

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 №13
дисциплины «Программирование на Python»

Выполнил:
Горбунов Данила Евгеньевич
2 курс, группа ИВТ-б-о-22-1,
09.03.01 «Информатика и
вычислительная техника»,
направленность (профиль)
«Программное обеспечение средств
вычислительной техники и
автоматизированных систем», очная
форма обучения

(подпись)

Руководитель практики:
Воронкин Р. А., канд. технических
наук, доцент кафедры
инфокоммуникаций

(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

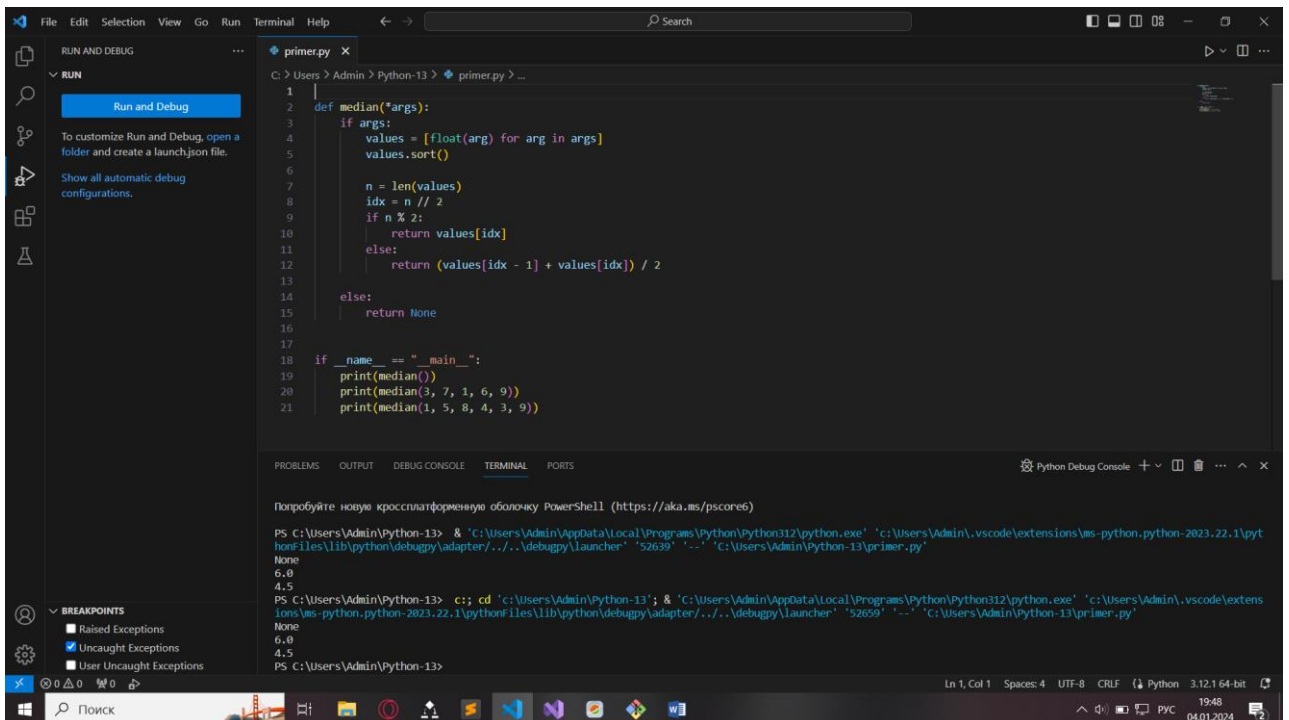
Ставрополь, 2023 г.

Тема: Функции с переменным числом параметров в Python

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. Создал общедоступный репозиторий на GitHub, в котором использована лицензия MIT и язык программирования Python. Выполнил клонирование созданного репозитория.
2. Дополнил файл .gitignore необходимыми правилами.
3. Организовал созданный репозиторий в соответствие с моделью ветвления git-flow.
4. Проработал пример лабораторной работы. Создал для него отдельный модуль языка Python. Привел в отчете скриншоты результата выполнения программы примера.



