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

Институт цифрового развития
Кафедра инфокоммуникаций

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ №4
дисциплины «Программирование на Python»

Выполнил:
Середа Кирилл Витальевич
1 курс, группа ИВТ-б-о-22-1,
09.03.01 «Информатика и вычислительная техника», очная форма обучения

_____
(подпись)

Руководитель практики:
Воронкин Роман Александрович

_____
(подпись)

Отчет защищен с оценкой _____ Дата защиты_____

Ставрополь, 2023 г.

Тема: Основы языка Python

Цель: исследование процесса установки и базовых возможностей языка Python версии 3.x.

Краткое теоретическое обоснование

Основные этапы установки Python в Windows и Linux:

В Windows: a. Скачать установочный файл Python с официального сайта (python.org). b. Запустить установщик и следовать инструкциям. c. Выбрать опцию "Add Python to PATH" (добавить Python в переменную среды PATH) для удобства использования Python из командной строки. d. Завершить процесс установки.

В Linux: a. Многие дистрибутивы Linux уже имеют Python предустановленным. В противном случае, можно установить его с помощью пакетного менеджера вашего дистрибутива, например, в Ubuntu: **sudo apt-get install python3**. b. После установки можно проверить версию с помощью команды **python3 --version**.

Отличие пакета Anaconda от стандартного пакета Python: Anaconda - это дистрибуция Python, предназначенная для научных вычислений и анализа данных. Основное отличие заключается в том, что Anaconda включает в себя множество предустановленных библиотек и инструментов, таких как NumPy, Pandas, Matplotlib, Jupyter и многие другие, что делает ее идеальным выбором для работы в области анализа данных и машинного обучения. Стандартный пакет Python с официального сайта включает только базовые библиотеки.

Проверка работоспособности Anaconda: Для проверки работоспособности Anaconda можно запустить интерактивную оболочку IPython или Jupyter Notebook. Также можно создать новое окружение и установить несколько библиотек для проверки.

Задание интерпретатора Python в PyCharm: В PyCharm можно задать интерпретатор Python в настройках проекта или в глобальных настройках IDE. Для этого перейдите в "File" -> "Settings" (или "Preferences" на macOS) -

> "Project: [имя проекта]" -> "Python Interpreter" и выберите нужный интерпретатор Python.

Запуск программы в PyCharm: Для запуска программы в PyCharm можно нажать кнопку "Run" (Запустить) или "Debug" (Отладка) в верхней панели. Также можно использовать комбинации клавиш, например, Shift + F10 для запуска.

Интерактивный и пакетный режимы Python: Интерактивный режим позволяет вводить команды Python построчно и немедленно видеть результат. Пакетный режим используется для выполнения скриптов и программ, которые выполняются целиком.

Язык Python называется динамическим из-за того, что типы данных переменных определяются автоматически во время выполнения программы, а не во время компиляции.

Основные типы данных в Python включают числа (int, float), строки (str), списки (list), кортежи (tuple), множества (set), словари (dict), булевы значения (bool), и многое другое.

Объекты в памяти создаются при присваивании значений переменным. Переменные - это ссылки на объекты. Процесс объявления переменных заключается в присвоении им значений, и Python автоматически выделяет память для хранения объектов.

Получение списка ключевых слов в Python можно сделать с помощью модуля **keyword**. Используйте **import keyword** и **keyword.kwlist** для получения списка ключевых слов.

Функция **id()** возвращает уникальный идентификатор объекта в памяти, а функция **type()** возвращает тип объекта.

Изменяемые типы данных могут быть изменены после создания, например, списки. Неизменяемые типы данных, такие как кортежи и строки, не могут быть изменены после создания.

Операция деления (/) возвращает результат в виде числа с плавающей точкой, а операция целочисленного деления (//) возвращает результат в виде целого числа.

Для работы с комплексными числами в Python используется встроенный тип **complex**.

Модуль **math** предоставляет функции для математических операций, такие как trigonometry, exponentiation, логарифмы и другие. Модуль **cmath** предоставляет аналогичные функции для комплексных чисел.

Параметры **sep** и **end** в функции **print()** используются для настройки разделителей между значениями и окончания вывода.

Метод **format()** используется для форматирования строк, позволяя вставлять значения в строку. Также в Python есть f-строки для удобного форматирования строк с использованием выражений.

Для ввода значений с консоли в Python используются функции **input()** для строк, **int(input())** для целых чисел и **float(input())** для вещественных чисел.

