

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 «Основы программной инженерии»**

Выполнил:  
Баратов Семен Григорьевич  
2 курс, группа ПИЖ-б-о-22-1,  
09.03.04 «Программная инженерия»,  
направленность (профиль) «Разработка  
и сопровождение программного  
обеспечения», очная форма обучения

---

(подпись)

Преподаватель:  
Воронкин Р.А., канд. тех. наук, доцент,  
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.

### Результаты выполнения

1. Создали репозиторий с лицензией MIT, добавили в .gitignore необходимые правила для работы с IDE PyCharm, клонировали репозиторий, организовали репозиторий в соответствии с моделью git-flow.

```
Last login: Tue Oct 24 20:29:02 on ttys000
itssyoma@MacBook-Air-Sema Основы программной инженерии % git clone https://github.com/itssyoma/megarepo_21.git
Cloning into 'megarepo_21'...
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (4/4), done.
itssyoma@MacBook-Air-Sema Основы программной инженерии % cd megarepo_21
itssyoma@MacBook-Air-Sema megarepo_21 % git checkout -b develop
Switched to a new branch 'develop'
itssyoma@MacBook-Air-Sema megarepo_21 % git branch release
itssyoma@MacBook-Air-Sema megarepo_21 % git branch develop
fatal: a branch named 'develop' already exists
itssyoma@MacBook-Air-Sema megarepo_21 % git branch hotfix
itssyoma@MacBook-Air-Sema megarepo_21 % git branch feature
itssyoma@MacBook-Air-Sema megarepo_21 %
```

Рисунок 1 – Работа с репозиторием в командной строке.

2. Проработали пример.

```
example.py > ...
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4
5  def median(*args):
6      if args:
7          values = [float(arg) for arg in args]
8          values.sort()
9          n = len(values)
10         idx = n // 2
11         if n % 2:
12             return values[idx]
13         else:
14             return (values[idx - 1] + values[idx]) / 2
15     else:
16         return None
17
18
19 if __name__ == "__main__":
20     print(median())
21     print(median(3, 7, 1, 6, 9))
22     print(median(1, 5, 8, 4, 3, 9))
23
```

ПРОБЛЕМЫ    ВЫХОДНЫЕ ДАННЫЕ    КОНСОЛЬ ОТЛАДКИ    ТЕРМИНАЛ


```
/usr/local/bin/python3 "/Users/itssyoma/Yandex.Disk.localized/Y-
itssyoma@MacBook-Air-Sema megarepo_210 % /usr/local/bin/python3
repo_210/example.py"
None
6.0
4.5
```

Рисунок 2 – Код и результат выполнения программы.

3. Выполнили задание №1. Решить поставленную задачу: написать функцию, вычисляющую среднее геометрическое своих аргументов  $a_1, a_2, \dots, a_n$

$$G = \sqrt[n]{\prod_{k=1}^n a_k}.$$

Если функции передается пустой список аргументов, то она должна возвращать значение None.



```

task1.py > ...
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4
5  def geometric_mean(*a):
6      if not a:
7          return None
8      n = len(a)
9      count = 1
10     for i in a:
11         count *= i
12     return count ** (1 / n)
13
14
15 if __name__ == "__main__":
16     print(geometric_mean(4, 5, 8))
17     print(geometric_mean())
18

```

ПРОБЛЕМЫ    ВЫХОДНЫЕ ДАННЫЕ    КОНСОЛЬ О

```

• itssyoma@MacBook-Air-Sema megarepo_210 % /l
repo_210/task1.py"
5.428835233189813
None

```

Рисунок 3 – Код и результат выполнения программы.

