

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ
АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО
ОБРАЗОВАНИЯ «СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»
ИНСТИТУТ ЦИФРОВОГО РАЗВИТИЯ**

**Отчет по лабораторной работе №13
«Функции с переменным числом параметров в Python»**

по дисциплине «Основы программной инженерии»

Выполнила:

Первых Дарья Александровна,
2 курс, группа ПИЖ-б-о-20-1,

Проверил:

Доцент кафедры инфокоммуникаций,
Воронкин Р.А.

Ставрополь, 2021 г.

ВЫПОЛНЕНИЕ

```
1 def print_these(a, b, c):
2     print(a, "is stored in a")
3     print(b, "is stored in b")
4     print(c, "is stored in c")
5
6 print_these(1, 2, 3)
7
```

print_these()

main x

C:\Users\podar\study\anaconda\envs\13

1 is stored in a
2 is stored in b
3 is stored in c

Рисунок 1 – Пример простейшей функции сопоставления позиции аргументов и параметров

```
1 def print_these(a, b, c):
2     print(a, "is stored in a")
3     print(b, "is stored in b")
4     print(c, "is stored in c")
5
6 print_these(1, 2)
```

main x

C:\Users\podar\study\anaconda\envs\13LR\python.exe "C:/Users/podar/st

Traceback (most recent call last):

File "C:\Users\podar\study\PyCharm Community Edition 2021.2.3\13LR\

print_these(1, 2)

TypeError: print_these() missing 1 required positional argument: 'c'

Рисунок 2 – Результат неверного вывода команд

```
1 def print_these(a, b, c=None):  
2     print(a, "is stored in a")  
3     print(b, "is stored in b")  
4     print(c, "is stored in c")  
5  
6     print_these(1, 2)  
  
print_these()
```

main ×

C:\Users\podar\study\anaconda\envs\

1 is stored in a
2 is stored in b
None is stored in c

Рисунок 3 – Применение опциональных параметров

```
1 def print_these(a=None, b=None, c=None):  
2     print(a, "is stored in a")  
3     print(b, "is stored in b")  
4     print(c, "is stored in c")  
5  
6     print_these(1, 2)  
  
print_these()
```

main ×

C:\Users\podar\study\anaconda\envs\13LR\python

1 is stored in a
2 is stored in b
None is stored in c

Рисунок 4 – Установление трех опциональных параметров

```
a = [1, 2, 3]  
b = [*a, 4, 5, 6]  
print(b) # [1, 2, 3, 4, 5, 6]
```

main ×

C:\Users\podar\study\anaconda\envs\

[1, 2, 3, 4, 5, 6]

Рисунок 5 – Применение оператора «звездочка»

```
1 def print_scores(student, *scores):
2     print(f"Student Name: {student}")
3     for score in scores:
4         print(score)
5
6 print_scores("Jonathan", 100, 95, 88, 92, 99)
```

main x

C:\Users\podar\study\anaconda\envs\13LR\python.exe

Student Name: Jonathan

100

95

88

92

99

```
1 def print_pet_names(owner, **pets):
2     print(f"Owner Name: {owner}")
3     for pet_name in pets.items():
4         print(f"{pet}: {name}")
5
6 print_pet_names(
7     "Jonathan",
8     dog="Brock", fish=["Larry", "Curly", "Moe"],
9     turtle="Shelldon"
10 )
```

main x

C:\Users\podar\study\anaconda\envs\13LR\python.exe "C

Owner Name: Jonathan

dog: Brock

fish: ['Larry', 'Curly', 'Moe']

turtle: Shelldon

Рисунок 6 – Использование функций *args и *kwargs

```
1  ▶  #!/usr/bin/env python3
2      # -*- coding: utf-8 -*-
3
4      def median(*args):
5          if args:
6              values = [float(arg) for arg in args]
7              values.sort()
8
9              n = len(values)
10             idx = n // 2
11             if n % 2:
12                 return values[idx]
13             else:
14
15
16  if __name__ == "__main__":
17
18
19
20
21
22
23
24
25
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
```