Ход выполнения:

1) Создал и клонировал репозиторий и организовал работу соответственно модели git-flow

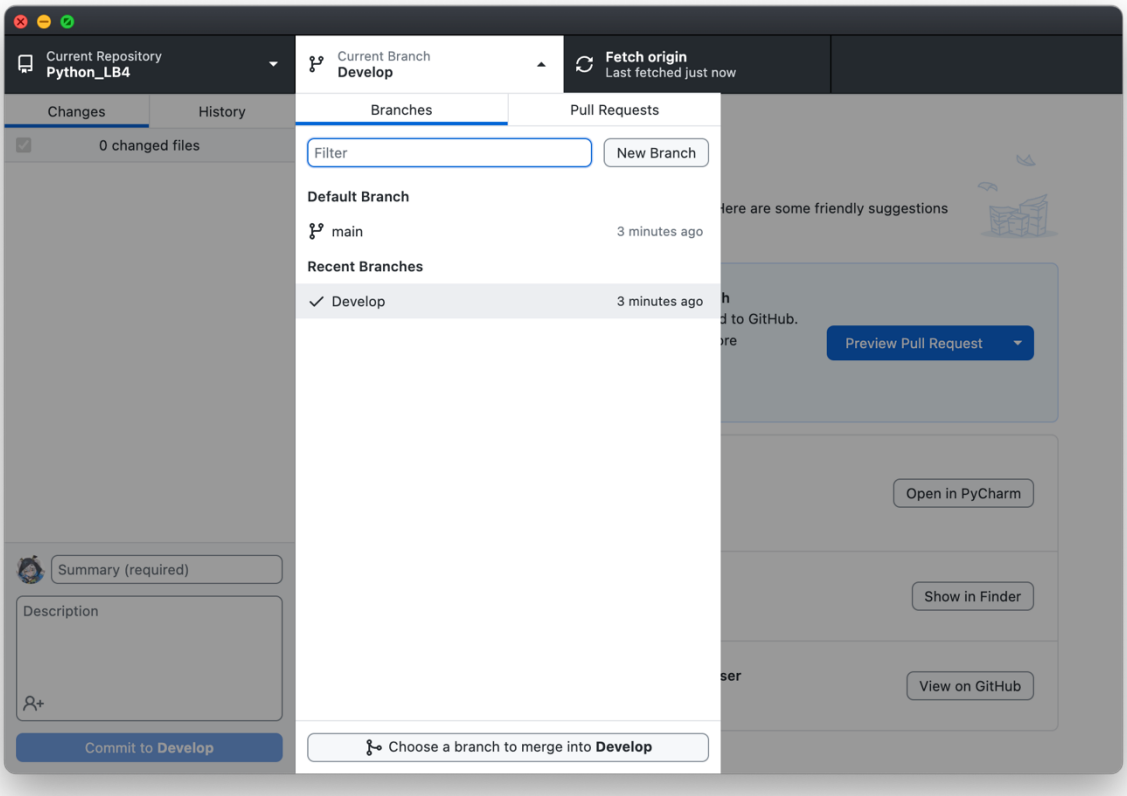

Рисунок 1 – ветки репозитория

2) Дополнил файл .gitignore

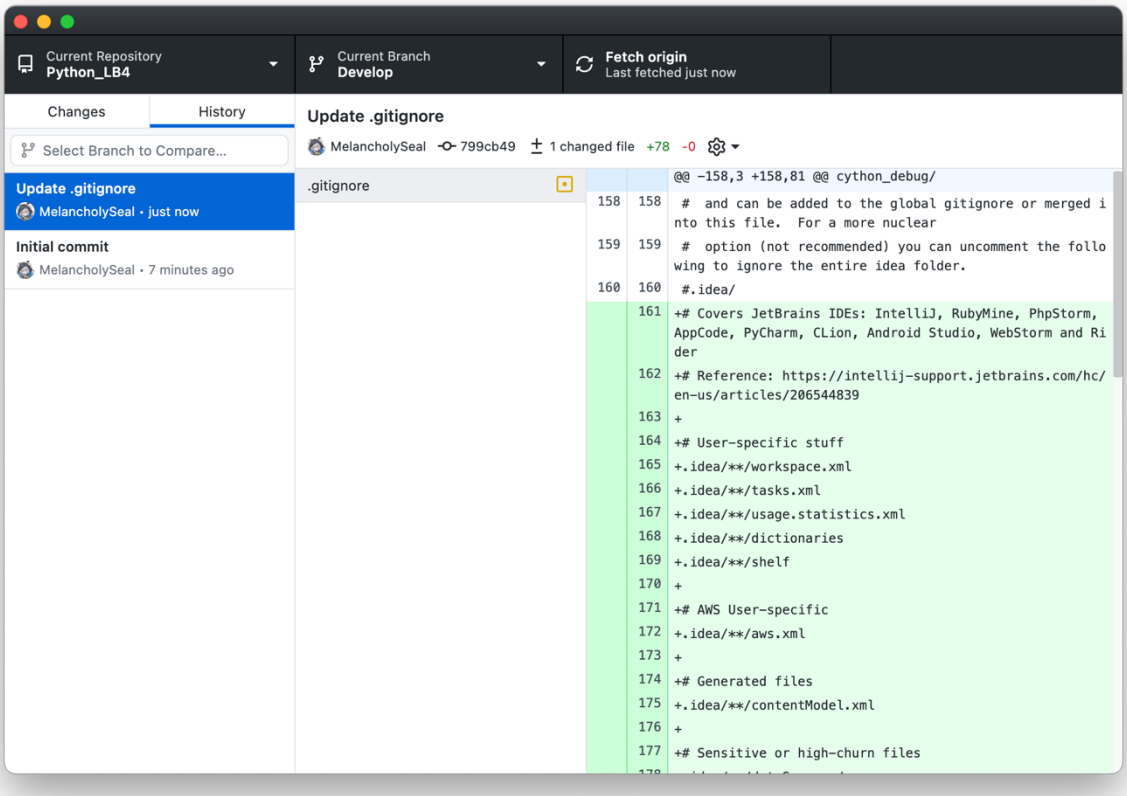