The screenshot shows the Visual Studio Code interface. The editor window displays a Python script named `primer.py` with the following code:

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

The terminal window at the bottom shows the execution of the script using PowerShell. The output is:

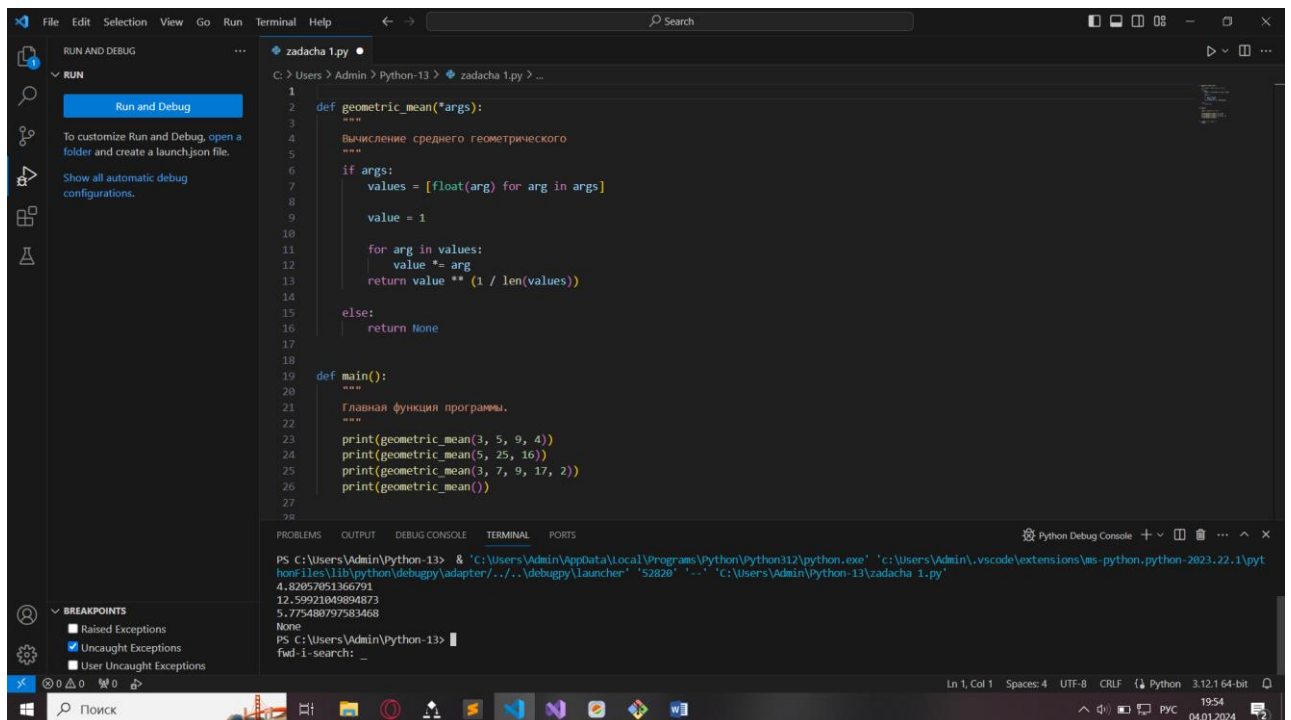
```
PS C:\Users\Admin\Python-13> & 'c:\Users\Admin\AppData\Local\Programs\Python\Python312\python.exe' 'c:\Users\Admin\.vscode\extensions\ms-python.python-2023.22.1\pythonFiles\lib\python\debugpy\adapter\..\..\debugpy\launcher' '52639' '-.' 'c:\Users\Admin\Python-13\primer.py'
None
6.0
4.5
PS C:\Users\Admin\Python-13> c++; cd 'c:\Users\Admin\Python-13'; & 'c:\Users\Admin\AppData\Local\Programs\Python\Python312\python.exe' 'c:\Users\Admin\.vscode\extensions\ms-python.python-2023.22.1\pythonFiles\lib\python\debugpy\adapter\..\..\debugpy\launcher' '52659' '-.' 'c:\Users\Admin\Python-13\primer.py'
4.5
PS C:\Users\Admin\Python-13>
```

Рисунок 1. Результат работы программы из примера 1

5. Решил следующую 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



```
1 def geometric_mean(*args):
2     """
3     Вычисление среднего геометрического
4     """
5
6     if args:
7         values = [float(arg) for arg in args]
8
9         value = 1
10
11        for arg in values:
12            value *= arg
13        return value ** (1 / len(values))
14
15    else:
16        return None
17
18
19 def main():
20     """
21     Главная функция программы.
22     """
23     print(geometric_mean(3, 5, 9, 4))
24     print(geometric_mean(5, 25, 16))
25     print(geometric_mean(3, 7, 9, 17, 2))
26     print(geometric_mean())
27
28
29 if __name__ == '__main__':
30     main()
```

Terminal Output:

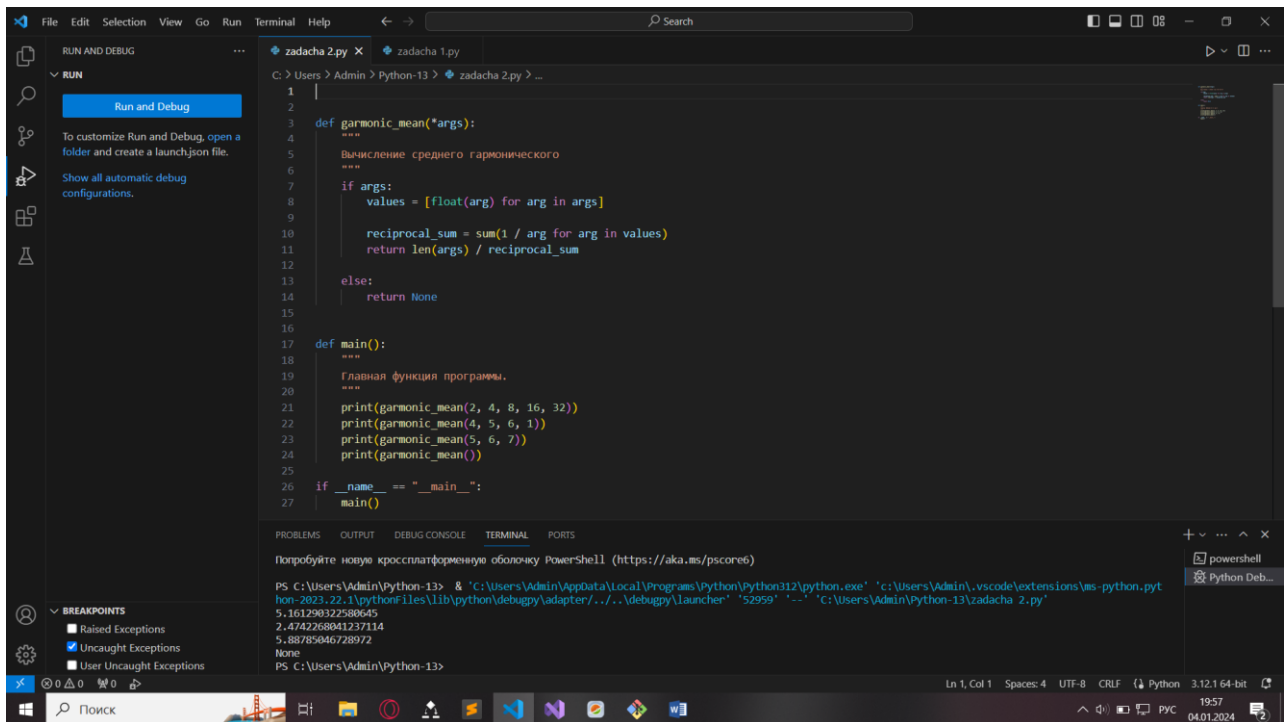
```
PS C:\Users\Admin\Python-13> & 'C:\Users\Admin\AppData\Local\Programs\Python\Python312\python.exe' 'c:\Users\Admin\.vscode\extensions\ms-python.python-2023.22.1\pythonFiles\lib\python\debugpy\adapter\..\..\debugpy\launcher' '52820' '-.' 'C:\Users\Admin\Python-13\zadacha_1.py'
4.80857851366791
12.59921049894873
5.775480797583468
None
PS C:\Users\Admin\Python-13>
```

Рисунок 2. Результат работы программы из задачи 1

6. Решил следующую задачу: написать функцию, вычисляющую среднее гармоническое своих аргументов a_1, a_2, \dots, a_n

