

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 №2.10**  
**дисциплины «Основы кроссплатформенного программирования»**

Выполнила:  
Мурашко Анастасия Юрьевна  
1 курс, группа ИТС-б-о-22-1,  
11.03.02 «Инфокоммуникационные  
технологии и системы связи»,  
направленность (профиль)  
«Инфокоммуникационные системы и  
сети», очная форма обучения

---

(подпись)

Руководитель практики:  
Воронкин Р. А., доцент кафедры  
инфокоммуникаций

---

(подпись)

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

Ставрополь, 2023 г.

**Тема:** функции с переменным числом параметров в Python.

**Цель:** приобретение навыков по работе с функциями с переменным числом параметров при написании программ с помощью языка программирования Python версии 3.x.

### Ход работы:

**Задание 1.** Создала проект PyCharm в папке репозитория. Приступила к работе с примером. Добавил новый файл primer.py.

```
C:\Users\user>git clone https://github.com/welli/laba2-10.git
Cloning into 'laba2-10'...
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.

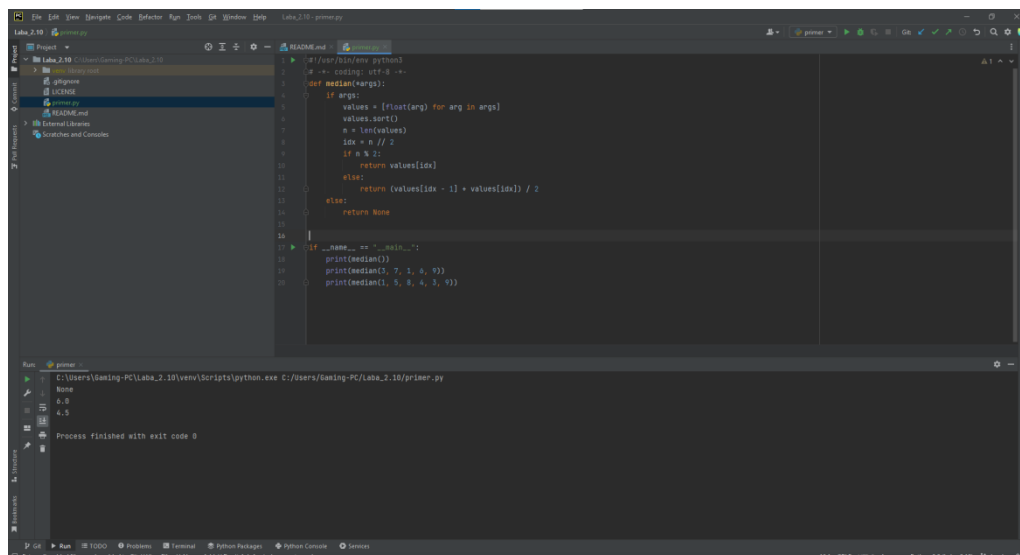
C:\Users\user>cd C:\Users\user\laba2-10

C:\Users\user\laba2-10>git flow init

which branch should be used for bringing forth production releases?
- main
Branch name for production releases: [main]
Branch name for "next release" development: [develop]

How to name your supporting branch prefixes?
Feature branches? [feature/]
Bugfix branches? [bugfix/]
Release branches? [release/]
Hotfix branches? [hotfix/]
Support branches? [support/]
Version tag prefix? []
Hooks and filters directory? [C:/Users/user/laba2-10/.git/hooks]
```

**Условие примера:** Разработать функцию для определения медианы значений аргументов функции. Если функции передается пустой список аргументов, то она должна возвращать значение None.