Рисунок 2 – коммит с изменением .gitignore

3) Создал проект в репозитории

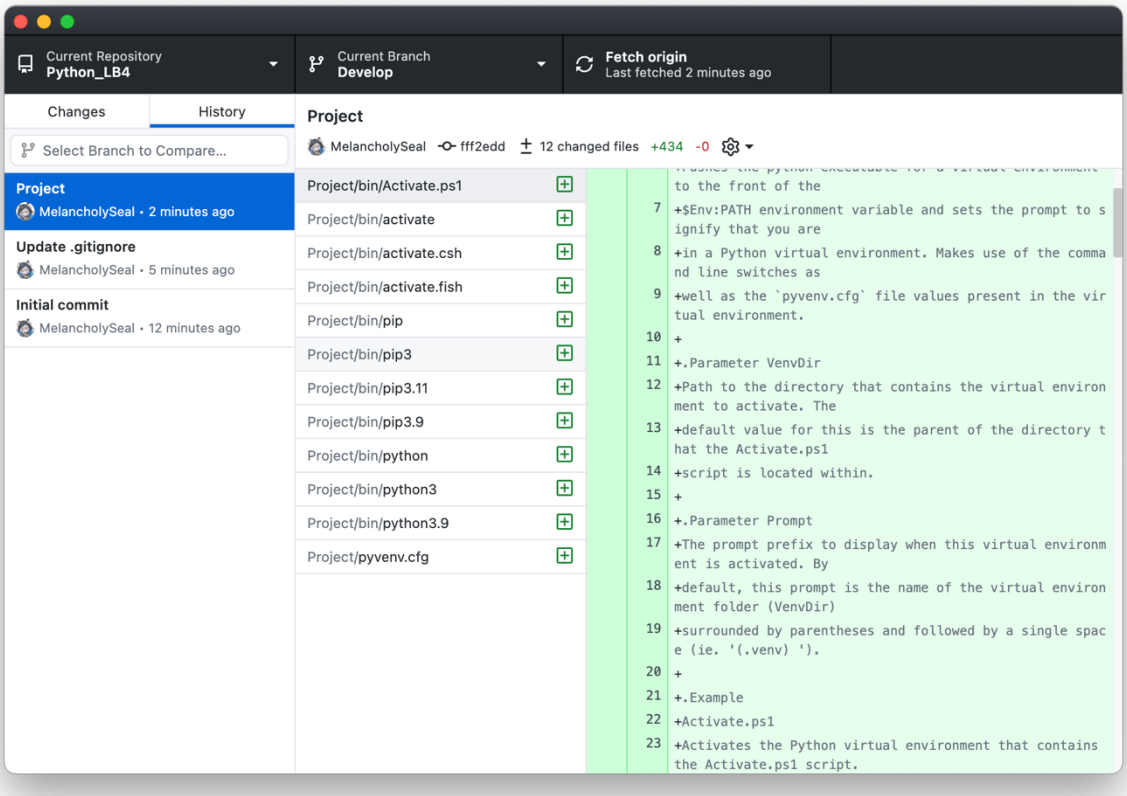

Рисунок 3 – Коммит сделанный после создание проекта

4) Написал программу, которая запрашивает данные пользователя и затем выводит в консоль