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

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



```
1 |
2 |
3 | def garmonic_mean(*args):
4 |     """
5 |     Вычисление среднего гармонического
6 |     """
7 |     if args:
8 |         values = [float(arg) for arg in args]
9 |
10 |        reciprocal_sum = sum(1 / arg for arg in values)
11 |        return len(args) / reciprocal_sum
12 |
13 |    else:
14 |        return None
15 |
16 |
17 | def main():
18 |     """
19 |     Главная функция программы.
20 |     """
21 |     print(garmonic_mean(2, 4, 8, 16, 32))
22 |     print(garmonic_mean(4, 5, 6, 1))
23 |     print(garmonic_mean(5, 6, 7))
24 |     print(garmonic_mean())
25 |
26 | if __name__ == "__main__":
27 |     main()
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

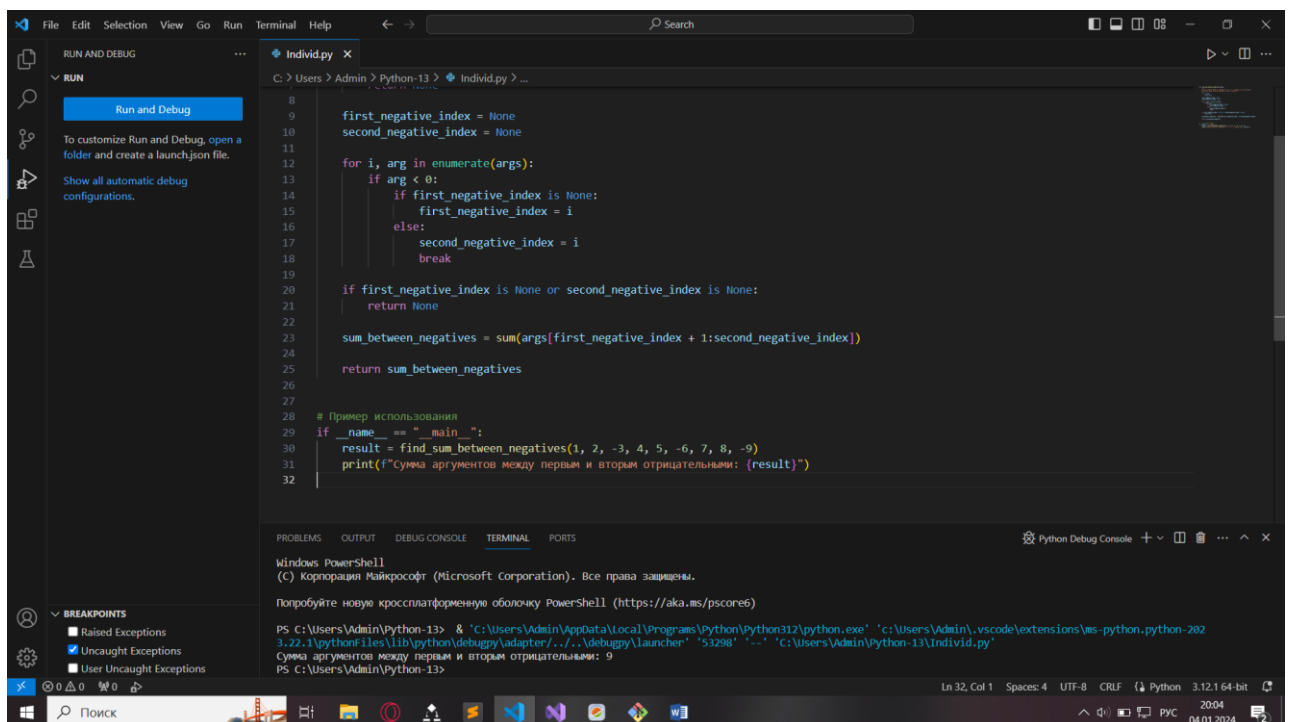
Попробуйте новую кроссплатформенную оболочку PowerShell (https://aka.ms/powershell)

PS C:\Users\Admin\Python-13> & 'C:\Users\Admin\AppData\Local\Programs\Python\Python312\python.exe' 'c:\Users\Admin\.vscode\extensions\ms-python.python-2023.22.1\pythonFiles\lib\python\debugpy\adapter\..\..\debugpy\launcher' '52959' '--' 'C:\Users\Admin\Python-13\zadacha_2.py'

5.161298322580645
2.4742268841237114
5.88785946728972
None
PS C:\Users\Admin\Python-13>

Рисунок 3. Результат работы программы из задачи 2

7. Выполнил индивидуальное задание, согласно варианту 8. 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. Согласно варианту, необходимо найти сумму аргументов, расположенных между первым и вторым отрицательными аргументами.



The screenshot shows the Visual Studio Code interface with a Python file named 'Individ.py' open. The code defines a function 'find_sum_between_negatives' that takes an arbitrary number of arguments and returns the sum of elements between the first and second negative numbers. The function returns None if there are fewer than two negative numbers. A test case is provided in the main block, calling the function with the arguments (1, 2, -3, 4, 5, -6, 7, 8, -9) and printing the result.

