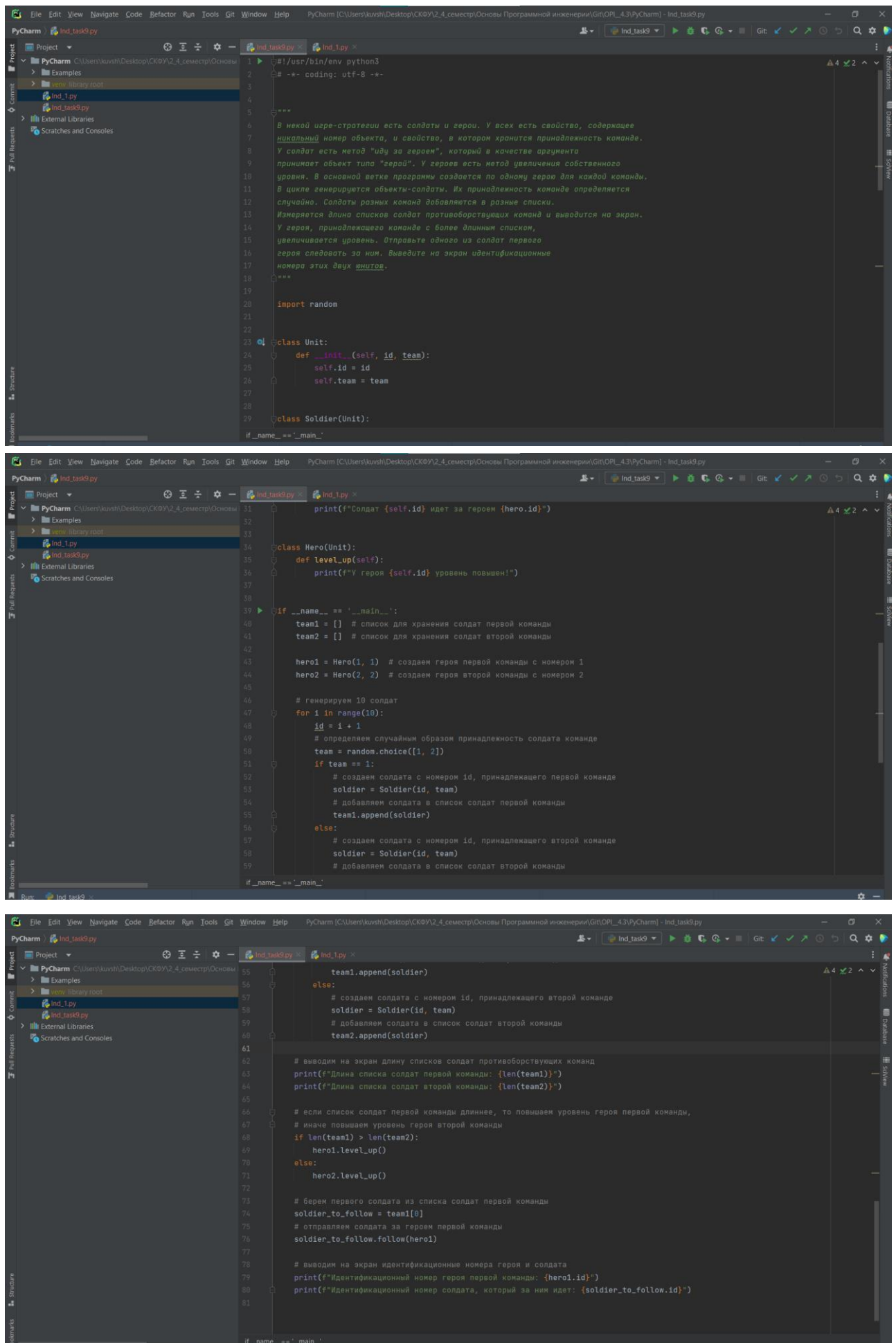


Рисунок 4.3.3 – Проработка программы

```
Ind_task9 x
"C:\Users\kuvsh\Desktop\СКФУ\2_4_семестр\Основы Программной инженерии\Git\OPI__4.3\
Длина списка солдат первой команды: 5
Длина списка солдат второй команды: 5
У героя 2 уровень повышен!
Солдат 2 идет за героем 1
Идентификационный номер героя первой команды: 1
Идентификационный номер солдата, который за ним идет: 2

Process finished with exit code 0
```

```
Ind_task9 x
"C:\Users\kuvsh\Desktop\СКФУ\2_4_семестр\Основы Программной инженер
Длина списка солдат первой команды: 2
Длина списка солдат второй команды: 8
У героя 2 уровень повышен!
Солдат 5 идет за героем 1
Идентификационный номер героя первой команды: 1
Идентификационный номер солдата, который за ним идет: 5

Process finished with exit code 0
```

Рисунок 4.3.4 – Результат работы программы

10. Выполните индивидуальные задания. Приведите в отчете скриншоты работы программ решения индивидуального задания.

### Задание 1

Составить программу с использованием иерархии классов. Номер варианта необходимо получить у преподавателя. В раздел программы, начинающийся после инструкции `if __name__ == '__main__':` добавить код, демонстрирующий возможности разработанных классов.

8. Создать класс `Triangle` с полями-сторонами. Определить методы изменения сторон, вычисления углов, вычисления периметра. Создать производный класс `RightAngled` (прямоугольный), имеющий поле площади. Определить метод вычисления площади.