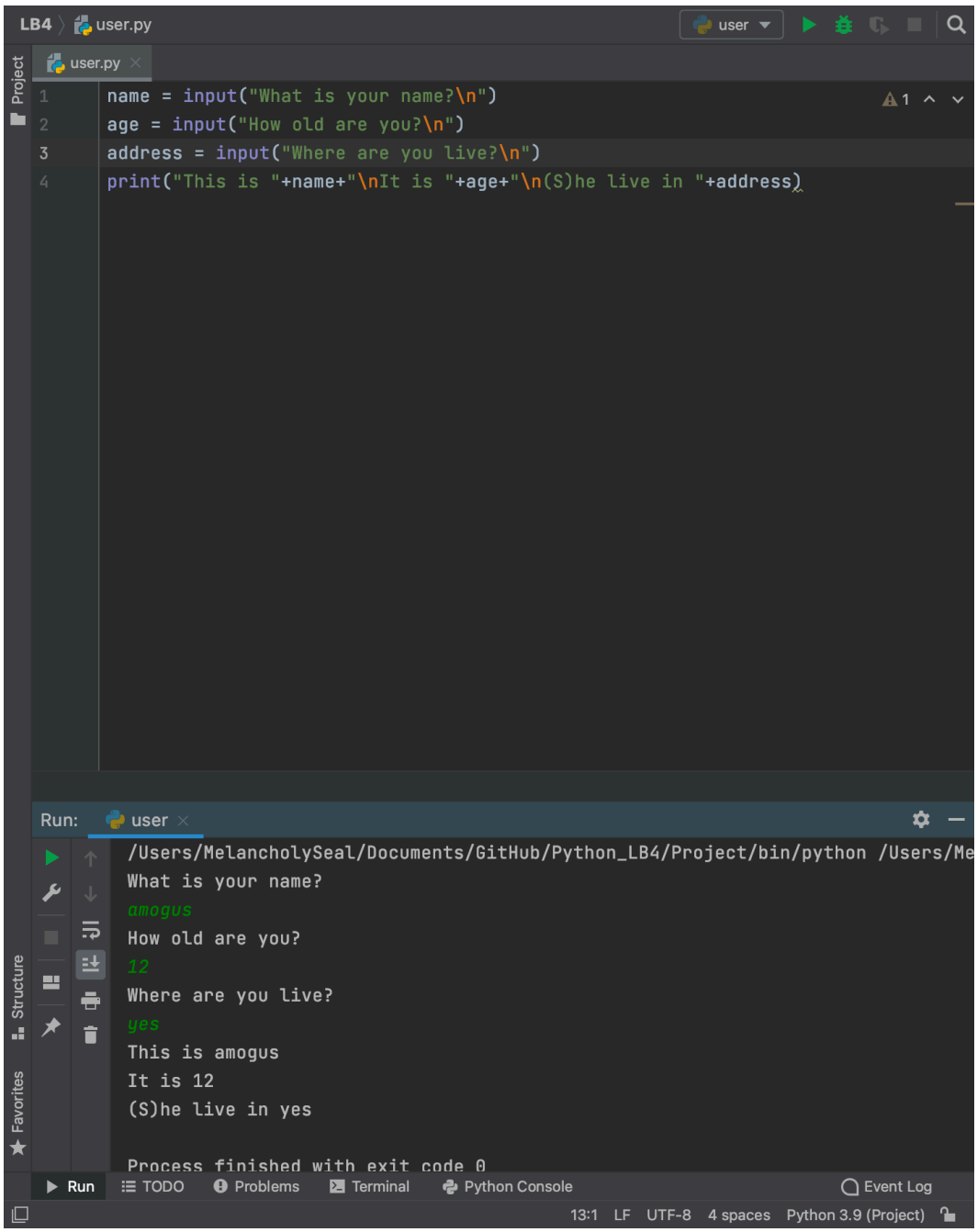

Рисунок 4 – Код программы и результат ее работы

5) Написал программу запрашивающую ответ на пример

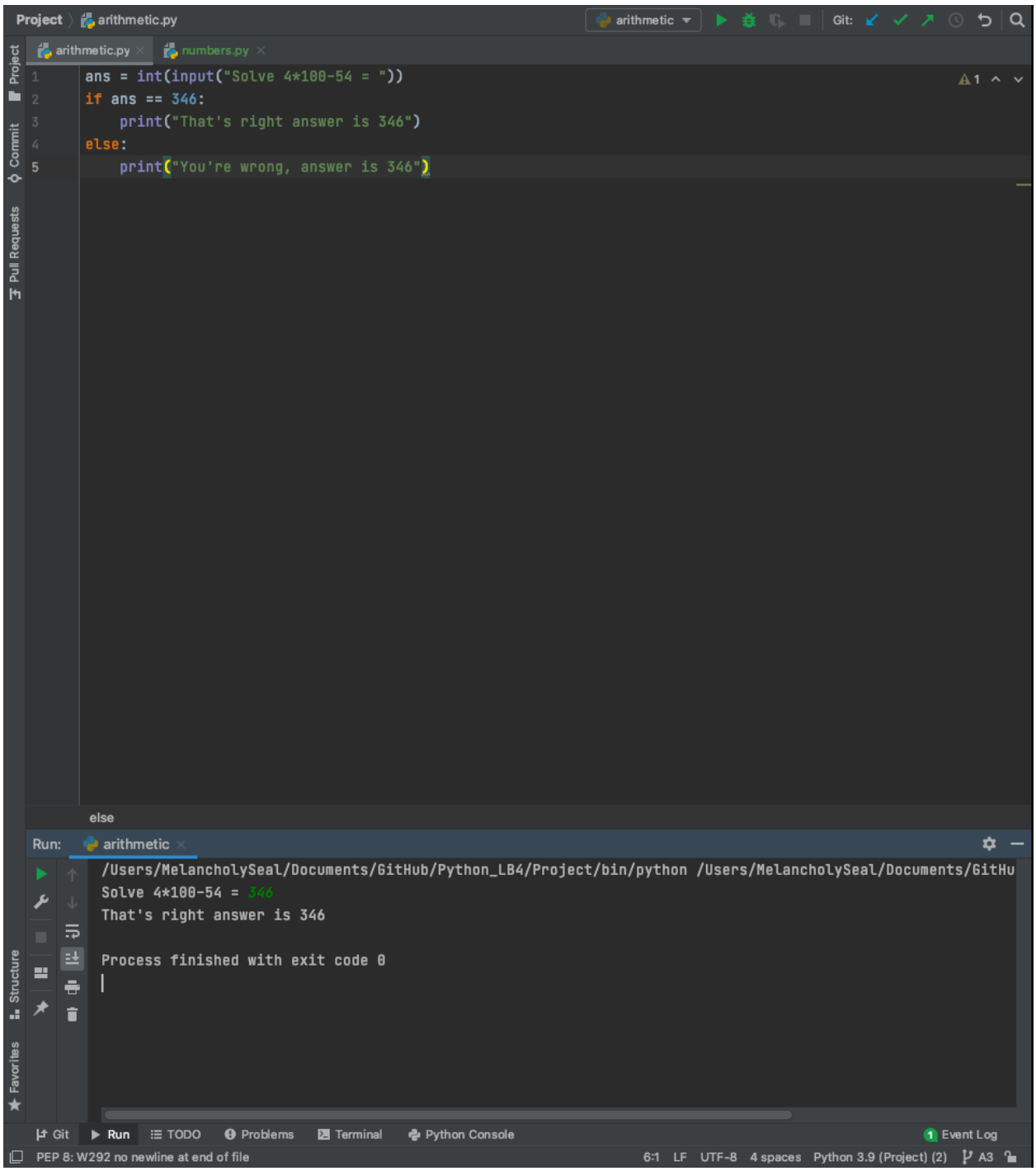

Рисунок 5 – Код программы и результат ее работы

