

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное
образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Кафедра инфокоммуникаций

Отчет по лабораторной работе № 12
Функции с переменным числом параметров в языке Python
по дисциплине «Технологии программирования и алгоритмизации»

Выполнил студент группы ИВТ-б-о-20-1

Злыгостев И.С. « » _____ 20__ г.

Подпись студента _____

Работа защищена « » _____ 20__ г.

Проверил Воронкин Р.А. _____

(подпись)

Ставрополь 2021

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

Ход работы

1. После изучения теоретической части методических указаний, приступил к выполнению общих заданий.

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

def composition(*arg):
    a = 1
    for i in arg:
        a *= i
    a = a ** (1/len(arg))
    return a

if __name__ == '__main__':
    try:
        print('Введите список аргументов через запятую: ')
        arg = list(map(float, input().split(',')))
        print("Среднее геометрическое элементов: ",
              composition(*arg))
    except ValueError:
        print(None)
```

Рисунок 12.1 – Код первого общего задания

```
Введите список аргументов через запятую: 4,2,1,5,6
Среднее геометрическое: 2.9925557394776896
```

Рисунок 12.2 – Первый результат выполнения кода

```
Введите список аргументов через запятую:  
None
```

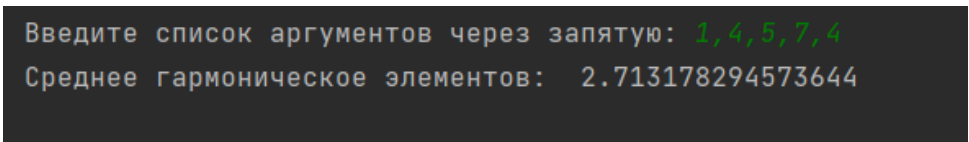
Рисунок 12.3 – Второй результат выполнения кода

```
Введите список аргументов через запятую: 5,1,4,9,0,23  
Среднее геометрическое: 0.0
```

Рисунок 12.4 – Третий результат выполнения кода

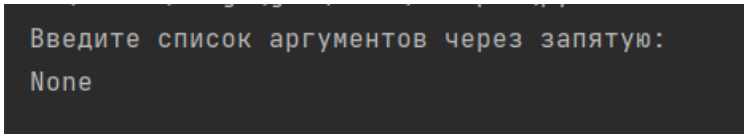
```
#!/usr/bin/env python3  
# -*- coding: utf-8 -*-  
  
def harmonic_mean_of_numbers(*arg):  
    summ = 0  
    for i in arg:  
        if i == 0:  
            return 'Невозможно посчитать, т.к. в списке есть 0'  
        else:  
            summ += 1/float(i)  
    harmonic = 1/(1/len(arg)*summ)  
    return harmonic  
  
if __name__ == '__main__':  
    try:  
        print('Введите список аргументов через запятую: ')  
        arg = list(map(float, input().split(',')))  
        print("Среднее гармоническое элементов: ",  
              harmonic_mean_of_numbers(*arg))  
    except ValueError:  
        print(None)
```

Рисунок 12.5 – Код второго общего задания



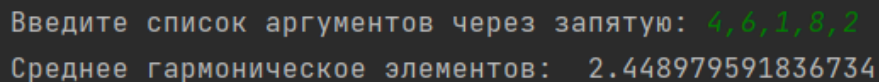
```
Введите список аргументов через запятую: 1,4,5,7,4
Среднее гармоническое элементов: 2.713178294573644
```

Рисунок 12.6 – Первый результат выполнения кода