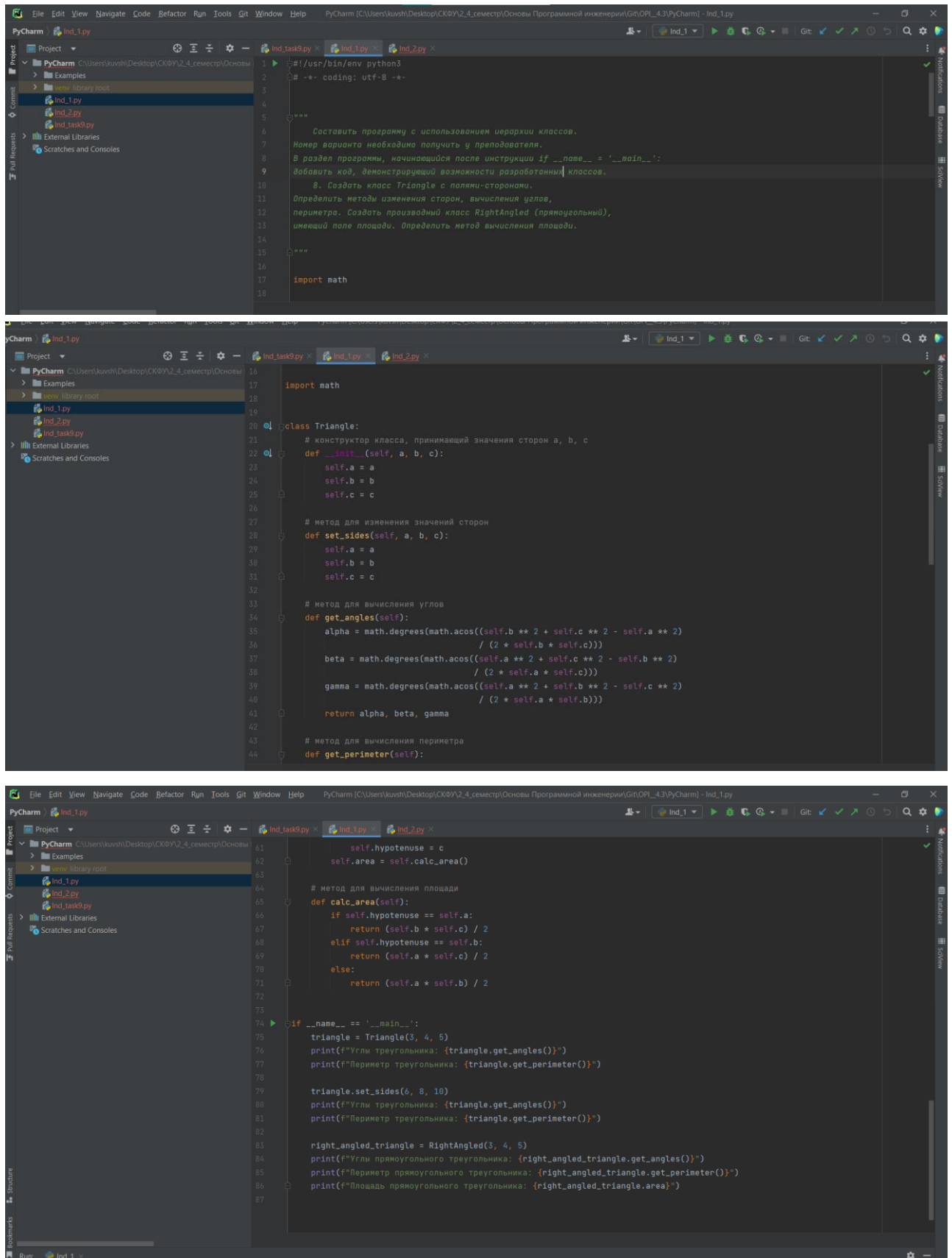
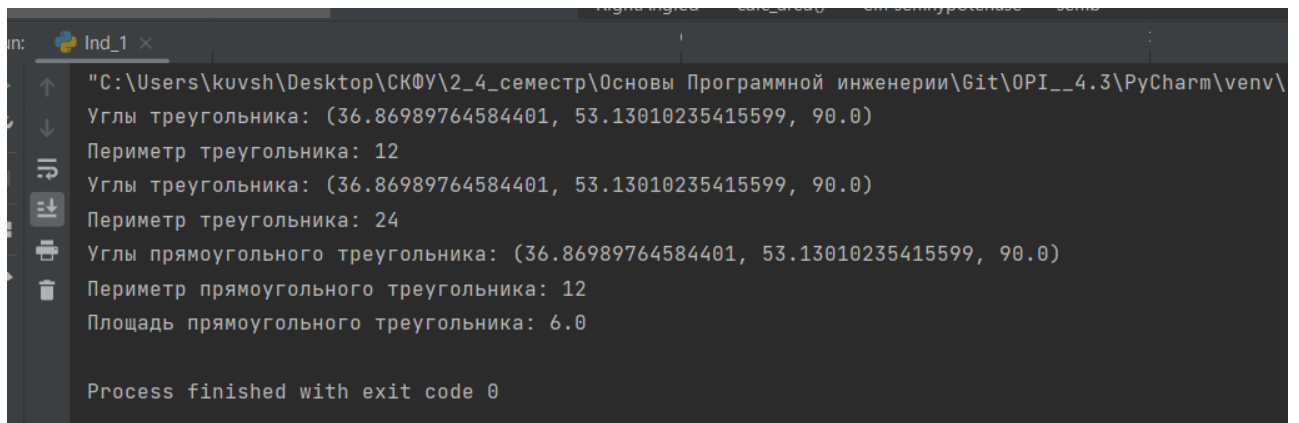


Рисунок 4.3.5 – Проработка программы



```
"C:\Users\kuvsh\Desktop\СКФУ\2_4_семестр\Основы Программной инженерии\Git\OPI__4.3\PyCharm\venv\
Углы треугольника: (36.86989764584401, 53.13010235415599, 90.0)
Периметр треугольника: 12
Углы треугольника: (36.86989764584401, 53.13010235415599, 90.0)
Периметр треугольника: 24
Углы прямоугольного треугольника: (36.86989764584401, 53.13010235415599, 90.0)
Периметр прямоугольного треугольника: 12
Площадь прямоугольного треугольника: 6.0

Process finished with exit code 0
```

Рисунок 4.3.6 – Результат работы программы

## Задание 2

В следующих заданиях требуется реализовать абстрактный базовый класс, определив в нем абстрактные методы и свойства. Эти методы определяются в производных классах. В базовых классах должны быть объявлены абстрактные методы ввода/вывода, которые реализуются в производных классах. Вызывающая программа должна продемонстрировать все варианты вызова переопределенных абстрактных методов. Написать функцию вывода, получающую параметры базового класса по ссылке и демонстрирующую виртуальный вызов. Номер варианта необходимо получить у преподавателя.