6) Написал программу работающую с цифрами

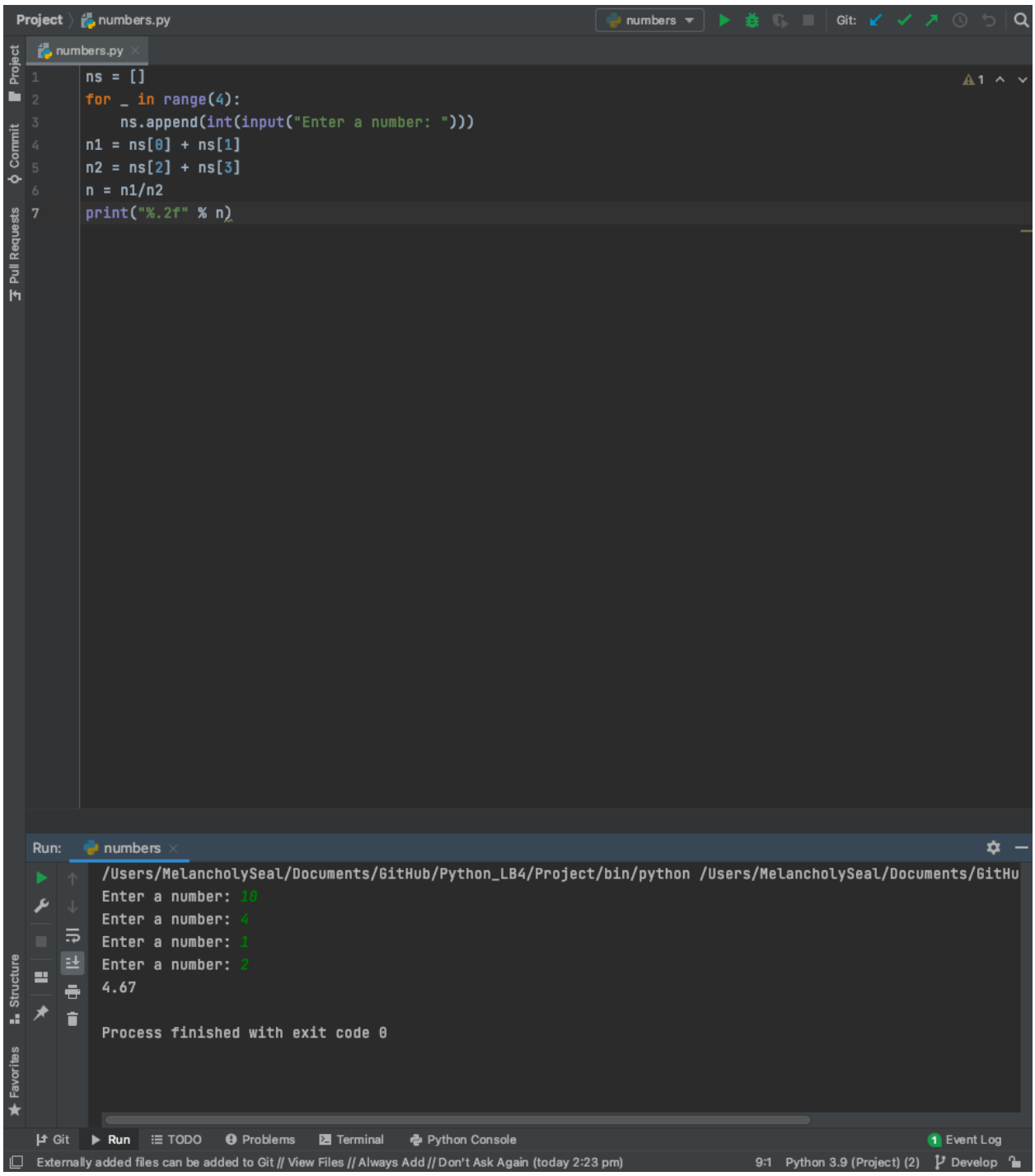

Рисунок 6 - Код программы и результат ее работы

7) Выполнил индивидуальное задание (вариант 6): Даны координаты на плоскости двух точек. Найти расстояние между этими точками

