# МИНИCTEPCTBO НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙ- СКОЙ ФЕДЕРАЦИИ

**Федеральное государственное автономное образовательное учреждение высшего образования**

# «СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

**Кафедра инфокоммуникаций**

# Основы кроссплатформенного программирования Отчет по лабораторной работе №2.2

Условные операторы и циклы в языке Python

|  |
| --- |
| Выполнил студент группы  ИВТ-б-о-21-1 |
| Урусов.М« » 20 г. |
| Подпись студента |
| Работа защищена « » 20 г. |
| Проверил доцент  Кафедры инфокоммуникаций, старший преподаватель  Воронкин Р.А.  (подпись) |

Ставрополь 2022

**Тема:** Условные операторы и циклы в языке Python

**№1.** Создать новый репозиторий и проработать примеры из лаборатор- ной работы.

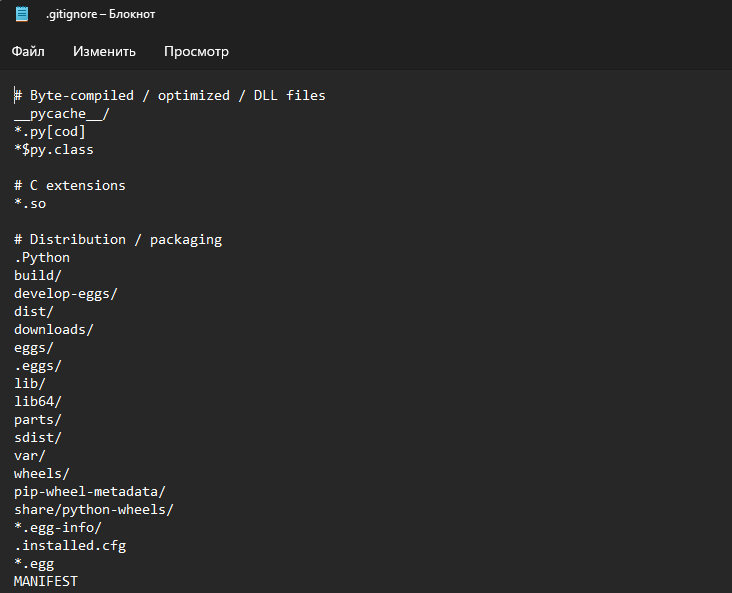


Рисунок 1 – Изменённый файл .gitignore

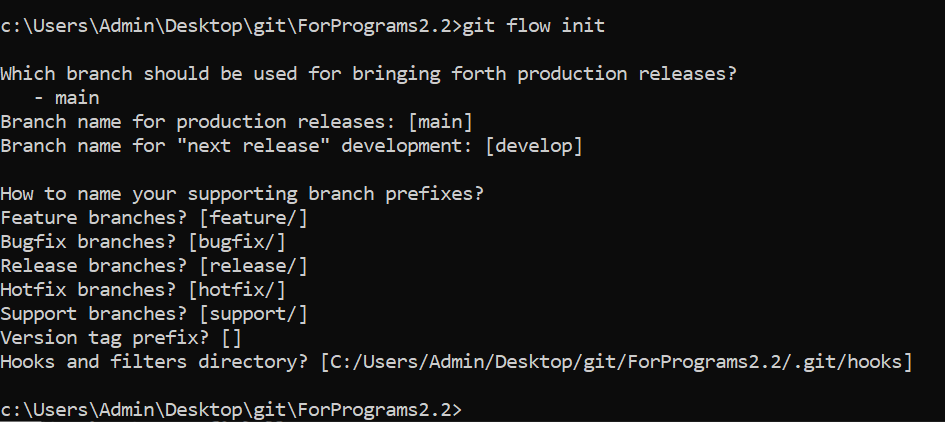


Рисунок 2 – Организация репозитория согласно модели git-flow

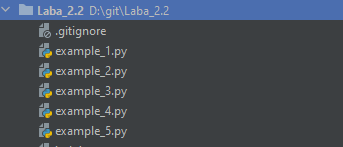


Рисунок 3 – Все проработанные примеры в лабораторной работе

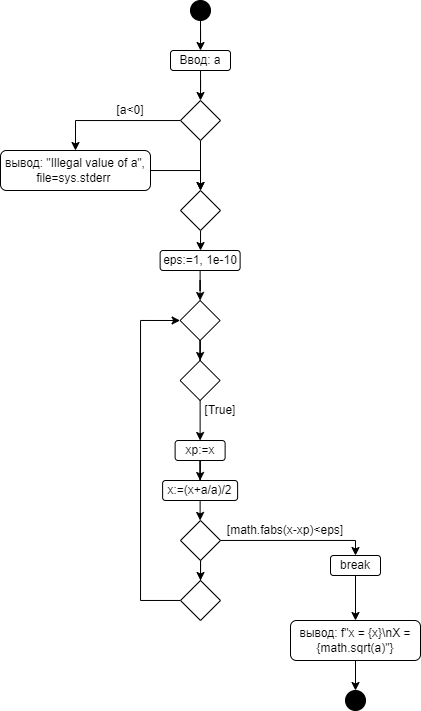


Рисунок 4 – UML-диаграмма для примера №4

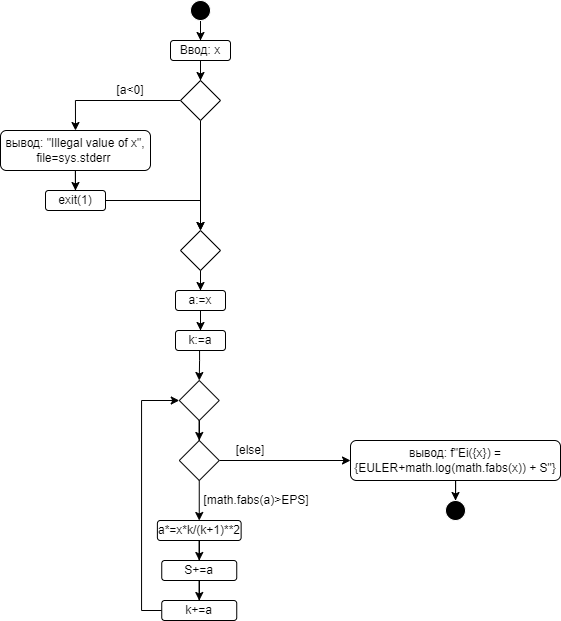


Рисунок 5 – UML-диаграмма для примера №5

**№2.** Выполнить индивидуальные задания и задание повышенной слож- ности согласно своему варианту(В-3). Построить UML-диаграммы написан- ных программ.

