

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

**Кафедра инфокоммуникаций**

**Отчет по лабораторной работе №2.10  
Функции с переменным числом параметров в Python  
по дисциплине «Технологии программирования»**

Выполнил студент группы ИВТ-б-о-20-1

Ищенко Т.С. « » \_\_\_\_\_ 20\_\_ г.

Подпись студента \_\_\_\_\_

Работа защищена « » \_\_\_\_\_ 20\_\_ г.

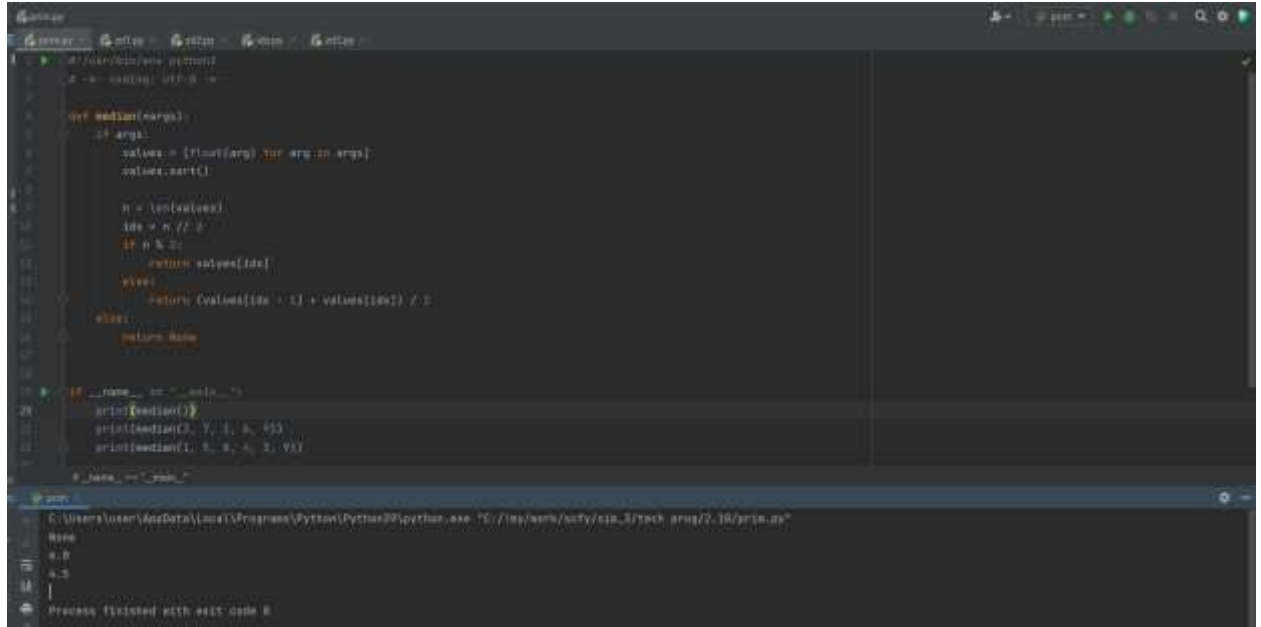
Проверила Воронкин Р.А. \_\_\_\_\_

(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. Произвёл выполнение примера



```
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, 4, 6, 9))
    print(median(1, 5, 8, 4, 2, 9))

if __name__ == '__main__':
```

The screenshot shows a Python IDE with a dark theme. The main editor displays a function `median` that takes a list of arguments, sorts them, and returns the median. Below the function, there are three calls to `print(median())` with different argument lists. The bottom panel shows the execution output: `None`, `6.5`, and `6.5`. The status bar at the bottom indicates "Process finished with exit code 0".

