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 ПРАКТИЧЕСКОЙ РАБОТЕ №13
дисциплины «Программирование на Python»
Вариант 23

Выполнил:
Мотовилов Вадим Борисович
2 курс, группа ИВТ-б-о-22-1,
09.03.01 «Информатика и
вычислительная техника»,
направленность (профиль)
«Информатика и вычислительная
техника», очная форма обучения

_____
(подпись)

Руководитель практики:
Воронкин Роман Александрович
_____
(подпись)

Отчет защищен с оценкой _____ Дата защиты_____

Ставрополь, 2023 г.

Порядок выполнения работы:

1. Создал репозиторий и скопировал его

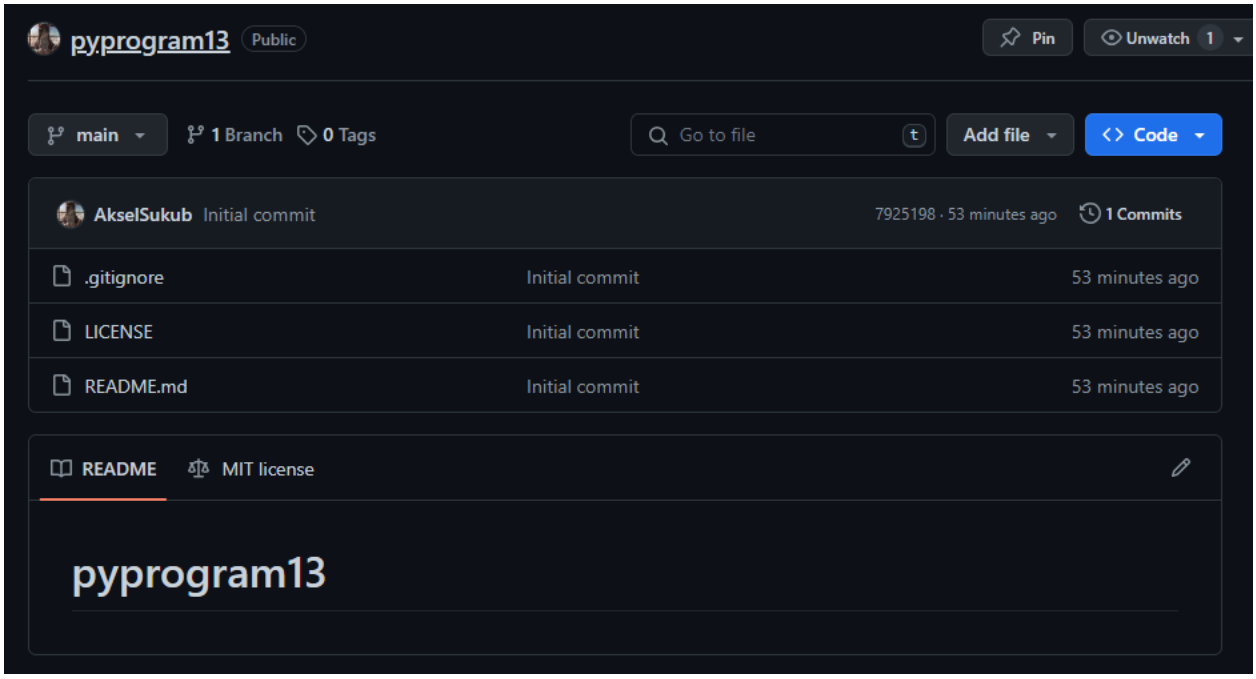

Рисунок 1. Созданный репозиторий

2. Изменил файл .gitignore и README.rm и добавил git flow

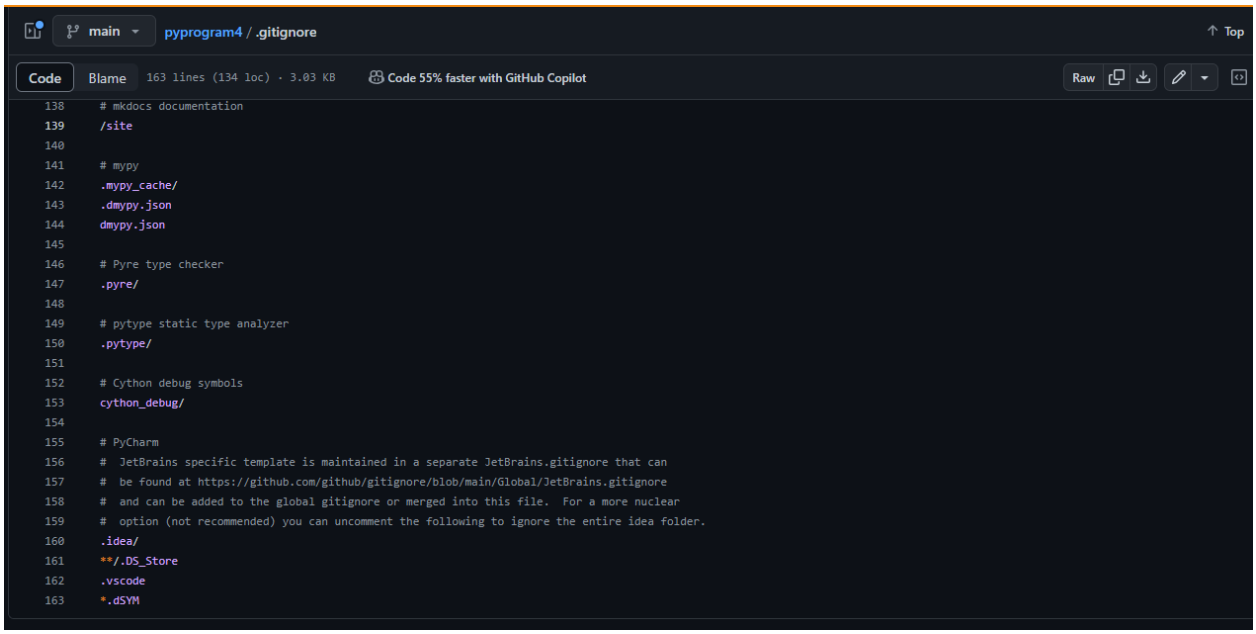

Рисунок 2. Измененный файл .gitignore

3. Выполнил задания

8. Решить поставленную задачу: написать функцию, вычисляющую среднее геометрическое своих аргументов $a_1, a_2, \ldots a_n$

$$G = \sqrt[n]{\prod_{k=1}^{n} a_k}. \tag{1}$$

Если функции передается пустой список аргументов, то она должна возвращать значение `None`.

9. Решить поставленную задачу: написать функцию, вычисляющую среднее гармоническое своих аргументов $a_1, a_2, \ldots, a_n$

$$\frac{n}{H} = \sum_{k=1}^{n} \frac{1}{a_k}. \tag{2}$$

Если функции передается пустой список аргументов, то она должна возвращать значение `None`.

zadanie1.py > calculate_geometric_mean

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

from math import prod


def calculate_geometric_mean(*args):
    if not args:
        return None

    product = prod(args)
    geometric_mean = product ** (1 / len(args))
    return geometric_mean
```

zadanie2.py > calculate_harmonic_mean

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

def calculate_harmonic_mean(*args):
    if not args:
        return None

    reciprocal_sum = sum(1 / num for num in args)
    harmonic_mean = len(args) / reciprocal_sum
    return harmonic_mean
```

Рисунок 3 . Выполнение заданий

Сумму аргументов, расположенных до последнего положительного аргумента.