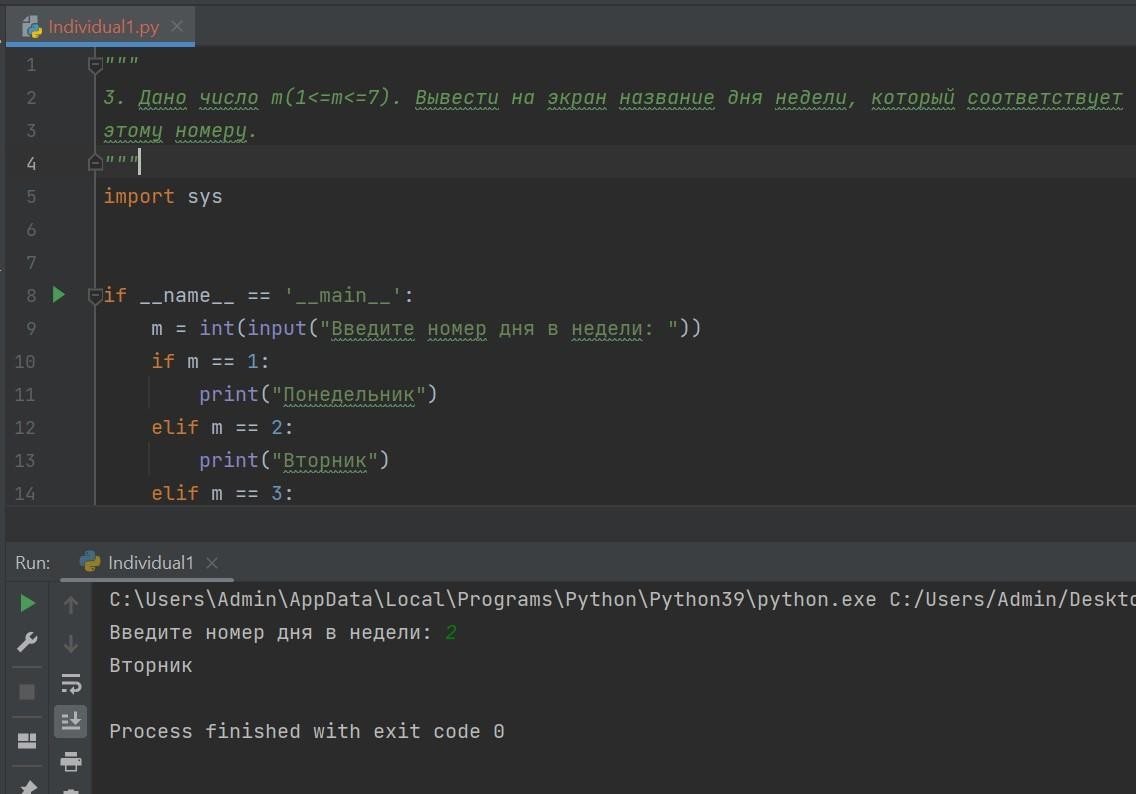


Рисунок 2.1 – Индивидуальное задание №1

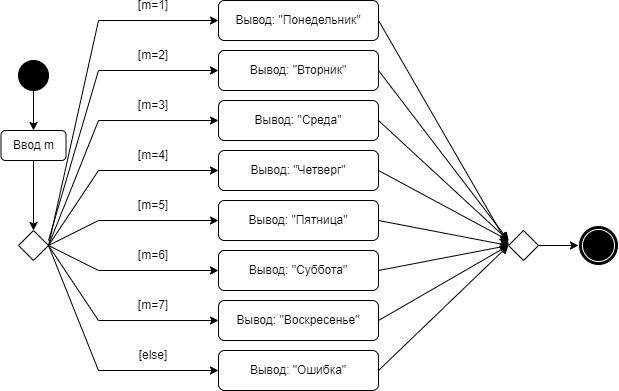


Рисунок 2.2 – UML-диаграмма для индивидуального задания №1