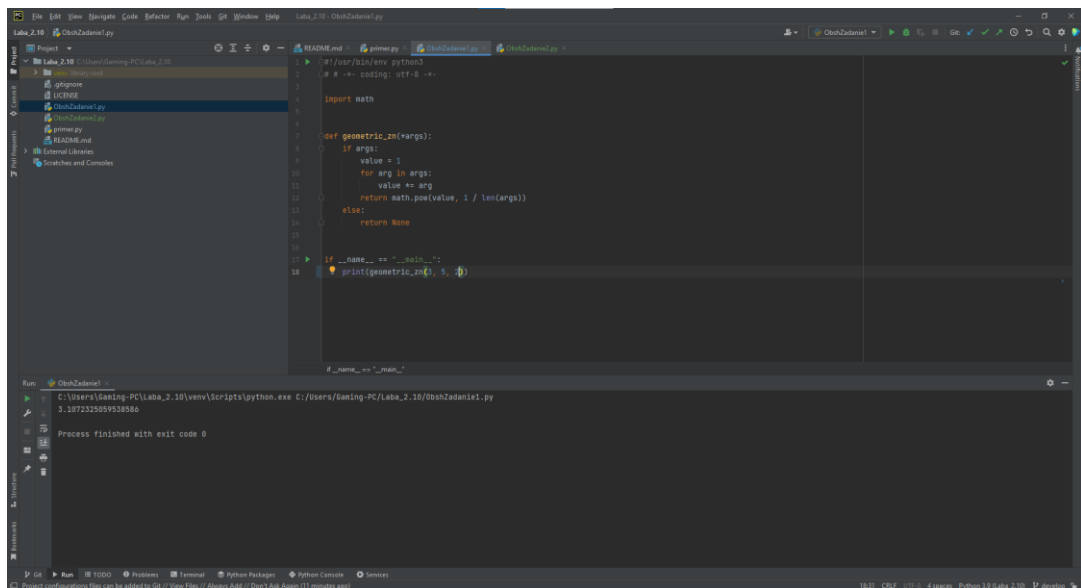
### Рисунок 3. Реализация примера

**Задание 2.** Создала новый файл под названием ObshZadanie1. Приступила к работе с общим заданием №1.

#### Условие примера:

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

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

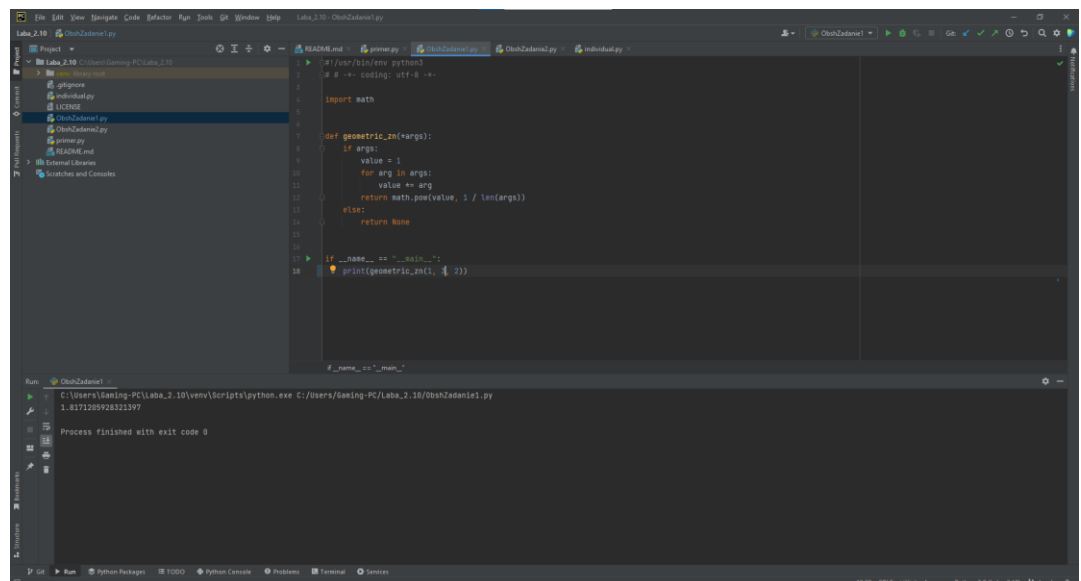


The screenshot shows a code editor with a file named 'ObshZadanie1.py'. The code defines a function 'geometric\_mn(args)' that calculates the geometric mean. It uses a loop to multiply all arguments and then takes the nth root. The function is tested with a single example: 'print(geometric\_mn(5, 5))'. The output in the console is '3.162277660168379'.

```
def geometric_mn(args):
    if args:
        value = 1
        for arg in args:
            value *= arg
        return math.pow(value, 1 / len(args))
    else:
        return None

if __name__ == "__main__":
    print(geometric_mn(5, 5))
```