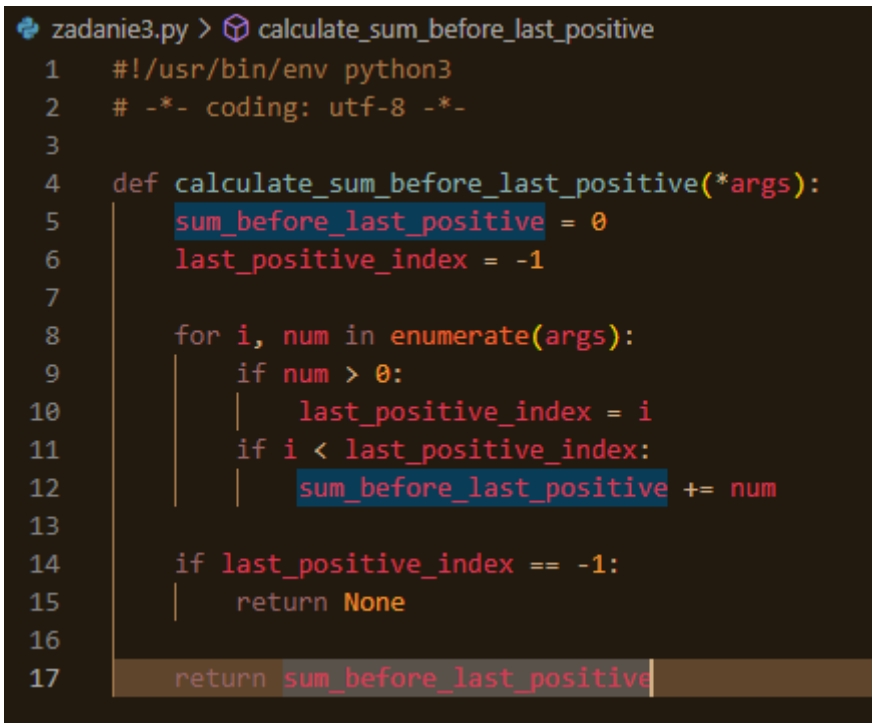

Рисунок 4 . Выполнение индивидуального задания

Ответы на контрольные вопросы:

1. В Python аргументы, которые передаются в функцию без явного указания имени, называются позиционными аргументами. Они связываются с параметрами функции в порядке их расположения.

Пример:

def greet(name, age):

print(f"Привет, {name}! Тебе {age} лет.")

greet("Алиса", 25)

В этом примере "Алиса" и 25 - позиционные аргументы, которые связываются с параметрами name и age соответственно.

2. В Python аргументы, которые передаются в функцию с явным указанием имени, называются именованными аргументами. При использовании именованных аргументов порядок передачи не имеет значения.

Пример:

def greet(name, age):

print(f"Привет, {name}! Тебе {age} лет.")

```
greet(age=25, name="Алиса")
```

В этом примере мы передаем аргументы age и name с явным указанием их имени.

3. Оператор * в Python используется для распаковки последовательности (например, списка или кортежа) в аргументы функции. Это позволяет передавать переменное количество аргументов в функцию.

Пример:

```
def add_numbers(a, b, c):
return a + b + c

numbers = [1, 2, 3]
result = add_numbers(*numbers)
print(result)  # Вывод: 6
```

В этом примере мы используем оператор * для распаковки списка numbers и передачи его элементов в функцию add_numbers в качестве аргументов.

4. Конструкции *args и **kwargs в Python используются для обработки переменного числа аргументов в функциях.

*args позволяет передавать произвольное количество позиционных аргументов в функцию. Аргументы будут доступны внутри функции в виде кортежа.

Пример:

```
def print_arguments(*args):
for arg in args:
print(arg)

print_arguments("Алиса", "Боб", "Кэрол")
```

В этом примере функция print_arguments принимает произвольное количество позиционных аргументов и выводит их на экран.

**kwargs позволяет передавать произвольное количество именованных аргументов в функцию. Аргументы будут доступны внутри функции в виде словаря.

Пример:

```
def print_person_info(**kwargs):
for key, value in kwargs.items():
print(f"{key}: {value}")

print_person_info(name="Алиса", age=25, city="Москва")
```

В этом примере функция print_person_info принимает произвольное количество именованных аргументов и выводит их на экран.