

**РОССИЙСКОЙ ФЕДЕРАЦИИ**  
**Федеральное государственное автономное**  
**образовательное учреждение высшего образования**  
**«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»**

**Кафедра инфокоммуникаций**

**«Аннотация типов»**

**Отчет по лабораторной работе № 4.5**  
**по дисциплине «Объектно-ориентированное программирование»**

Выполнил студент группы ИВТ-б-о-21-1

Толубаев Рамиль Ахметович

Подпись студента\_\_\_\_\_

Работа защищена « »\_\_\_\_\_20\_\_г.

Проверил Воронкин Р.А. \_\_\_\_\_  
(подпись)

Ставрополь 2023

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

**Порядок выполнения работы:**

1. Создал общедоступный репозиторий на GitHub, в котором использована лицензия MIT и язык программирования Python.

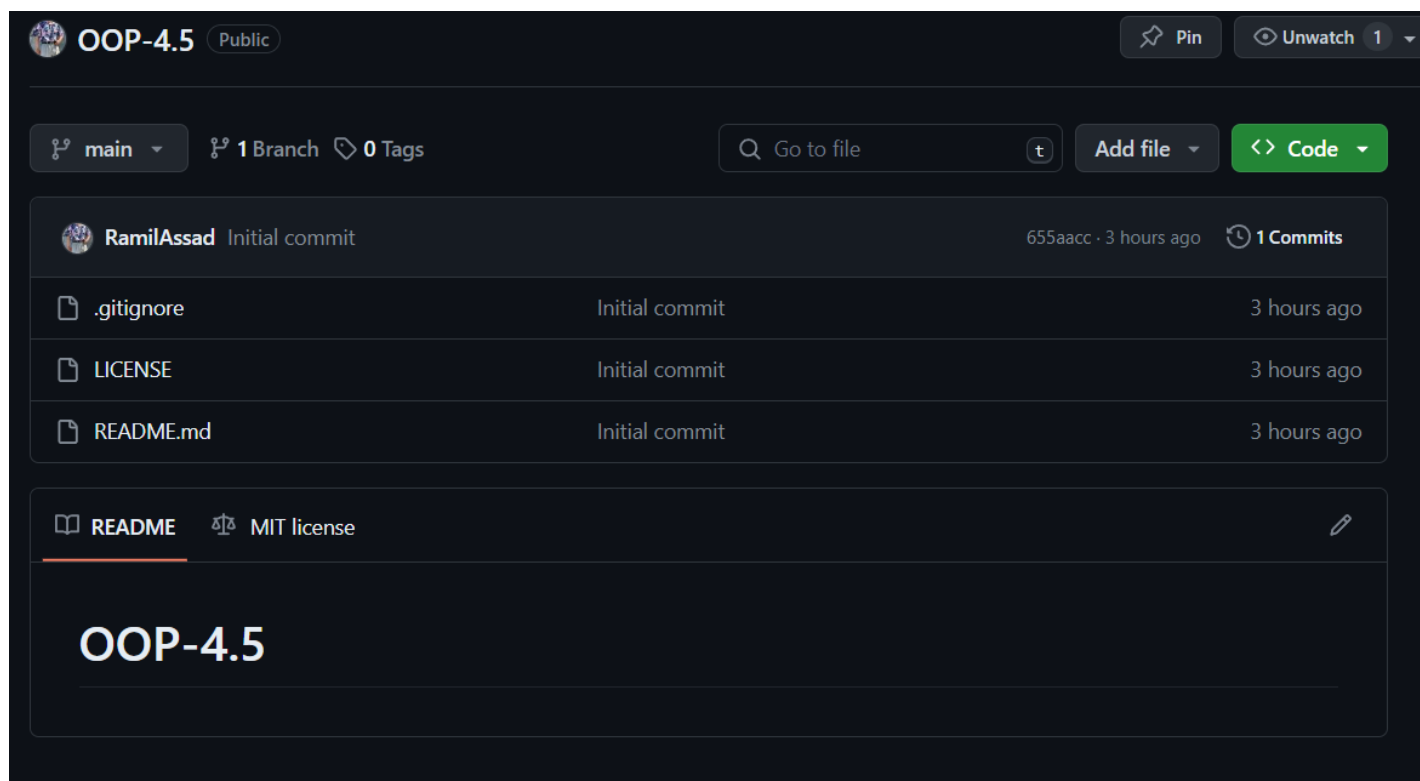


Рисунок 1 - Создание репозитория

2. Выполните клонирование созданного репозитория.

```
PS C:\Users\Asus\OneDrive\Рабочий стол\учёба 3 курс\ООП> git clone https://github.com/RamilAssad/OOP-4.2.git
Cloning into 'OOP-4.2'...
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 5 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (5/5), done.
PS C:\Users\Asus\OneDrive\Рабочий стол\учёба 3 курс\ООП>
```

Рисунок 2 - Клонирование репозитория

3. Организуйте свой репозиторий в соответствие с моделью ветвления git-flow.

```
C:\Users\Asus\Desktop\Учеба\5 семестр\ООП\ООП_lw_4.1>git checkout -b develop
Switched to a new branch 'develop'

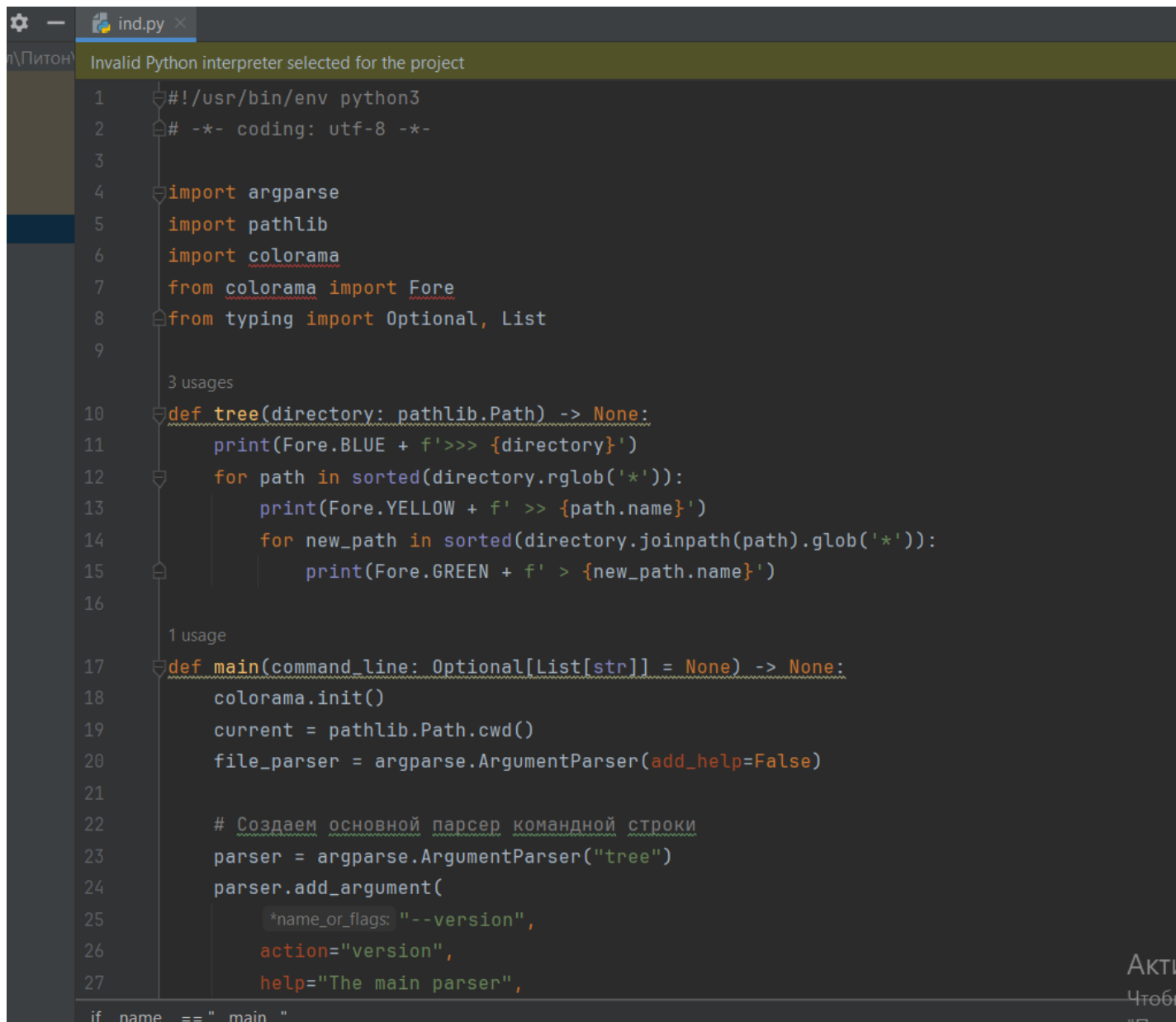
C:\Users\Asus\Desktop\Учеба\5 семестр\ООП\ООП_lw_4.1>
```