8. Создать абстрактный базовый класс Function (функция) с виртуальными методами вычисления значения функции  $y = f(x)$  в заданной точке  $x$  и вывода результата на экран. Определить производные классы Ellipse (эллипс), Hyperbola (гипербола) с собственными функциями вычисления  $y$  в зависимости от входного параметра  $x$ . Уравнение эллипса  $x^2/a^2 + y^2/b^2 = 1$ ; гиперболы:  $x^2/a^2 - y^2/b^2 = 1$ .



```
PyCharm [C:\Users\kuvsh\Desktop\ОКФ\12_4_семестр\Основы Программной инженерии\Git\OPL_43\PyCharm] - Ind_2.py

# Ind_2.py
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4
5
6  """
7  В следующих заданиях требуется реализовать абстрактный базовый класс,
8  определив в нем абстрактные методы и свойства. Эти методы определяются
9  в производных классах. В базовых классах должны быть объявлены абстрактные
10 методы ввода/вывода, которые реализуются в производных классах.
11
12 Вызывающая программа должна продемонстрировать все варианты вызова переопределенных
13 абстрактных методов. Написать функции вывода, получающие параметры базового
14 класса по ссылке и демонстрирующие виртуальный вызов. Номер варианта необходимо получить у преподавателя.
15
16 8. Создать абстрактный базовый класс Function (функция) с виртуальными методами
17 вычисления значения функции в заданной точке и вывода результата на экран.
18 Определить производные классы Ellipse (эллипс), Hyperbola (гипербола) с собственными
19 функциями вычисления y в зависимости от входного параметра x. Уравнение эллипса
20  $x^2/a^2 + y^2/b^2 = 1$ ; гиперболы:  $x^2/a^2 - y^2/b^2 = 1$ .
21
22 """
23
24 from abc import ABC, abstractmethod
25 from math import sqrt
26
27 # Определим абстрактный базовый класс Function с методами calculate() и printResult()
```

Run: Ind\_2.py

Результат вычисления функции (гипербола): 0.0

```
PyCharm [C:\Users\kuvsh\Desktop\ОКФ\12_4_семестр\Основы Программной инженерии\Git\OPL_43\PyCharm] - Ind_2.py

# Ind_2.py
25 # Определим абстрактный базовый класс Function с методами calculate() и printResult()
26 class Function(ABC):
27     @abstractmethod
28     def calculate(self, x: float) -> float:
29         pass
30
31     @abstractmethod
32     def print_result(self):
33         pass
34
35 # Определим класс Ellipse, который будет реализовывать вычисление эллипса
36 class Ellipse(Function):
37     def __init__(self, a: float, b: float):
38         self.a = a
39         self.b = b
40         self.result = 0.0
41
42     def calculate(self, x: float) -> float:
43         # Формула для вычисления эллипса
44         self.result = sqrt(self.b * self.b * (1 - (x * x) / (self.a * self.a)))
45         return self.result
46
47     def print_result(self):
48         # Вывод результата вычисления функции
```

