

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 \_\_\_\_**

Выполнил:  
Иващенко Олег Андреевич  
2 курс, группа ИВТ-б-о-22-1,  
09.03.02 «Информационные и  
вычислительные машины»,  
направленность (профиль)  
«Программное обеспечение средств  
вычислительной техники и  
автоматизированных систем»

---

(подпись)

Руководитель практики:  
Воронкин Роман Александрович,  
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.

### Порядок выполнения работы

Таблица 1 – Код программы example.py

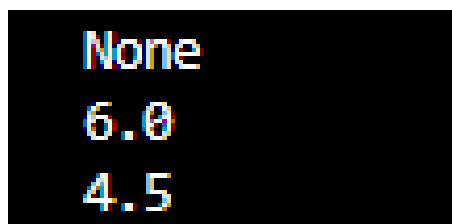
```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

def median(*args):
    if args:
        values = [float(arg) for arg in args]
        values.sort()

        n = len(values)
        idx = n // 2
        if n % 2:
            return values[idx]
        else:
            return (values[idx - 1] + values[idx]) / 2

    else:
        return None

if __name__ == "__main__":
    print(median())
    print(median(3, 7, 1, 6, 9))
    print(median(1, 5, 8, 4, 3, 9))
```



```
None
6.0
4.5
```

Рисунок 1 – Вывод программы example.py

Таблица 2 – Код программы general\_1.py

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

def average_geo(*args):
    """
    Функция для расчёта среднего геометрического значения
    введённых чисел. Если числа отсутствуют, то функция
    вернёт None
    """

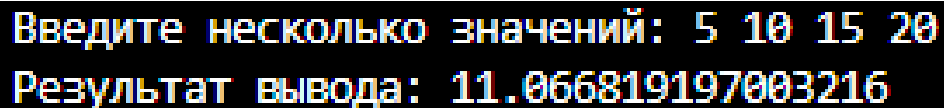
    if args:
        list = [int(arg) for arg in args]
        result = 1

        for value in list:
            result *= value

        return result**(1 / len(list))

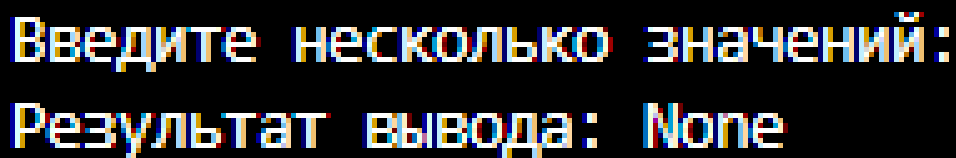
    else:
        return None

if __name__ == "__main__":
    arg_list = input("Введите несколько значений: ").split()
    print(f"Результат вывода: {average_geo(*arg_list)}")
```



Введите несколько значений: 5 10 15 20  
Результат вывода: 11.066819197003216

Рисунок 2.1 – Результат вывода программы general\_1.py при вводе значений



Введите несколько значений:  
Результат вывода: None

Рисунок 2.2 – Результат вывода программы general\_1.py при отсутствии введённых значений

Таблица 3 – Код программы general\_2.py

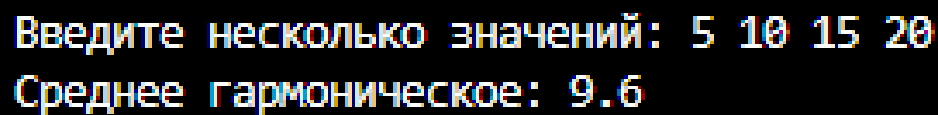
```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
```

```
def average_harmonic(*args):
    """
    Функция вычисления среднего гармонического значения
    введённых аргументов. Выводит None, если аргументов нет
    """

    if args:
        harmonic = sum(1 / int(arg) for arg in args)
        return len(args) / harmonic

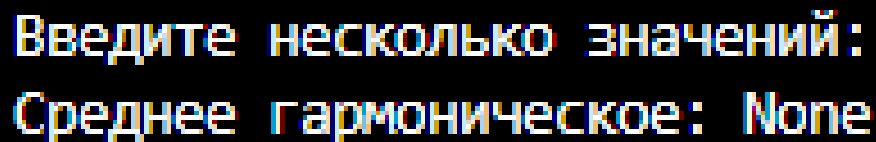
    else:
        return None

if __name__ == "__main__":
    arg_list = input("Введите несколько значений: ").split()
    print(f"Среднее гармоническое: {average_harmonic(*arg_list)}")
```