4. Выполнили задание №2. Решить поставленную задачу: написать функцию, вычисляющую среднее гармоническое своих аргументов  $a_1, a_2, \dots, a_n$

$$\frac{n}{H} = \sum_{k=1}^n \frac{1}{a_k}.$$

Если функции передается пустой список аргументов, то она должна возвращать значение None.

```
task2.py > ...
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4
5  def average_harmonic(*a):
6      if not a:
7          return None
8      count = 0
9      for i in a:
10         count += i ** -1
11     return len(a) / count
12
13
14 if __name__ == "__main__":
15     print(average_harmonic(1, 5, 7, 3))
16     print(average_harmonic())
17
```

ПРОБЛЕМЫ    ВЫХОДНЫЕ ДАННЫЕ    КОНСОЛЬ ОТЛАДКИ

```
itssyoma@MacBook-Air-Sema megarepo_210 % /usr/local/bin/python3 "/Users/itssyoma/Yandex.Disk.localized/Учеба/Основы программной инженерии/megarepo_210/task2.py"
2.3863636363636367
None
```

Рисунок 4 – Код и результат выполнения программы.

5. Выполнили индивидуальное задание №1 (вариант 1). Произведение аргументов, расположенных между максимальным и минимальным аргументами.

```
individual1.py > ...
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  import numpy as np
5
6
7  def product_between_min_max(*args):
8      if args:
9          start_index = min(args.index(min(args)), args.index(max(args))) + 1
10         end_index = max(args.index(min(args)), args.index(max(args)))
11         nums_for_product = [i for i in args[start_index:end_index]]
12         return np.prod(nums_for_product)
13     else:
14         return None
15
16
17 if __name__ == "__main__":
18     print("Произведение чисел между max и min =", product_between_min_max(1, 2, 3, 300, 5, 6, 7, -300, 14, 15, 16))
19     print("Результат передачи пустого списка аргументов: ", product_between_min_max())
20
```

ПРОБЛЕМЫ    ВЫХОДНЫЕ ДАННЫЕ    КОНСОЛЬ ОТЛАДКИ    ТЕРМИНАЛ    ПОРТЫ

```
/usr/local/bin/python3 "/Users/itssyoma/Yandex.Disk.localized/Учеба/Основы программной инженерии/megarepo_210/individual1.py"
itssyoma@MacBook-Air-Sema megarepo_210 % /usr/local/bin/python3 "/Users/itssyoma/Yandex.Disk.localized/Учеба/Основы программной инженерии/megarepo_210/individual1.py"
Произведение чисел между max и min = 210
Результат передачи пустого списка аргументов: None
```

Рисунок 5 – Код и результат выполнения программы.

6. Выполнили индивидуальное задание №2. Допустим, у нас есть функция, которая принимает переменное количество именованных аргументов, представляющих информацию о различных пользователях. Наша задача - вывести информацию о каждом пользователе.

```
individual2.py > ...
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4
5  def print_user_info(**users):
6      for username, info in users.items():
7          print(f"User: {username}")
8          for key, value in info.items():
9              print(f"{key}: {value}")
10         print()
11
12
13 if __name__ == "__main__":
14     print_user_info(
15         Anna={"age": 25, "city": "Moscow"},
16         Boris={"age": 30, "city": "Stavropol", "email": "boris@mail.ru"},
17         Clara={"age": 28, "city": "Pyatigorsk", "phone": "31-67-64"}
18     )
19
```

ПРОБЛЕМЫ    ВЫХОДНЫЕ ДАННЫЕ    КОНСОЛЬ ОТЛАДКИ    **ТЕРМИНАЛ**    ПОРТЫ

```
itssyoma@MacBook-Air-Sema megarepo_210 % /usr/local/bin/python3 "/Users/itssyoma/Ya
repo_210/individual2.py"
User: Anna
age: 25
city: Moscow

User: Boris
age: 30
city: Stavropol
email: boris@mail.ru

User: Clara
age: 28
city: Pyatigorsk
phone: 31-67-64
```

Рисунок 6 – Код и результат выполнения программы.

## Ответы на контрольные вопросы

### 1. Какие аргументы называются позиционными в Python?

Позиционные аргументы в Python - это аргументы, которые передаются в функцию по порядку их объявления.

### 2. Какие аргументы называются именованными в Python?

Именованные аргументы в Python - это аргументы, которые передаются в функцию с указанием их имени.

### 3. Для чего используется оператор \* ?

Оператор \* используется для распаковки последовательности (например, списка или кортежа) в отдельные элементы.

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

Конструкция \*args используется для передачи переменного числа позиционных аргументов в функцию, а \*\*kwargs используется для передачи переменного числа именованных аргументов.