### Рисунок 4. Реализация общего задания 1



The screenshot shows the same code editor with 'ObshZadanie1.py'. The function 'geometric\_mn' is the same as in the previous image. However, the test call is now 'print(geometric\_mn(1, 4, 2))'. The output in the console is '1.61120992822197'.

```
def geometric_mn(args):
    if args:
        value = 1
        for arg in args:
            value *= arg
        return math.pow(value, 1 / len(args))
    else:
        return None

if __name__ == "__main__":
    print(geometric_mn(1, 4, 2))
```

### Рисунок 5. Реализация общего задания 1 с другими данными

**Задание 3.** Создала новый файл под названием ObshZadanie2. Приступил к работе с общим заданием №2.

### Условие примера:

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

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

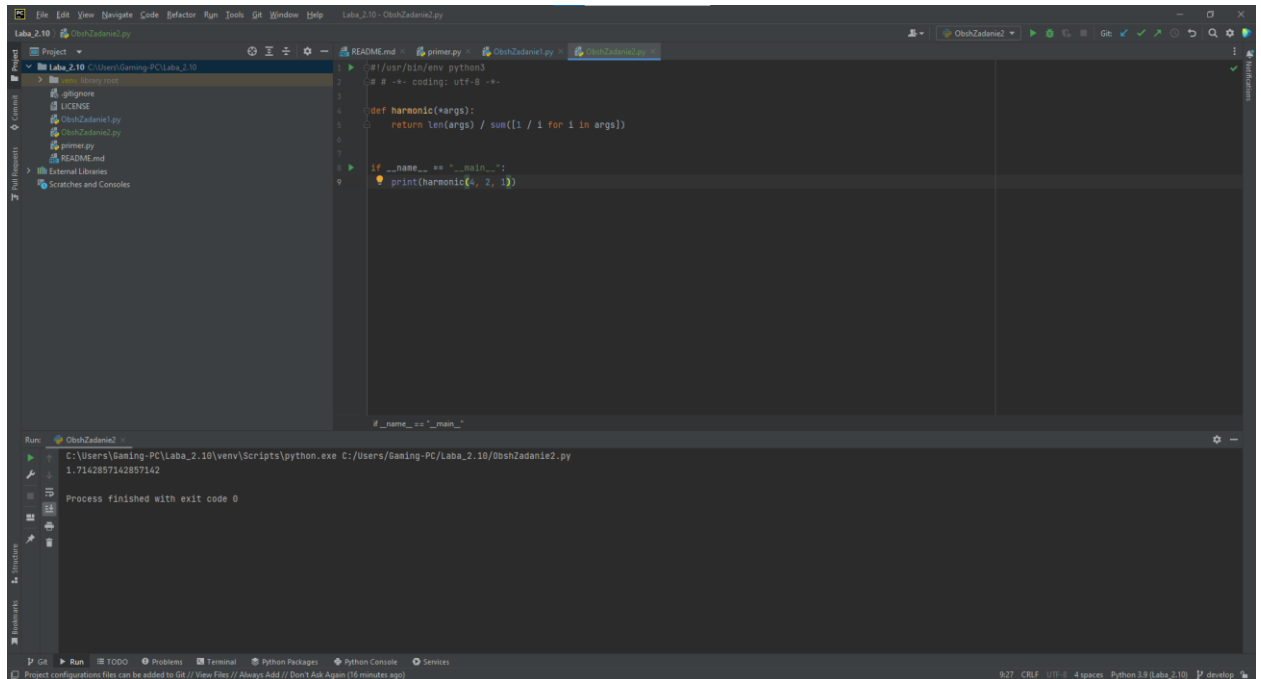


Рисунок 6. Реализация общего задания 2

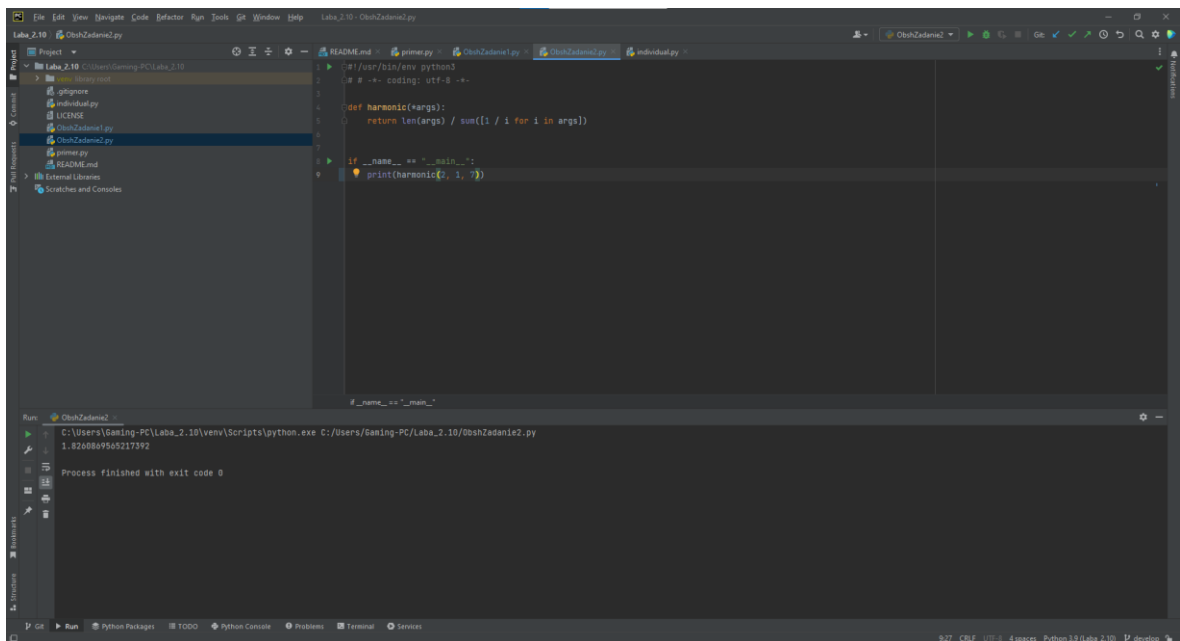


Рисунок 7. Реализация общего задания 2 с другими данными

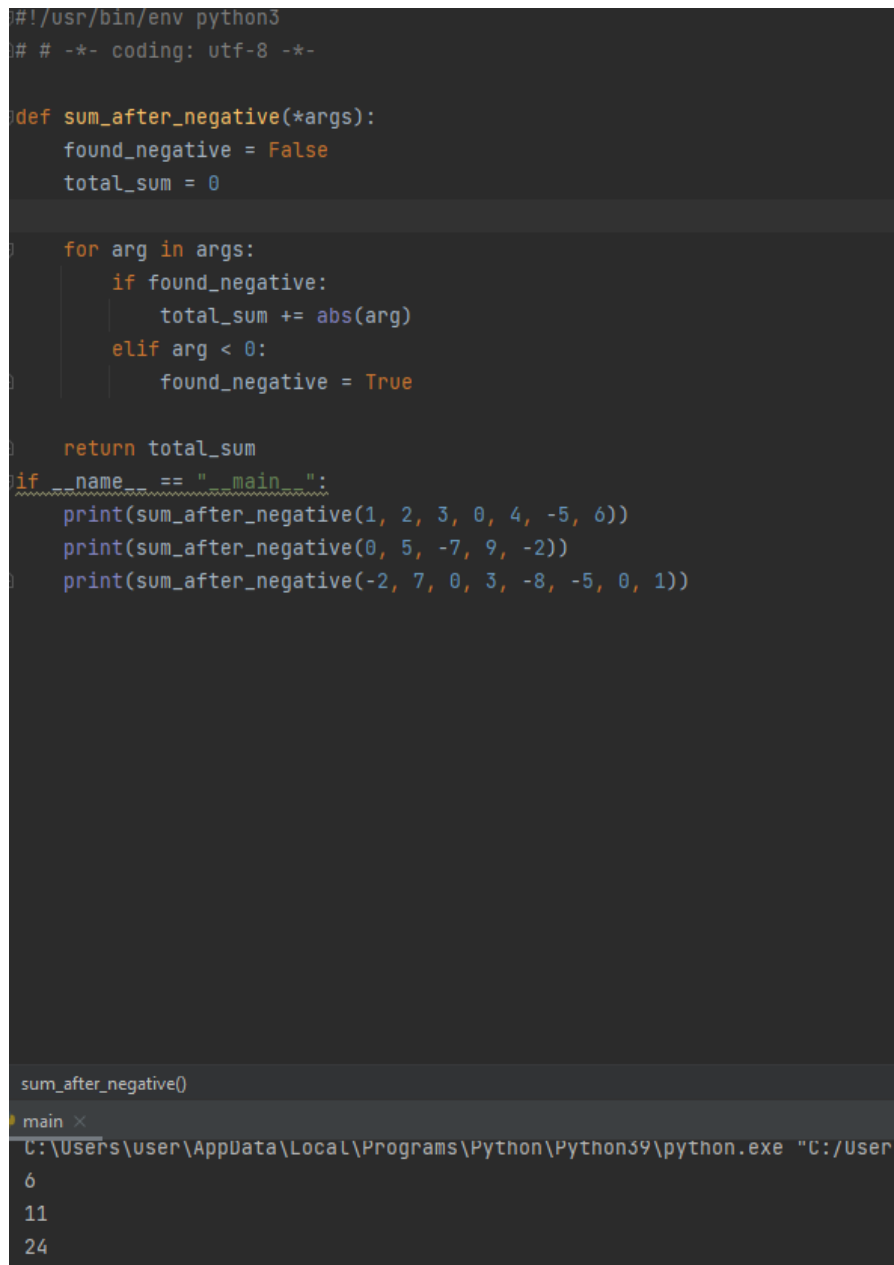
#### Задание 4.