```
Введите список аргументов через запятую:
None
```

Рисунок 12.7 – Второй результат выполнения кода



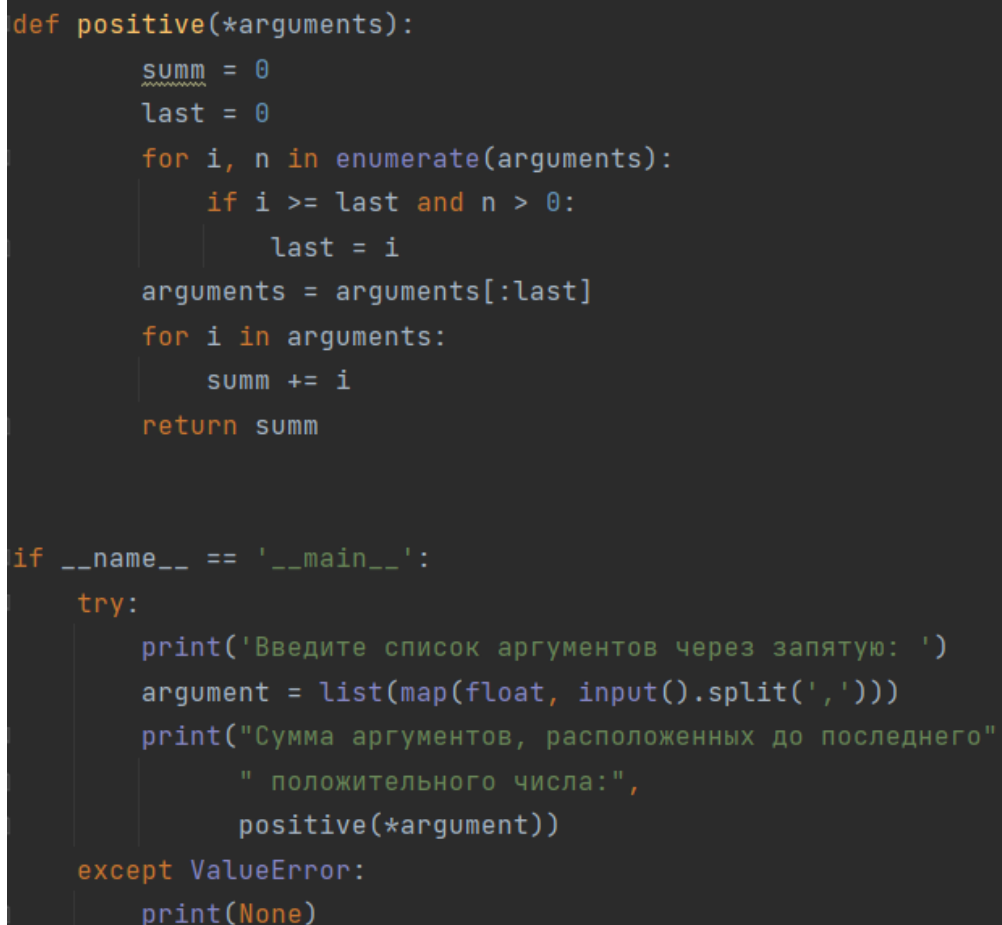
```
Введите список аргументов через запятую: 4,6,1,8,2
Среднее гармоническое элементов: 2.448979591836734
```

Рисунок 12.8 – Третий результат выполнения кода

Индивидуальное задание

Вариант 5

2. После выполнения общих заданий приступил к работе с индивидуальным.



```
def positive(*arguments):
    summ = 0
    last = 0
    for i, n in enumerate(arguments):
        if i >= last and n > 0:
            last = i
    arguments = arguments[:last]
    for i in arguments:
        summ += i
    return summ

if __name__ == '__main__':
    try:
        print('Введите список аргументов через запятую: ')
        argument = list(map(float, input().split(',')))
        print("Сумма аргументов, расположенных до последнего"
              " положительного числа:",
              positive(*argument))
    except ValueError:
        print(None)
```

Рисунок 12.9 – Код индивидуального задания

```
Введите список аргументов через запятую: 4,1,3,0,5,1,-6,4
Сумма аргументов, расположенных до последнего положительного числа: 8
```

Рисунок 12.10 – Результат выполнения кода

3. Затем придумал собственное задание: “С клавиатуры вводится команда: если exit – завершить работу программы, если add – добавить параметры о каком-либо событии или человеке (сперва вводится параметр, например, “Имя”, затем вводится значение, например, “Иван”), если list – показать все введённые ранее параметры” (Пример: должен вывести “Имя - Иван”). И приложил одно из решений к этому заданию.

```
def checklist(**argument):
    for key, value in argument.items():
        print("{} - {}".format(key, value))

def add(argument):
    message = input('Введите параметр: ')
    argument[message] = input("Введите значение: ")
    return argument

def main():
    argument = {}
    while True:
        mess = input('Введите команду: ').lower()
        if mess == "exit":
            exit()
        elif mess == "add":
            add(argument)
        elif mess == 'list':
            checklist(**argument)
        else:
            print("Неизвестная команда")

if __name__ == '__main__':
    main()
```

Рисунок 12.11 – Код придуманного задания

```
Введите команду: add
Введите параметр: Имя
Введите значение: Иван
Введите команду: add
Введите параметр: Возраст
Введите значение: 19
Введите команду: list
Имя - Иван
Возраст - 19
Введите команду: exit

Process finished with exit code 0
```

Рисунок 12.12 – Результат выполнения кода

Контрольные вопросы

1. Какие аргументы называются позиционными в Python?

Позиционные аргументы обрабатываются слева направо. То есть оказывается, что позиция аргумента, переданного функции, находится в прямом соответствии с позицией параметра, использованного в заголовке функции при её объявлении.

2. Какие аргументы называются именованными в Python?

Именованные аргументы передают функциям с указанием имён этих аргументов, соответствующих тем именам, которые им назначены при объявлении функции.

3. Для чего используется оператор *?

Этот оператор позволяет «распаковывать» объекты, внутри которых хранятся некие элементы.

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

При применении конструкции *args в параметр args попадают позиционные аргументы, представляемые в виде кортежа. При применении **kwargs в kwargs попадают именованные аргументы, представленные в виде словаря.

Вывод: в ходе выполнения лабораторной работы приобрел навыки по работе с функциями с переменным числом параметров при написании программ с помощью языка программирования Python версии 3.x.