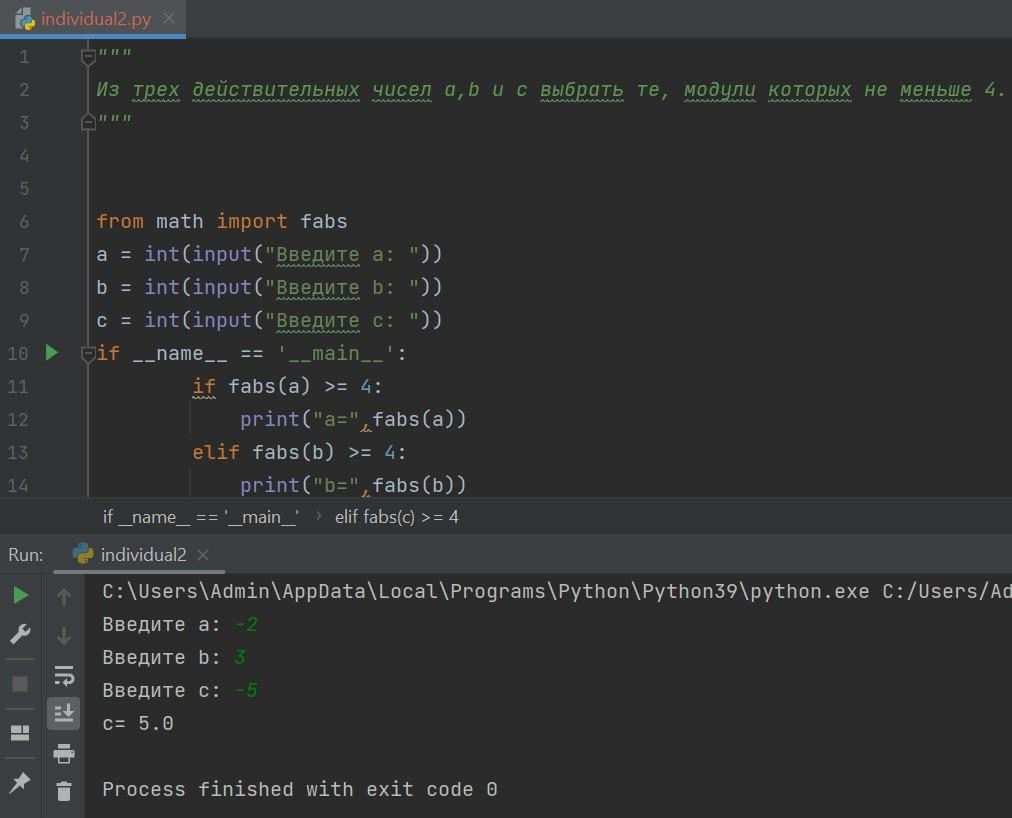


Рисунок 2.3 – Индивидуальное задание №2

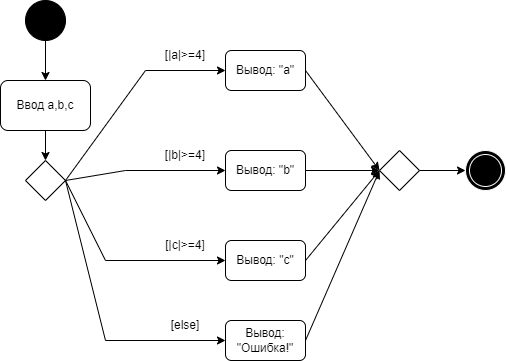
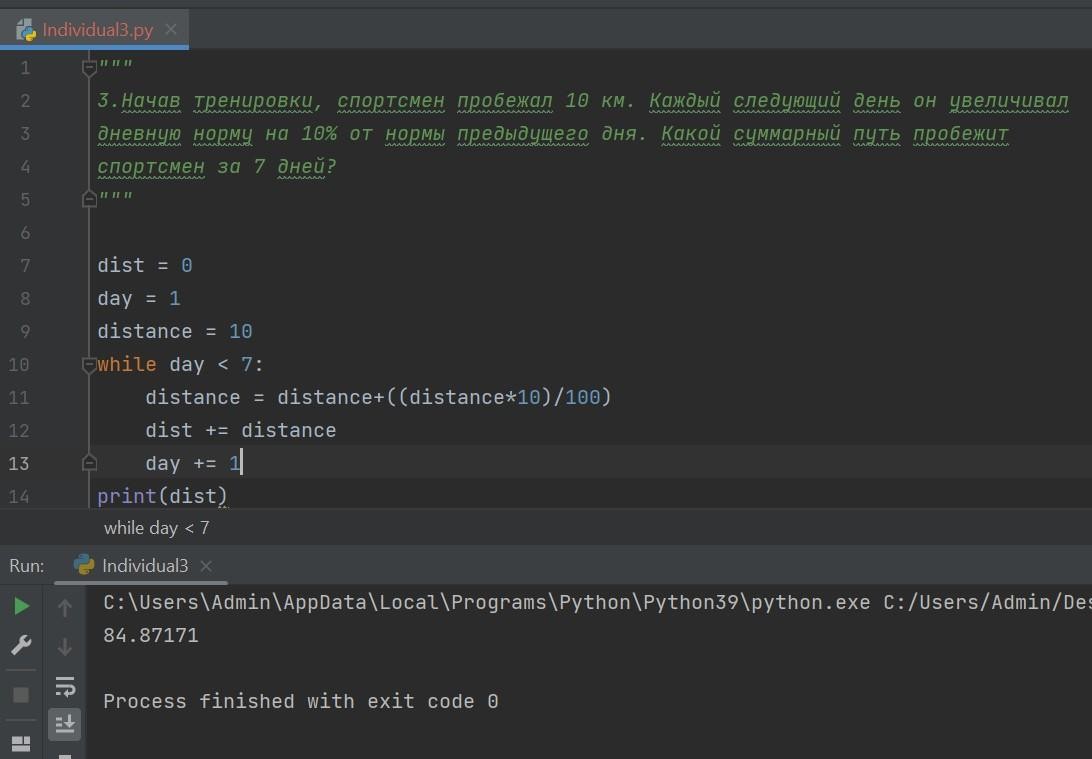