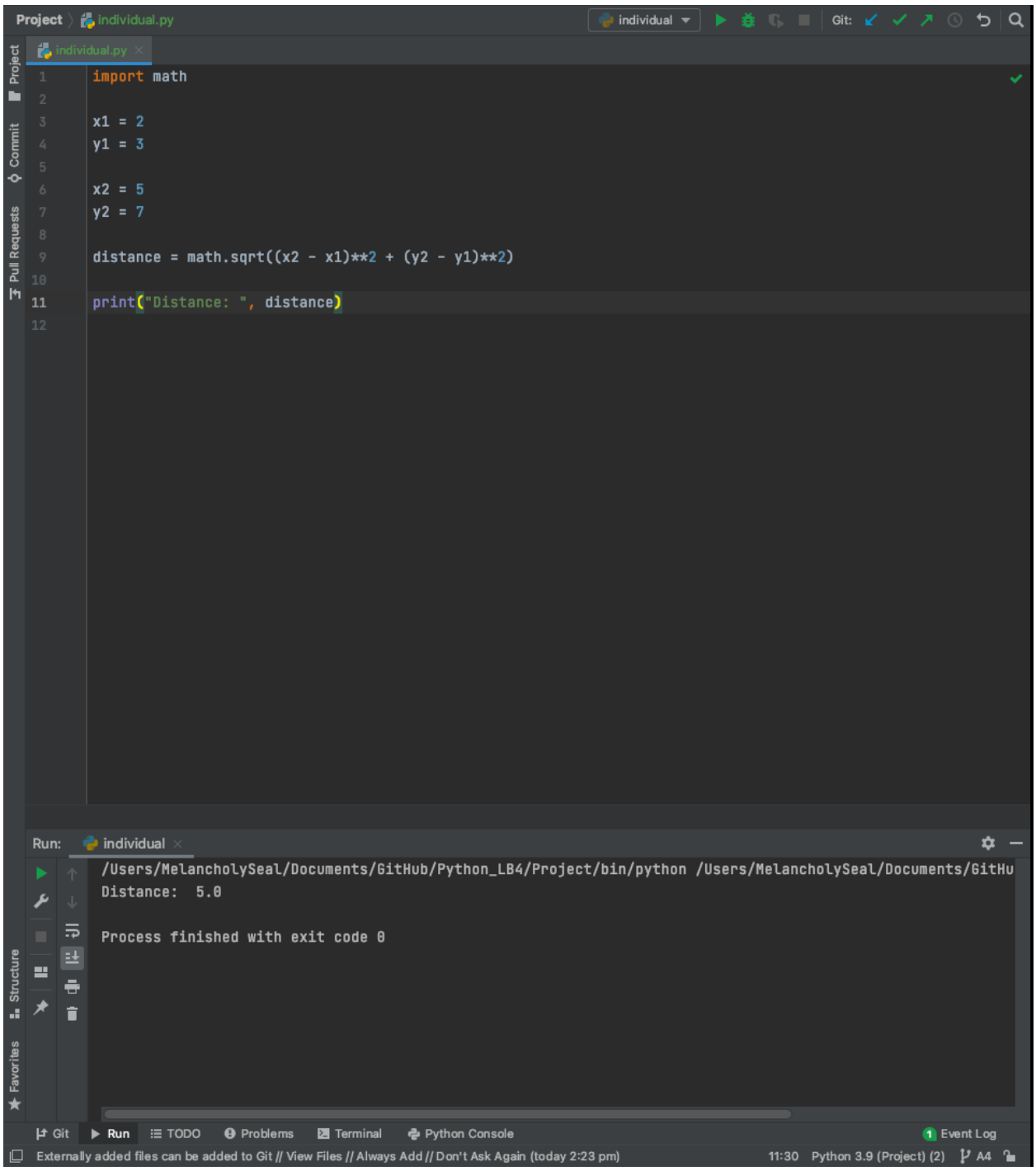

Рисунок 7 - Код программы и результат ее работы

8) Решил задачу повышенной сложности под номером 8: Даны два целых числа a и b. Если a делится на b или b делится на a, то вывести 1, иначе – любое другое число