Рисунок 3 - Ветвление по модели git-flow

4. Выполнить индивидуальные задания.

### Задание 1. Вариант 15

Выполнить индивидуальное задание 2 лабораторной работы 2.19, добавив аннотации типов. Выполнить проверку программы с помощью утилиты муру.



```
Invalid Python interpreter selected for the project

1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  import argparse
5  import pathlib
6  import colorama
7  from colorama import Fore
8  from typing import Optional, List
9
10  3 usages
11  def tree(directory: pathlib.Path) -> None:
12      print(Fore.BLUE + f'>> {directory}')
13      for path in sorted(directory.rglob('*')):
14          print(Fore.YELLOW + f' >> {path.name}')
15          for new_path in sorted(directory.joinpath(path).glob('*')):
16              print(Fore.GREEN + f' > {new_path.name}')
17
18  1 usage
19  def main(command_line: Optional[List[str]] = None) -> None:
20      colorama.init()
21      current = pathlib.Path.cwd()
22      file_parser = argparse.ArgumentParser(add_help=False)
23
24      # Создаем основной парсер командной строки
25      parser = argparse.ArgumentParser("tree")
26      parser.add_argument(
27          *name_or_flags: "--version",
28          action="version",
29          help="The main parser",
30      )
31
32  if __name__ == "__main__":
```

Рисунок 4 - Код индивидуального задания 1

### Контрольные вопросы:

#### 1. Для чего нужны аннотации типов в языке Python?

Аннотации типов в языке Python используются для явного указания типов данных аргументов функций, возвращаемых значений, и переменных, чтобы облегчить понимание кода и обеспечить инструменты статического анализа.

## **2. Как осуществляется контроль типа в языке Python?**

Контроль типов в Python может осуществляться с помощью сторонних инструментов, таких как майпай, пайчек или использования аннотаций типов в сочетании с статическим анализом кода.

## **3. Какие существуют предложения по усовершенствованию Python для работы с аннотациями типов?**

Существуют предложения по усовершенствованию Python для работы с аннотациями типов, такие как добавление более строгой поддержки статического анализа типов, улучшение синтаксиса аннотаций типов и другие.

## **4.. Как осуществляется аннотирование параметров и возвращаемых значений функций?**

Аннотирование параметров и возвращаемых значений функций происходит путем указания типов данных после двоеточия в определении функции. Например: `def my_function(param: int) -> str:`

## **5. Как выполнить доступ к аннотациям функций?**

Доступ к аннотациям функций можно получить с помощью встроенной функции Python - `annotations()`. Например, если у вас есть функция `function` с аннотацией типа, вы можете получить доступ к аннотациям вот так: `function.__annotations__`.

## **6. Как осуществляется аннотирование переменных в языке Python?**

Аннотирование переменных в Python осуществляется путем указания типа данных после имени переменной с помощью двоеточия. Например: `x: int = 5`.

## **7. Для чего нужна отложенная аннотация в языке Python?**

Отложенная аннотация в Python используется для работы с аннотациями типов, когда тип данных еще не определен в момент выполнения программы. Она может использоваться, например, при работе с модулями, которые позволяют указывать типы данных в аннотациях до того, как они были определены.