Рисунок 2.4 – UML-диаграмма для индивидуального задания №2



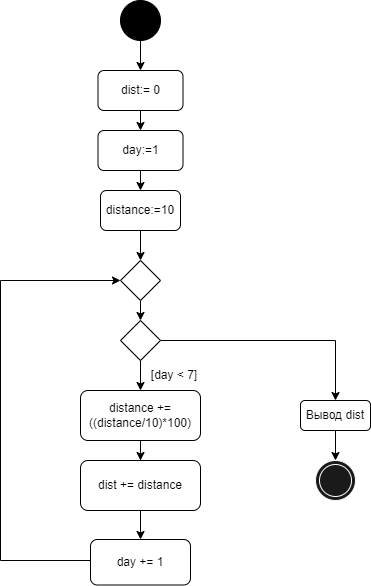
Рисунок 2.5 – Индивидуальное задание №3

Рисунок 2.6 – UML-диаграмма для индивидуального задания №3

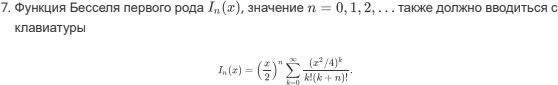


Рисунок 2.7 – Задание повышенной сложности В- 16

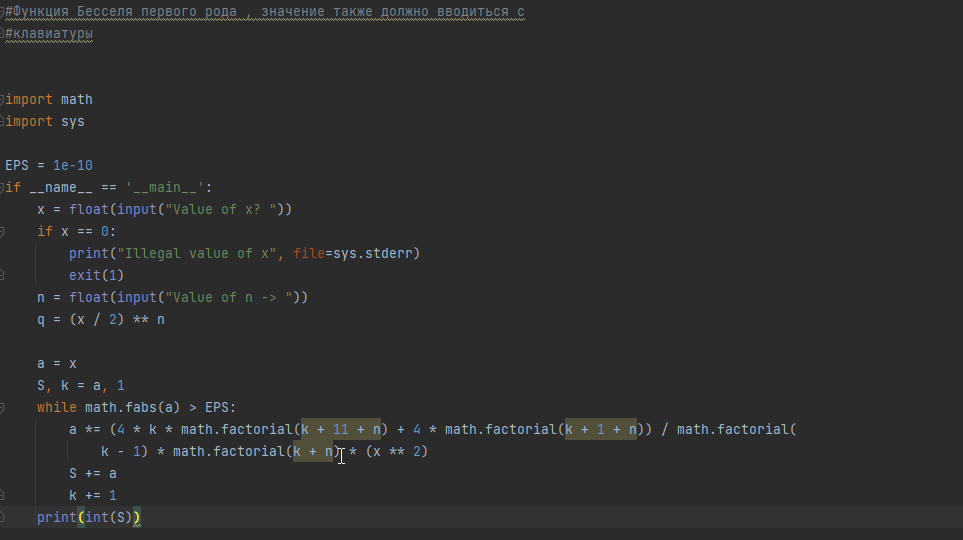


Рисунок 2.8 – Задание повышенной сложности В-3

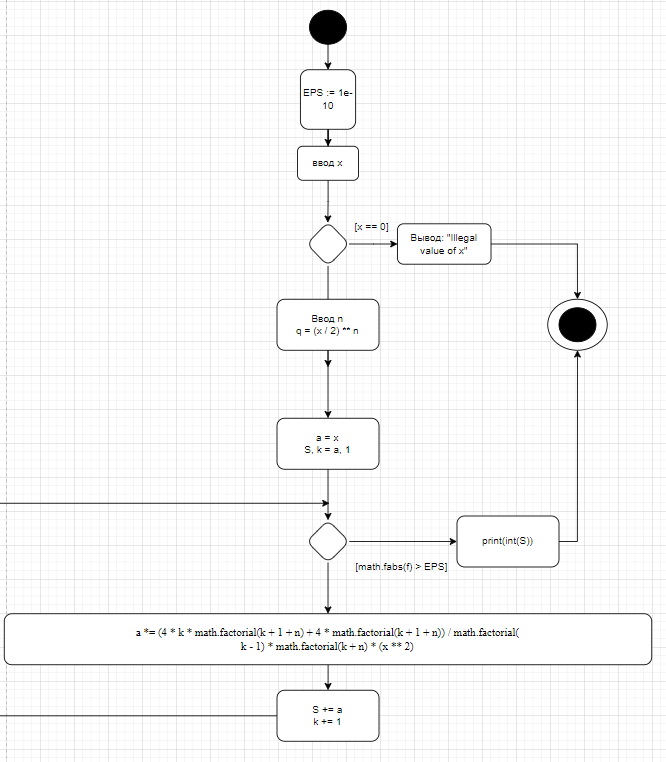


Рисунок 2.9 – UML-диаграмма для задачи повышенной сложности В-16

