

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

Отчет о лабораторной работе №13 по дисциплине «Основы
программной инженерии»

Выполнил:
Мамонтов Д.В.,
2 курс, группа ПИЖ-б-о-20-1,
Проверил:
Доцент кафедры инфокоммуникаций,
Воронкин Р.А.

Ставрополь, 2022 г

ХОД РАБОТЫ

```
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            return (values[idx - 1] + values[idx]) / 2
15
16    else:
17        return None
18
19
20 ▶ if __name__ == "__main__":
21     print(median())
22     print(median(3, 7, 1, 6, 9))
23     print(median(1, 5, 8, 4, 3, 9))
```

Рисунок 1 – код программы

```
None
6.0
4.5

Process finished with exit code 0
```

Рисунок 2 – результат работы программы

```
def geom(*args):
    if args:
        multi = 1
        values = [float(arg) for arg in args]
        n = len(values)
        for elem in values:
            multi *= elem
        return multi ** (1 / n)
    else:
        return None

if __name__ == '__main__':
    arguments = [float(i) for i in input("Enter the arguments: ").split()]
    print(f"The geometric mean of these arguments is: {geom(*arguments)}")
```

Рисунок 3 – код программы

```
5 4 6 4 6 3 7 8
5.135558978695357
```

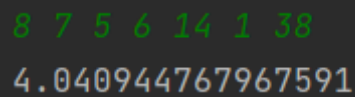
Рисунок 4 – результат работы программы

```
None
```

Рисунок 5 – результат работы программы при вводе пустого множества

```
1  def mid_harm(*args):
2      if args:
3          values = [float(arg) for arg in args]
4          n = len(values)
5          sum_of_reversed = 0
6          for value in values:
7              sum_of_reversed += (1 / value)
8          return n / sum_of_reversed
9      else:
10         return None
11
12
13  if __name__ == "__main__":
14      arguments = [float(i) for i in input().split()]
15      print(mid_harm(*arguments))
```

Рисунок 6 – код программы



```
8 7 5 6 14 1 38
4.040944767967591
```

Рисунок 7 – результат работы программы



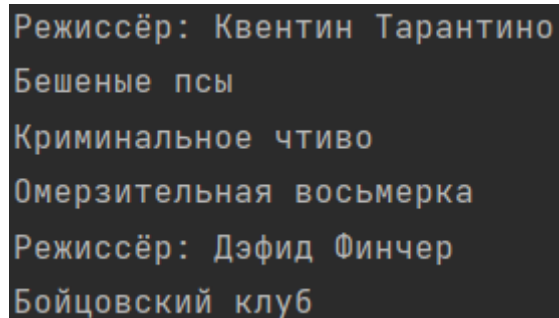
```
None
```

Рисунок 8 – результат работы программы при вводе пустого множества



```
1 def film(director, **films):
2     print(f"Режиссёр: {director}")
3     for films, name in films.items():
4         print(f"{name}")
5
6
7 if __name__ == '__main__':
8     film(
9         "Квентин Тарантино",
10        film1="Бешеные псы",
11        film2="Криминальное чтиво",
12        film3="Омерзительная восьмерка"
13    )
14    film(
15        "Дэвид Финчер",
16        film1="Бойцовский клуб"
17    )
```

Рисунок 9 – код программы



```
Режиссёр: Квентин Тарантино
Бешеные псы
Криминальное чтиво
Омерзительная восьмерка
Режиссёр: Дэвид Финчер
Бойцовский клуб
```

Рисунок 10 – результат работы программы

ИНДИВИДУАЛЬНОЕ ЗАДАНИЕ

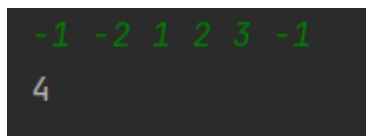
Задание:

Подсчитать сумму аргументов, расположенных после первого положительного аргумента

Код:

```
def after_sum(*args):
    if args:
        i = 0
        for index, arg in enumerate(args):
            if arg > 0:
                i = index
        pos_s = sum(arg for index, arg in enumerate(args) if index < i)
        return pos_s
    else:
        return None

if __name__ == "__main__":
    arguments = [int(i) for i in input().split()]
    arguments.reverse()
    print(after_sum(*arguments))
```



```
-1 -2 1 2 3 -1
4
```

Рисунок 1 – результат работы программы



```
None
```

Рисунок 2 – результат работы программы при вводе пустого множества

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Какие аргументы называются позиционными в Python?
Позиционные аргументы - это такие аргументы, значение которых будет зависеть от их позиции. Пример: `def test(a, b) -> a, b` – это позиционные аргументы. Именно по позиции, расположению аргумента, функция понимает, какому параметру он соответствует.
2. Какие аргументы называются именованными в Python?
Аргументы, передаваемые с именами, называются именованными. При вызове функции можно использовать имена параметров из ее определения.
3. Для чего используется оператор `*` ?

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

```
a = [1, 2, 3]
```

```
b = [*a, 4, 5, 6]
```

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

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

Оператор «звёздочка» в Python способен «вытаскивать» из объектов составляющие их элементы. Существует два вида параметров функций, а именно: `*args` — это сокращение от «arguments» (аргументы), а `**kwargs` — сокращение от «keyword arguments» (именованные аргументы).