

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 «Программирование на Python»**  
**Вариант № 15**

Выполнил:  
Кенесбаев Хилол Куат улы  
2 курс, группа ИВТ-б-о-22-1,  
09.03.01 «Информатика и  
вычислительная техника»,  
направленность (профиль)  
«Программное обеспечение средств  
вычислительной техники и  
автоматизированных систем», очная  
форма обучения

---

(подпись)

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

---

(подпись)

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

**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.

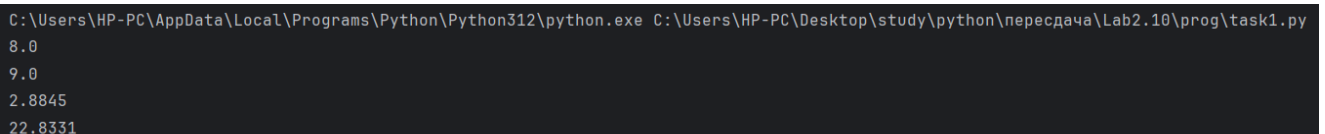
### **Порядок выполнения работы:**

1. Создал новый репозиторий, клонировал его, в нем создал ветку developer и перешел на нее.

2. Решил задачу 1: написать функцию, вычисляющую среднее геометрическое своих аргументов. Если функции передается пустой список аргументов, то она должна возвращать значение None.

### **Код программы:**

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
def geometric_mean(*args):"""
Поиск среднего геометрического аргументов"""
if args:
    values = [float(arg) for arg in args]
    product = 1.0
    for value in values:
        product *= value
    return round(pow(product,1/len(values)), 4)
else:
    return None
if __name__ == "__main__":
    print(geometric_mean(4, 8, 16))
    print(geometric_mean(3, 9, 27))
    print(geometric_mean(2, 3, 4))
    print(geometric_mean(31, 12, 32))
```



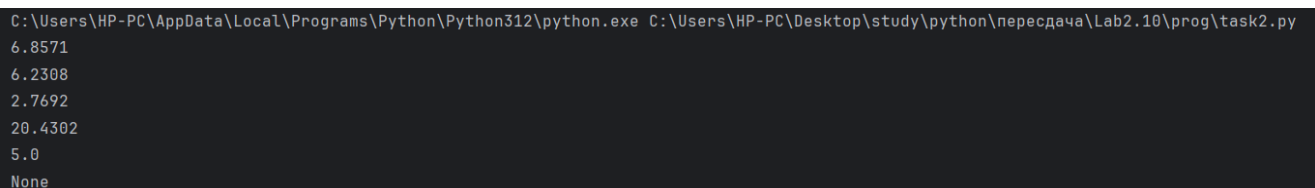
```
C:\Users\HP-PC\AppData\Local\Programs\Python\Python312\python.exe C:\Users\HP-PC\Desktop\study\python\непесдача\Lab2.10\prog\task1.py
8.0
9.0
2.8845
22.8331
```

Рисунок 1. Вывод программы task1

3. Решил задачу 2: написать функцию, вычисляющую среднее гармоническое своих аргументов. Если функции передается пустой список аргументов, то она должна возвращать значение None.

### Код программы:

```
#!/usr/bin/env
python3 # -*- coding:
utf-8 -*-
defharmonic_mean(*args):"""
Поиск среднего гармонического аргументов
"""
if args:
    values = [float(arg) for arg in args]
    product = 0
    for value in values:
        product += 1/value
    return round(len(values)/product, 4)
else:
    return None
if __name__ == "__main__":
    print(harmonic_mean(4, 8, 16))
    print(harmonic_mean(3, 9, 27))
    print(harmonic_mean(2, 3, 4))
    print(harmonic_mean(31, 12, 32))
    print(harmonic_mean(5))
    print(harmonic_mean())
```