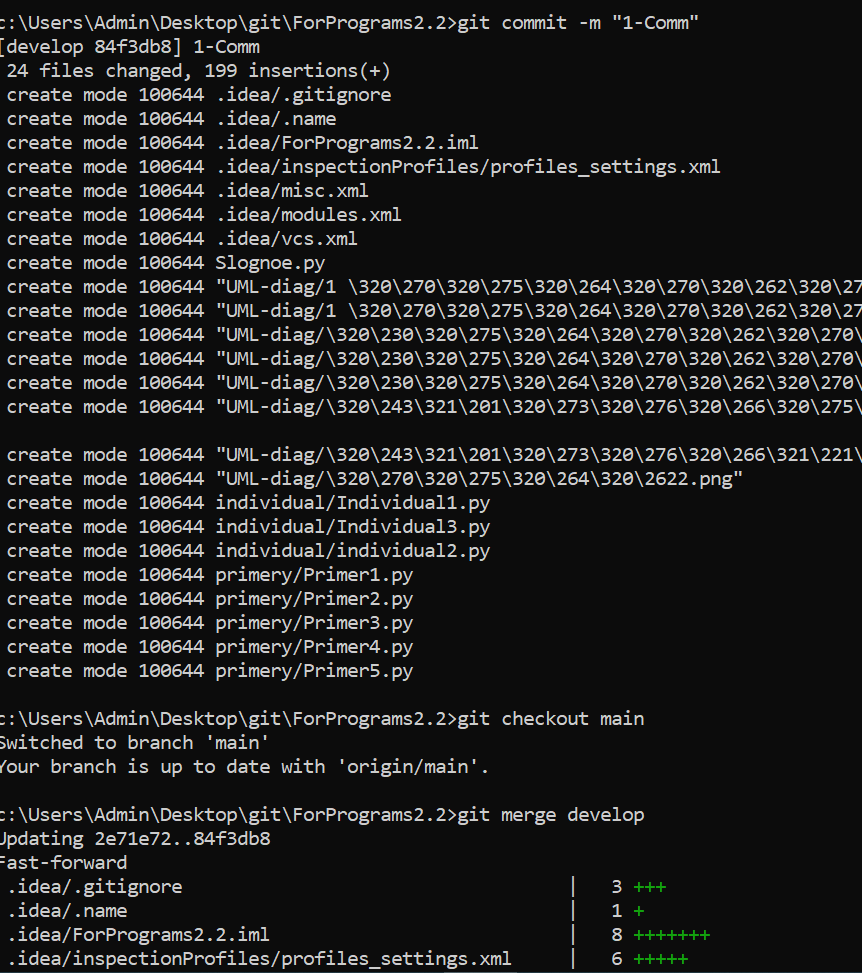


Рисунок 3.1 - Слияние ветки develop с веткой main

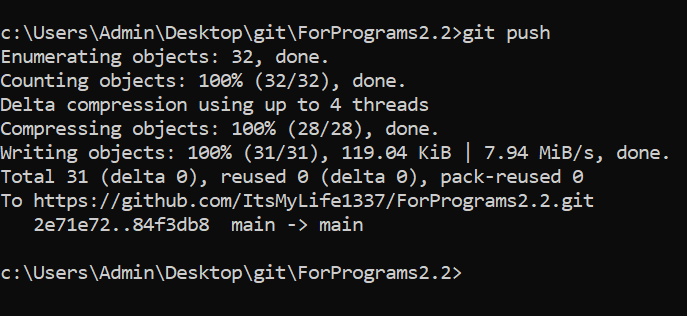


Рисунок 3.2 – Отправка изменений на удалённый репозиторий

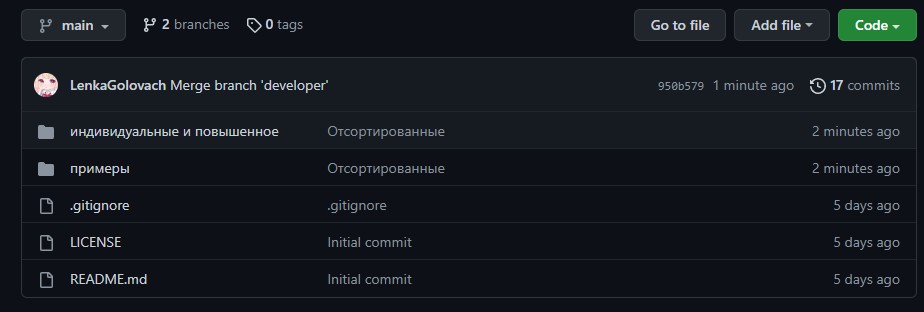


Рисунок 3.3 – Зафиксировал изменения на удалённом репозитории

# Ответы на контрольные вопросы:

1. **Для чего нужны диаграммы деятельности UML?**

Позволяет наглядно визуализировать алгоритм программы.