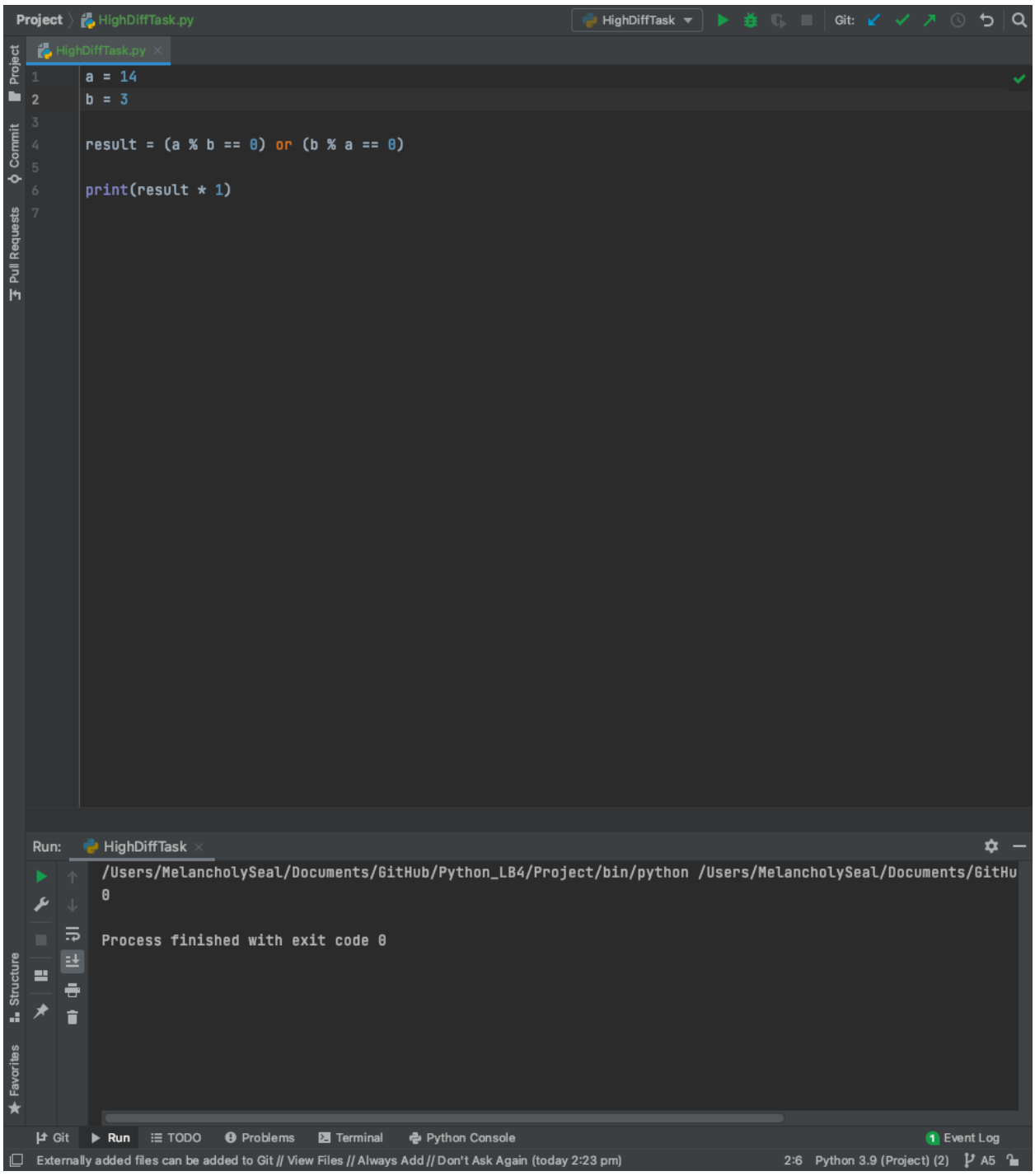

Рисунок 8 - Код программы и результат ее работы

9) Выполнил слияние веток main и Develop, а также отправил изменения на удаленный репозиторий