```
C:\Users\HP-PC\AppData\Local\Programs\Python\Python312\python.exe C:\Users\HP-PC\Desktop\study\python\пересдача\Lab2.10\prog\task2.py
6.8571
6.2308
2.7692
20.4302
5.0
None
```

Рисунок 2. Вывод программы task2

4. Решил индивидуальное задание: Сумму модулей аргументов, расположенных после минимального по модулю аргумента.

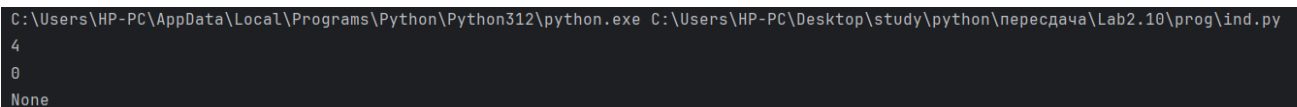
**Код программы:**

```
def sum_mod_after_min(*args):
    if not args: # Если список аргументов пуст,
        вернуть None
        return None

    min_abs_arg = min(args, key=abs) # Находим
    минимальный по модулю аргумент
    sum_mod = sum(abs(arg) for arg in
args[args.index(min_abs_arg) + 1:])

    return sum_mod

# Примеры использования
print(sum_mod_after_min(3, -5, 2, 4)) # Сумма
модулей аргументов после -5:  $|2| + |4| = 6$ 
print(sum_mod_after_min(-10, 5, -3, 7, 1)) #
Сумма модулей аргументов после -3:  $|7| + |1| = 8$ 
print(sum_mod_after_min()) # Пустой список
аргументов, возвращаем None
```



```
C:\Users\HP-PC\AppData\Local\Programs\Python\Python312\python.exe C:\Users\HP-PC\Desktop\study\python\пересдача\Lab2.10\prog\ind.py
6
8
None
```

Рисунок 3. Вывод программы ind.py

5. Выполнил задание: Самостоятельно подберите или придумайте задачу с переменным числом именованных аргументов. Приведите решение этой задачи.

**Код программы:**

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
def people(**kwargs):
    if kwargs:
        for name, age in kwargs.items():
            if age > 18:
                print(f'{name}: {age}')
            else:
                return None
    if __name__ == "__main__":
        people(Дима=6, Андрей=19, Кирилл=35)
```

```
C:\Users\HP-PC\AppData\Local\Programs\Python\Python312\python.exe C:\Users\HP-PC\Desktop\study\python\передача\Lab2.10\prog\my_ind.py
name: Хилл
age: 21

city: Ставрополь
country: Россия

hobby: Баскетбол
job: Студент
salary: 14000
```

Рисунок 4. Вывод программы my\_ind.py

### Ответы на контрольные вопросы:

1. Какие аргументы называются позиционными в Python?

В Python аргументы называются позиционными, если они передаются функции в том же порядке, в котором они определены в функции.

В функцию также можно передать переменное количество позиционных аргументов. Это делается с помощью оператора `*` перед именем аргумента в определении функции.

2. Какие аргументы называются именованными в Python?

В Python аргументы называются именованными, если они передаются функции с указанием имени аргумента, за которым следует значение аргумента.

В функцию также можно передать переменное количество именованных аргументов. Это делается с помощью оператора `**` перед именем аргумента в определении функции.

3. Для чего используется оператор `*`?

Оператор `*` чаще всего ассоциируется у людей с операцией умножения, но в Python он имеет и другой смысл. Этот оператор позволяет «распаковывать» объекты, внутри которых хранятся некие элементы.

4. Каково назначение конструкций `*args` и `**kwargs` ?

Итак, мы знаем о том, что оператор «звёздочка» в Python способен «вытаскивать» из объектов составляющие их элементы. Знаем мы и о том, что существует два вида параметров функций. А именно, `*args` – это сокращение от «arguments» (аргументы), а `**kwargs` – сокращение от «keyword arguments»

(именованные аргументы).

Каждая из этих конструкций используется для распаковки аргументов соответствующего типа, позволяя вызывать функции со списком аргументов переменной длины.

**Вывод:** в результате выполнения работы были приобретены навыки по работе с функциями с переменным числом параметров при написании программ с помощью языка программирования Python версии 3.x.