# Что такое состояние действия и состояние деятельности?

Состояние действия - частный вид состояния деятельности, а конкретнее

– такое состояние, которое не может быть подвергнуто дальнейшей декомпо- зиции.

Состояние деятельности можно представить как составное состояние, поток управления которого включает только другие состояния деятельности и действий.

# Какие нотации существуют для обозначения переходов и ветвле- ний в диаграммах деятельности?

Переходы, ветвление, алгоритм разветвляющейся структуры, алгоритм циклической структуры.

# Какой алгоритм является алгоритмом разветвляющейся струк- туры?

Алгоритм разветвляющейся структуры - это алгоритм, в котором вычис- лительный процесс осуществляется либо по одной, либо по другой ветви, в зависимости от выполнения некоторого условия.

# Чем отличается разветвляющийся алгоритм от линейного?

Линейный алгоритм - алгоритм, все этапы которого выполняются одно- кратно и строго последовательно.

Разветвляющийся алгоритм - алгоритм, содержащий хотя бы одно усло- вие, в результате проверки которого ЭВМ обеспечивает переход на один из нескольких возможных шагов.

# Что такое условный оператор? Какие существуют его формы?

Оператор, конструкция языка программирования, обеспечивающая вы- полнение определённой команды (набора команд) только при условии истин- ности некоторого логического выражения, либо выполнение одной из не- скольких команд.

Условный оператор имеет полную и краткую формы.

# Какие операторы сравнения используются в Python?

If, elif, else

# Что называется простым условием? Приведите примеры.

Простым условием называется выражение, составленное из двух ариф- метических выражений или двух текстовых величин.

Пример: a == b

# Что такое составное условие? Приведите примеры.

Составное условие – логическое выражение, содержащее несколько про- стых условий объединенных логическими операциями. Это операции not, and, or.

Пример: (a == b or a == c)

# Какие логические операторы допускаются при составлении сложных условий?

not, and, or.

# Может ли оператор ветвления содержать внутри себя другие ветвления?

Может.

# Какой алгоритм является алгоритмом циклической структуры?

Циклический алгоритм — это вид алгоритма, в процессе выполнения ко- торого одно или несколько действий нужно повторить.

# Типы циклов в языке Python.

В Python есть 2 типа циклов: - цикл while, - цикл for.

# Назовите назначение и способы применения функции range.

Функция range генерирует серию целых чисел, от значения start до stop, указанного пользователем. Мы можем использовать его для цикла for и

обходить весь диапазон как список.

# Как с помощью функции range организовать перебор значений от 15 до 0 с шагом 2?

Range (15, 0, 2)

# Могул ли быть циклы вложенными?

Могут.

# Как образуется бесконечный цикл и как выйти из него?

Бесконечный цикл в программировании — цикл, написанный таким об- разом, что условие выхода из него никогда не выполняется.

# Для чего нужен оператор break?

Используется для выхода из цикла.

# Где употребляется оператор continue и для чего он используется?

Оператор continue используется только в циклах. В операторах for , while

, do while , оператор continue выполняет пропуск оставшейся части кода тела цикла и переходит к следующей итерации цикла.

# Для чего нужны стандартные потоки stdout и stderr?

Ввод и вывод распределяется между тремя стандартными потоками: stdin — стандартный ввод (клавиатура), stdout — стандартный вывод

(экран),

stderr — стандартная ошибка (вывод ошибок на экран)

# Как в Python организовать вывод в стандартный поток stderr?

Указать в print (…, file=sys.stderr).

# Каково назначение функции exit?

Функция exit() модуля sys - выход из Python.

**Вывод:** в результате выполнения работы были приобретены навыки программирования разветвляющихся алгоритмов и алгоритмов циклической структуры. Также, освоены операторы языка Python версии 3.x if , while , for , break и continue , позволяющих реализовывать разветвляющиеся алгоритмы и алгоритмы циклической структуры.