```
Введите несколько значений: 5 10 15 20
Среднее гармоническое: 9.6
```

Рисунок 3.1 – Результат выполнения программы general\_2.py при вводе значений



```
Введите несколько значений:
Среднее гармоническое: None
```

Рисунок 3.2 – Результат выполнения программы general\_2.py при отсутствии ввода значений

Индивидуальное задание. Написать функцию, принимающую произвольное количество аргументов, и возвращающая требуемое значение. Если функции передаётся пустой список аргументов, то она должна возвращать значение None. В процессе решения не использовать преобразования конструкции \*args в список или иную структуру данных.

Сумму аргументов, расположенных после минимального аргумента.

Таблица 4 – Код программы individual.py

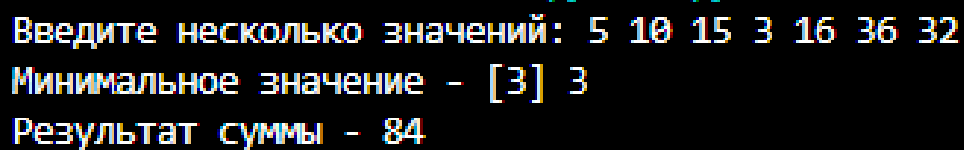
```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

def args_sum(*args):
    """
    Функция находит минимальный элемент в ряде значений
    и считает сумму элементов, находящихся после минимального.
    В случае, если значений нет, функций возвращает None
    """

    if args:
        min_value = min(args)
        min_index = args.index(min_value)
        result = sum(args[min_index + 1:])
        return min_index, min_value, result

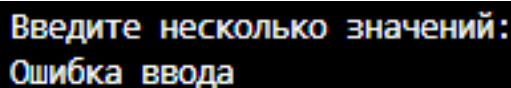
    else:
        return None

if __name__ == "__main__":
    arg_list = tuple(map(int, input("Введите несколько значений: ").split()))
    result = args_sum(*arg_list)
    if result is None:
        print("Ошибка ввода")
    else:
        print(f"Минимальное значение - [{result[0]}] {result[1]}")
        print(f"Результат суммы - {result[2]}")
```



```
Введите несколько значений: 5 10 15 3 16 36 32
Минимальное значение - [3] 3
Результат суммы - 84
```

Рисунок 4.1 – Результат выполнения программы individual.py при вводе значений



```
Введите несколько значений:
Ошибка ввода
```

Рисунок 4.2 – Результат выполнения программы individual.py при отсутствии ввода значений

1. Какие аргументы называются позиционными в Python?

Позиционные аргументы – это аргументы, передаваемые в функцию в порядке их определения в сигнатуре функции. Значения этих аргументов соответствуют позициям, на которых они указаны при вызове функции.

2. Какие аргументы называются именованными в Python?

Именованные аргументы – это аргументы, передаваемые в функцию с явным указанием их имени. При использовании именованных аргументов порядок передачи не важен, поскольку каждый аргумент определяется по его имени. Например:

```
def example_function(arg1, arg2):  
    ...  
  
example_function(arg2 = 5, arg1 = 10)
```

3. Для чего используется оператор \*?

Оператор «\*» используется для распаковки последовательности (списка или кортежа) в аргументы функции или для определения переменного числа аргументов в сигнатуре функции. Например:

```
def example_function(arg1, arg2, arg3):  
    ...  
  
arg_list = [1, 2, 3]  
example_function(*arg_list)
```

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

«\*args» используется для передачи переменного числа позиционных аргументов в функцию. «args» - это кортеж, который содержит все переданные позиционные аргументы. Пример:

```
def example_function(*args):  
    ....  
  
example_function(1, 2, 3)
```

«\*\*kwargs» используется для передачи переменного числа именованных аргументов в функцию. «kwargs» - это словарь, который содержит все переданные именованные аргументы. Пример:

```
def example_function(**kwargs):  
    ...  
  
example_function(arg1 = 1, arg2 = 2, arg3 = 3)
```

**Выводы:** В процессе выполнения лабораторной работы были приобретены навыки по работе с функциями с переменным числом параметров при написании программ с помощью языка программирования Python версии 3.x, были написаны 4 программы: пример работы с такими функциями, 2 общих задания и индивидуальное задание.