Рисунок 1 – Результат выполнения примера

### 2. Выполнил первое задание

```
"""
Решить поставленную задачу: написать функцию,
вычисляющую среднее геометрическое
своих аргументов a1, a2, ..., an.
Если функции передается пустой список аргументов,
то она должна возвращать значение None
"""

def func(*arg):
    if arg:
        g = 1.0
        for i in arg:
            g *= i
        g = g ** (1 / len(arg))
        return g
    else:
        return None

if __name__ == '__main__':
    mass = list(map(float, input("Введите массив из чисел: ").split()))
    print(func(*mass))

func() / else

zd1 x
C:\Users\user\AppData\Local\Programs\Python\Python39\python.exe "C:/!my/work/scfy/sim_3/tech prog/2.10/zd1.py"
Введите массив из чисел: 1 1 1
1.8171205928321397
```

Рисунок 2 – Результат выполнения первого задания

3. Выполнил второе задание

```
7      вычисляющую среднее гармоническое
8      своих аргументов a1, a2, ... an
9      Если функции передается пустой список аргументов,
10     то она должна возвращать значение None
11     """
12
13
14     def func(*arg):
15         summ = 0
16         if arg:
17             for i in arg:
18                 if i == 0:
19                     return None
20                 else:
21                     summ += 1 / float(i)
22             otvet = 1 / (1 / len(arg) * summ)
23             return otvet
24         else:
25             return None
26
27
28     if __name__ == '__main__':
29         mass = list(map(float, input("Введите массив из чисел: ").split()))
30         print(func(*mass))
31
```

func() > if arg

Python Shell (zd2)

C:\Users\user\AppData\Local\Programs\Python\Python39\python.exe "C:/!my

Введите массив из чисел: 1 2 3

1.6363636363636365

Рисунок 3 – Результат выполнения второго задания

4. Выполнил третье задание

```
2 # -*- coding: utf-8 -*-
3
4 """
5 Есть список студентов, в котором указано:
6 1. Имя студента
7 2. Абсолютная сумма баллов у студента
8 Найти студента с наибольшим баллом
9 """
10
11
12 def func(**studs):
13     count = 0
14     for name, bal in studs.items():
15         if bal > count:
16             count = bal
17     return f"Самый высокий балл у студента по имени: {name}"
18
19
20 if __name__ == '__main__':
21     print(func(
22         (Ilya=11,
23         Sergey=12,
24         Andrey=15,
25         Vasya=14))
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
```

Самый высокий балл у студента по имени: Илья

Рисунок 4 – Результат выполнения третьего задания

5. Выполнил индивидуальное задание

```

'''
Сумму аргументов, расположенных между первым
и вторым отрицательными аргументами.
'''

def func(*mass):
    summ = 0
    if mass:
        for i, x in enumerate(mass):
            if x < 0:
                for count in mass[i+1:]:
                    if count < 0:
                        break
                    else:
                        summ += count
                return summ
            else:
                return None

if __name__ == '__main__':
    mass = list(map(float, input("Введите массив из чисел: ").split()))
    print(func(*mass))

```

Рисунок 6 – Результат выполнения индивидуального задания

## 6. Провел проверку на PEP8 всех заданий

```

(base) C:\Users\user>conda activate tools

(tools) C:\Users\user>cd C:\!my\work\scfy\sim_3\tech prog\2.10

(tools) C:\!my\work\scfy\sim_3\tech prog\2.10>flake8
\zd3.py:25:21: W292 no newline at end of file

(tools) C:\!my\work\scfy\sim_3\tech prog\2.10>flake8

(tools) C:\!my\work\scfy\sim_3\tech prog\2.10>

```

Рисунок 7 – Результат проверки на соответствие PEP8

Контрольные вопросы:

1. Какие аргументы называются позиционными в Python? - При вызове функции аргументы можно передавать как позиционные – передаются в том же порядке, в котором они определены при создании функции. То есть, порядок передачи аргументов определяет, какое значение получит каждый аргумент.

2. Какие аргументы называются именованными в Python? - Аргументы, передаваемые с именами, называются именованными. При вызове функции можно использовать имена параметров из ее определения. Благодаря `**kwargs` создаётся словарь, в котором содержатся именованные аргументы, переданные функции при её вызове.

3. Для чего используется оператор `*`? - Этот оператор позволяет «распаковывать» объекты, внутри которых хранятся некие элементы. `a = [1, 2, 3]` `b = [*a, 4, 5, 6]` `print(b)` # `[1, 2, 3, 4, 5, 6]` Тут берётся содержимое списка `a`, распаковывается, и помещается в список `b`.

4. Каково назначение конструкций `*args` и `**kwargs`? `*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: по итогу выполнения лабораторной работы были приобретены навыки по работе с функциями с переменным числом параметров при написании программ с помощью языка программирования Python.