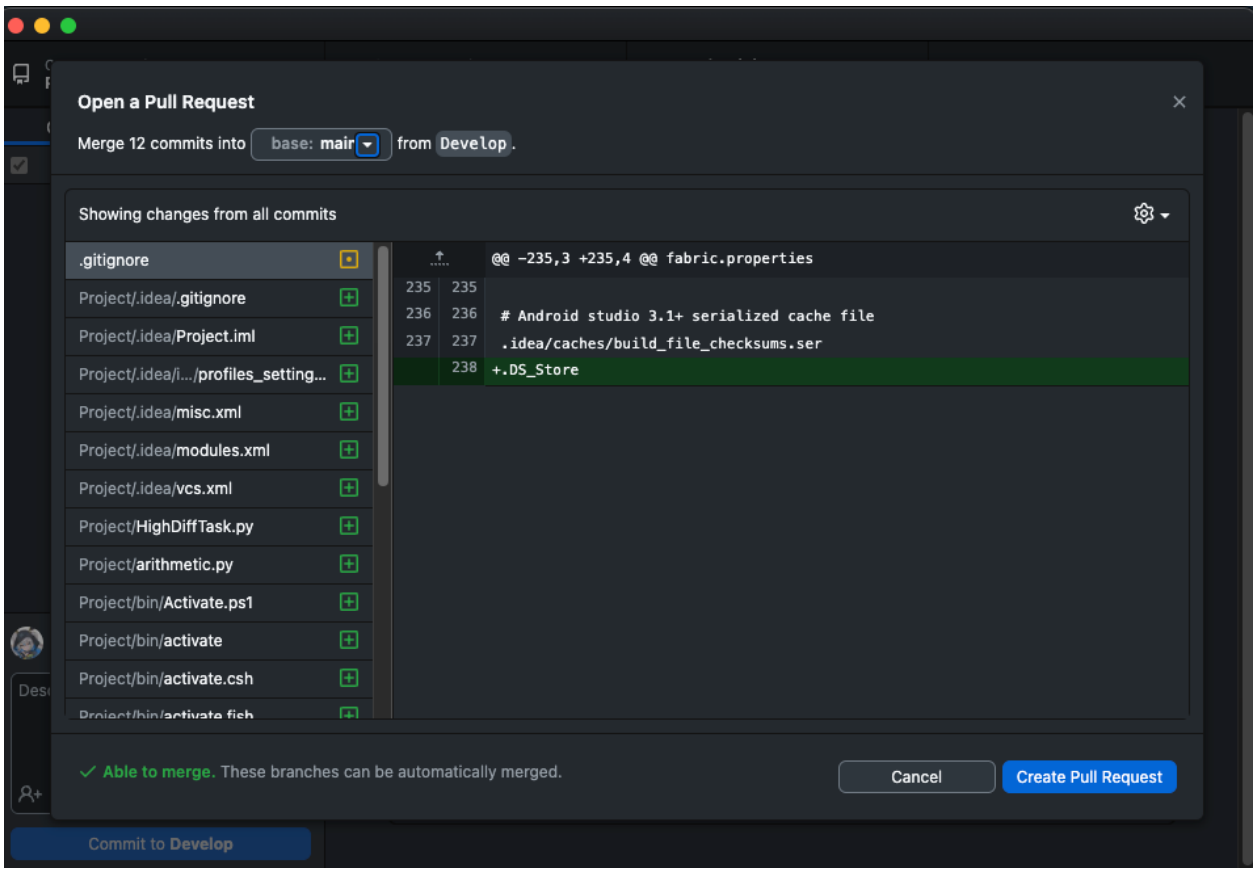

Рисунок 9 – Слияние веток

Ответы на вопросы:

1. Основные этапы установки Python в Windows и Linux:

В Windows: a. Скачать установочный файл Python с официального сайта (python.org). b. Запустить установщик и следовать инструкциям. c. Выбрать опцию "Add Python to PATH" (добавить Python в переменную среды PATH) для удобства использования Python из командной строки. d. Завершить процесс установки.

В Linux: a. Многие дистрибутивы Linux уже имеют Python предустановленным. В противном случае, можно установить его с помощью пакетного менеджера вашего дистрибутива, например, в Ubuntu: **sudo apt-get install python3**. b. После установки можно проверить версию с помощью команды **python3 --version**.

2. Отличие пакета Anaconda от стандартного пакета Python: Anaconda - это дистрибуция Python, предназначенная для научных вычислений и анализа данных. Основное отличие заключается в том, что

Anaconda включает в себя множество предустановленных библиотек и инструментов, таких как NumPy, Pandas, Matplotlib, Jupyter и многие другие, что делает ее идеальным выбором для работы в области анализа данных и машинного обучения. Стандартный пакет Python с официального сайта включает только базовые библиотеки.

3. Проверка работоспособности Anaconda: Для проверки работоспособности Anaconda можно запустить интерактивную оболочку IPython или Jupyter Notebook. Также можно создать новое окружение и установить несколько библиотек для проверки.

4. Задание интерпретатора Python в PyCharm: В PyCharm можно задать интерпретатор Python в настройках проекта или в глобальных настройках IDE. Для этого перейдите в "File" -> "Settings" (или "Preferences" на macOS) -> "Project: [имя проекта]" -> "Python Interpreter" и выберите нужный интерпретатор Python.

5. Запуск программы в PyCharm: Для запуска программы в PyCharm можно нажать кнопку "Run" (Запустить) или "Debug" (Отладка) в верхней панели. Также можно использовать комбинации клавиш, например, Shift + F10 для запуска.

6. Интерактивный и пакетный режимы Python: Интерактивный режим позволяет вводить команды Python построчно и немедленно видеть результат. Пакетный режим используется для выполнения скриптов и программ, которые выполняются целиком.

7. Язык Python называется динамическим из-за того, что типы данных переменных определяются автоматически во время выполнения программы, а не во время компиляции.

8. Основные типы данных в Python включают числа (int, float), строки (str), списки (list), кортежи (tuple), множества (set), словари (dict), булевы значения (bool), и многое другое.

9. Объекты в памяти создаются при присваивании значений переменным. Переменные - это ссылки на объекты. Процесс объявления

переменных заключается в присвоении им значений, и Python автоматически выделяет память для хранения объектов.

10. Получение списка ключевых слов в Python можно сделать с помощью модуля **keyword**. Используйте **import keyword** и **keyword.kwlist** для получения списка ключевых слов.

11. Функция **id()** возвращает уникальный идентификатор объекта в памяти, а функция **type()** возвращает тип объекта.

12. Изменяемые типы данных могут быть изменены после создания, например, списки. Неизменяемые типы данных, такие как кортежи и строки, не могут быть изменены после создания.

13. Операция деления (/) возвращает результат в виде числа с плавающей точкой, а операция целочисленного деления (//) возвращает результат в виде целого числа.

14. Для работы с комплексными числами в Python используется встроенный тип **complex**.

15. Модуль **math** предоставляет функции для математических операций, такие как trigonometry, exponentiation, логарифмы и другие. Модуль **cmath** предоставляет аналогичные функции для комплексных чисел.

16. Параметры **sep** и **end** в функции **print()** используются для настройки разделителей между значениями и окончания вывода.

17. Метод **format()** используется для форматирования строк, позволяя вставлять значения в строку. Также в Python есть f-строки для удобного форматирования строк с использованием выражений.

18. Для ввода значений с консоли в Python используются функции **input()** для строк, **int(input())** для целых чисел и **float(input())** для вещественных чисел.

Вывод: в ходе выполнения лабораторной работы был изучен метод работы с git-flow, а также освоены базовые навыки программирование на Python