#### Индивидуальное задание

#### Вариант 10

Создала новый файл под названием individual.py.

**Условие задания:** сумму модулей аргументов, расположенных после первого отрицательного аргумента.



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

def sum_after_negative(*args):
    found_negative = False
    total_sum = 0

    for arg in args:
        if found_negative:
            total_sum += abs(arg)
        elif arg < 0:
            found_negative = True

    return total_sum

if __name__ == "__main__":
    print(sum_after_negative(1, 2, 3, 0, 4, -5, 6))
    print(sum_after_negative(0, 5, -7, 9, -2))
    print(sum_after_negative(-2, 7, 0, 3, -8, -5, 0, 1))
```

The screenshot shows a terminal window with the Python script executed. The output displays the results of three function calls: 6, 11, and 24.

Рисунок 8. Программа индивидуального задания

**Задание 5.** Создала новый файл под названием idz2. Выбрала пример задания из списка индивидуальных заданий.

**Условие примера:** Сумму модулей аргументов, расположенных после первого аргумента, равного нулю.

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

def sum_modulo(*args):
    if args:
        zero = 0
        zero_index = args.index(zero)
        return sum(args[zero_index+1:])

if __name__ == "__main__":
    print(sum_modulo(1, 2, 3, 0, 4, -5, 6))
    print(sum_modulo(6, 5, -7, 2, 0, -2))
    print(sum_modulo(-7, 7, 0, 3, -8, -5, 0, 1))

if __name__ == "__main__":
    main
```

main x

C:\Users\user\AppData\Local\Programs\Python\Python39\python.exe "C:/Users/user/PycharmProjects/nb 2.1/main.py"

5  
-2  
-9

Рисунок 9. Программа выбранного задания

### Задание 6.

После выполнения работы на ветке develop, слила ее с веткой main и отправила изменения на удаленный сервер.

### Ответы на контрольные вопросы:

#### 1. Какие аргументы называются позиционными в Python?

Позиционные аргументы - это аргументы функции, передаваемые в порядке их указания при вызове функции. Значение каждого позиционного аргумента соответствует порядку его указания в списке аргументов функции.

Последовательность передачи аргументов важна и должна точно соответствовать порядку параметров функции.

## **2. Какие аргументы называются именованными в Python?**

Именованные аргументы - это аргументы функции, которые передаются в форме `name=value`. В отличие от позиционных аргументов, порядок их передачи не важен, поскольку они связываются с именами параметров функции, а не с их позицией.

## **3. Для чего используется оператор \*?**

Оператор `*` используется в Python для распаковки последовательностей (например, списка или кортежа) или коллекций (например, словаря) в отдельные элементы. Он также может использоваться в определении функций для передачи переменного числа аргументов (позиционных или именованных).

## **4. Каково назначение конструкций \*args и \*\*kwargs ?**

`*args` и `**kwargs` предназначены для передачи произвольного количества аргументов в функцию.

**Вывод:** приобрел навыки по работе с функциями с переменным числом параметров при написании программ с помощью языка программирования Python версии 3.x.