Run: Ind\_2.py



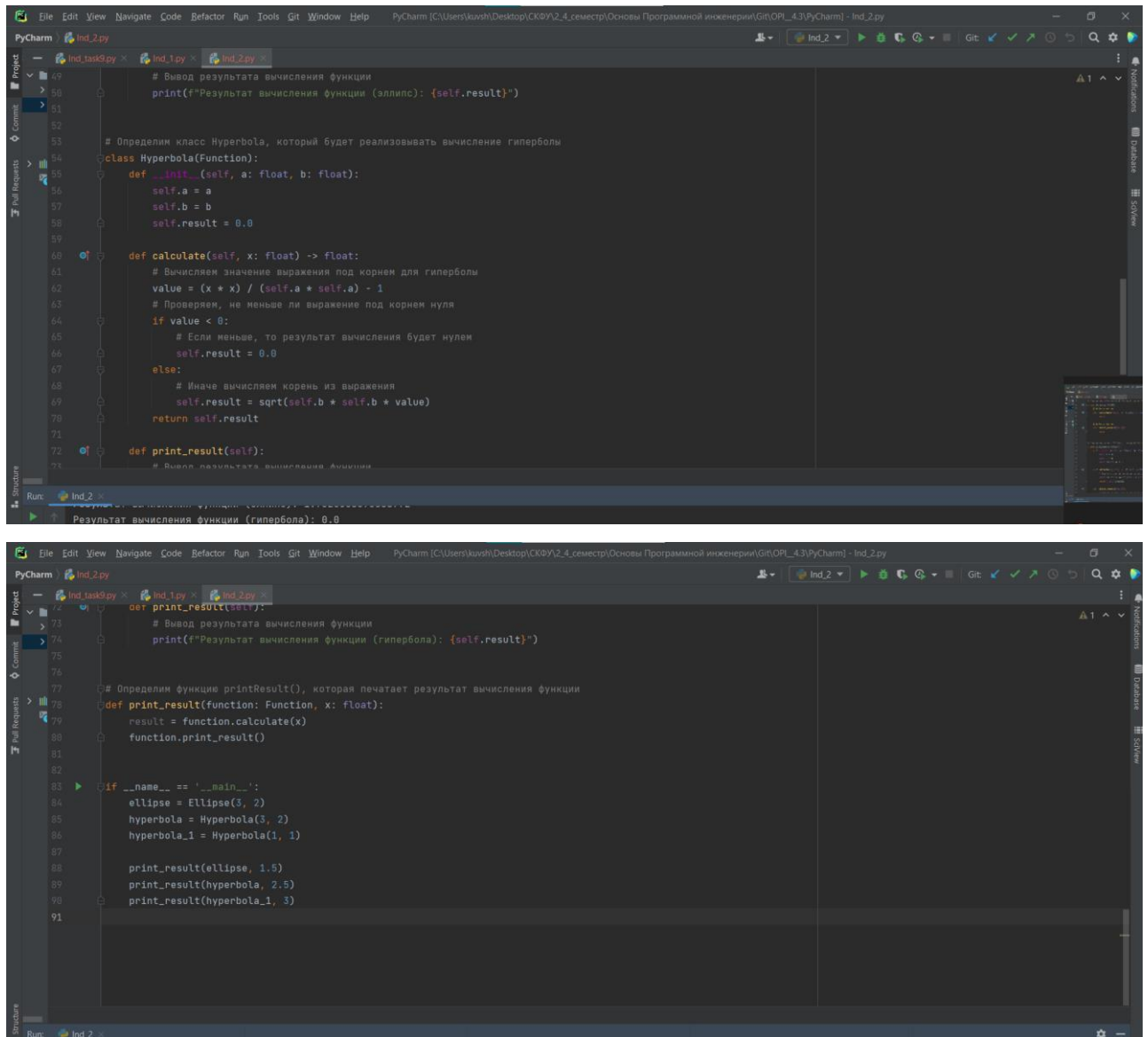


Рисунок 4.3.7 – Проработка программы

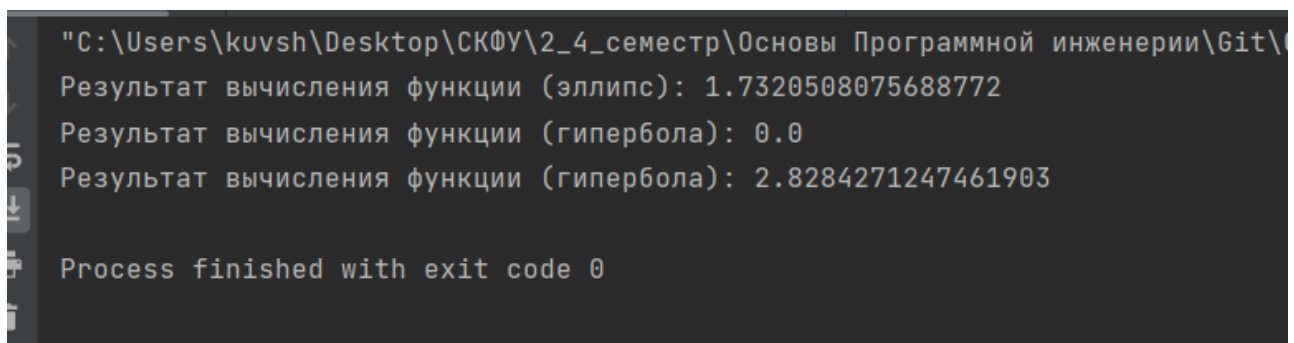


Рисунок 4.3.8 – Результат работы программы

11. Зафиксируйте сделанные изменения в репозитории.
12. Выполните слияние ветки для разработки с веткой main (master).
13. Отправьте сделанные изменения на сервер GitHub.

## Контрольные вопросы

### 1. Что такое наследование как оно реализовано в языке Python?

В организации наследования участвуют как минимум два класса: класс родитель и класс потомок. При этом возможно множественное наследование, в этом случае у класса потомка может быть несколько родителей. Не все языки программирования поддерживают множественное наследование, но в Python можно его использовать. По умолчанию все классы в Python являются наследниками от `object`, явно этот факт указывать не нужно. Синтаксически создание класса с указанием его родителя выглядит так:

```
class имя_класса(имя_родителя1, [имя_родителя2,..., имя_родителя_n])
```

### 2. Что такое полиморфизм и как он реализован в языке Python?

Полиморфизм - это возможность объектов с одинаковой сигнатурой методов вызывать разные реализации этого метода в зависимости от текущего типа объекта. В Python полиморфизм реализуется через вызов методов класса объекта без необходимости указывать явно тип объекта.

### 3. Что такое "утиная" типизация в языке программирования Python?

"Утиная" типизация - это стиль программирования, при котором проверка на соответствие типу объекта происходит во время выполнения, а не на этапе компиляции. В Python все объекты имеют общий тип `object`, и проверка соответствия типу может быть выполнена с помощью ключевого слова `isinstance`.