main ×

C:\Users\podar\study\anaconda\envs\13LR\python.exe

None

6.0

4.5

Рисунок 8 – Пример

```
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  def sr_geom(*args):
5      if args:
6          summa = 0
7          for arg in args:
8              summa = summa + arg
9          n = len(args)
10         return summa / n
11     else:
12         return None
13
14  if __name__ == "__main__":
15      arguments = list(map(int, input('Введите аргументы: ').split()))
16      print('Среднее геометрическое значение аргументов: ', sr_geom(*arguments))
17
18  if __name__ == "__main__":
19
20
21
22
23
24
25
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
```

main ×

C:\Users\podar\study\anaconda\envs\13LR\python.exe "C:/Users/podar/s

Введите аргументы: 5 12 17 27 5 4 19

Среднее геометрическое значение аргументов: 12.714285714285714

Рисунок 9 – Решение 1 задания

```
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4 def sr_garmonic(*args):
5     if args:
6         summa = float(0)
7         for arg in args:
8             summa = summa + (1 / arg)
9         n = len(args)
10        return n / summa
11    else:
12        return None
13
14 if __name__ == "__main__":
15     arguments = list(map(int, input('Введите аргументы: ').split()))
16     print('Среднее гармоническое значение аргументов: ', sr_garmonic(
17
18 if __name__ == "__main__"
```

main x

C:\Users\podar\study\anaconda\envs\13LR\python.exe "C:/Users/podar/study
Введите аргументы: 5 6 7 12 27 17 19 7 2
Среднее гармоническое значение аргументов: 6.501920376702704

Рисунок 10 – Решение 2 задания

```
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4 def knigi(author, **books):
5     print(f"Автор: {author}")
6     for books, name in books.items():
7         print(f"{books}: {name}")
8
9
10 if __name__ == "__main__":
11     knigi(
12         "Джо Диспенза",
13         книга="Сила подсознания",
```

main x

C:\Users\podar\study\anaconda\envs\13LR\pytho
Автор: Джо Диспенза
книга: Сила подсознания
книга_1: Сам себе плацебо
книга_2: Сверхестественный разум
книга_3: Развивай свой мозг

Рисунок 11 – Решение 3 задания

Индивидуальное задание. Напишите функцию, принимающую произвольное количество аргументов, и возвращающую требуемое значение. Если функции передается пустой список аргументов, то она должна возвращать значение None . Номер варианта определяется по согласованию с преподавателем.

В процессе решения не использовать преобразования конструкции *args в список или иную структуру данных.

Сумму целых частей аргументов, расположенных после последнего отрицательного аргумента.

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

```
def summ(*args):
    if args:
        summa = 0
        args = reversed(args)
        for argument in args:
            if argument > 0:
                summa += argument
            else:
                break
        return summa
    else:
        return None

if __name__ == "__main__":
    arguments = list(map(float, input('Введите числа: ').split()))
    print('Сумма чисел после последнего отрицательного: ',
          summ(*arguments))
```

```
2  # -*- coding: utf-8 -*-
3
4
5  def summ(*args):
6      if args:
7          summa = 0
8          args = reversed(args)
9          for argument in args:
10             if argument > 0:
11                 summa += argument
12             else:
13                 break
14             return summa
15         else:
16             return None
17
```

summa() > if args > for argument in args > if argument > 0

main x

C:\Users\podar\study\anaconda\envs\13LR\python.exe

Введите числа: 4 5 -1 5 6 -5 8.1 5.2 3

Сумма чисел после последнего отрицательного: 16

Рисунок 12 – Результат решения идз

Вопросы

1. Какие аргументы называются позиционными в Python?

Это аргументы, передаваемые в вызов в определённой последовательности (на определённых позициях), без указания их имён. Элементы объектов, поддерживающих итерирование, могут использоваться в качестве позиционных аргументов, если их распаковать при помощи *.

2. Какие аргументы называются именованными в Python?

Это аргументы, передаваемые в вызов при помощи имени (идентификатора), либо словаря с его распаковкой при помощи **.

3. Для чего используется оператор *?

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

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

*args используется для передачи произвольного числа именованных аргументов функции.

**kwargs позволяет передавать произвольное число именованных аргументов в функцию.