### 4. Каково назначение модуля `abc` языка программирования Python?

По умолчанию Python не предоставляет абстрактных классов. Python поставляется с модулем, который обеспечивает основу для определения абстрактных базовых классов (ABC), и имя этого модуля - `ABC`. `ABC` работает, декорируя методы базового класса как абстрактные, а затем регистрируя конкретные классы как реализации абстрактной базы. Метод становится абстрактным, если он украшен ключевым словом `@abstractmethod`.

### 5. Как сделать некоторый метод класса абстрактным?

Для того чтобы сделать метод класса абстрактным, нужно создать абстрактный метод в базовом классе с помощью декоратора `@abstractmethod`. Этот метод не должен иметь реализации в базовом классе, и должен быть переопределен в каждом наследнике.

### 6. Как сделать некоторое свойство класса абстрактным?

Для того чтобы сделать свойство класса абстрактным, нужно создать абстрактное свойство в базовом классе с помощью декоратора `@abstractmethod`. Это свойство не должно иметь реализации в базовом классе, и должно быть переопределено в каждом наследнике.

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

Функция `isinstance` используется для проверки соответствия типа объекта указанному классу или его наследнику. Она принимает два аргумента: объект, тип которого нужно проверить, и класс или кортеж классов, с которым нужно сравнить тип объекта. Если объект является экземпляром указанного класса или его наследника, то функция возвращает `True`, в противном случае - `False`.