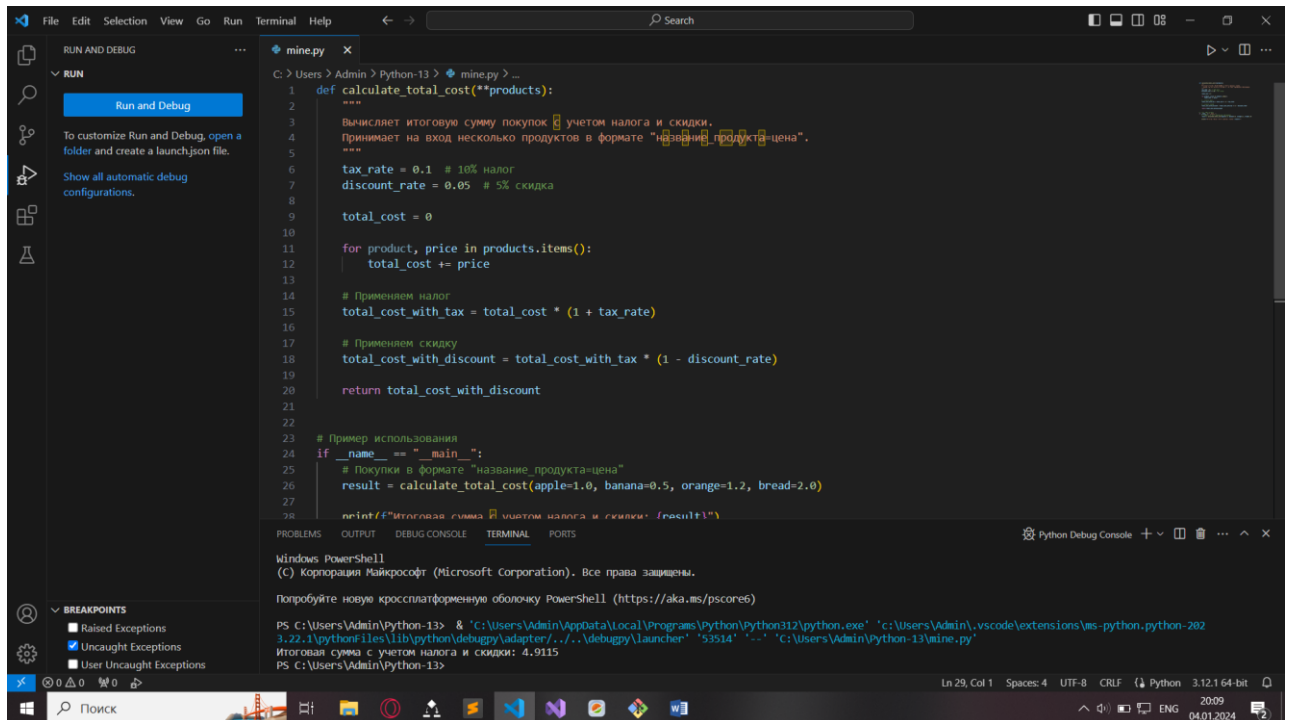
```
8
9
10 first_negative_index = None
11 second_negative_index = None
12
13 for i, arg in enumerate(args):
14     if arg < 0:
15         if first_negative_index is None:
16             first_negative_index = i
17         else:
18             second_negative_index = i
19             break
20
21 if first_negative_index is None or second_negative_index is None:
22     return None
23
24 sum_between_negatives = sum(args[first_negative_index + 1:second_negative_index])
25
26 return sum_between_negatives
27
28 # Пример использования
29 if __name__ == "__main__":
30     result = find_sum_between_negatives(1, 2, -3, 4, 5, -6, 7, 8, -9)
31     print(f"Сумма аргументов между первым и вторым отрицательными: {result}")
32
```

The terminal output shows the command prompt running the script, and the output is: "Сумма аргументов между первым и вторым отрицательными: 9".

Рисунок 4. Результат работы программы из индивидуального задания

8. Самостоятельно подберите или придумайте задачу с переменным числом именованных аргументов. Приведите решение этой задачи.

Задача: Напишите функци., которая принимает на вход несколько продуктов с их ценами, а затем вычисляет итоговую сумму с учетом налога и скидки.



The screenshot shows the Visual Studio Code interface. The editor window displays a Python file named `mine.py` with the following code:

```
1 def calculate_total_cost(**products):
2     """
3     Вычисляет итоговую сумму покупок с учетом налога и скидки.
4     Принимает на вход несколько продуктов в формате "название_продукта=цена".
5     """
6     tax_rate = 0.1 # 10% налог
7     discount_rate = 0.05 # 5% скидка
8
9     total_cost = 0
10
11     for product, price in products.items():
12         total_cost += price
13
14     # Применяем налог
15     total_cost_with_tax = total_cost * (1 + tax_rate)
16
17     # Применяем скидку
18     total_cost_with_discount = total_cost_with_tax * (1 - discount_rate)
19
20     return total_cost_with_discount
21
22
23 # Пример использования
24 if __name__ == "__main__":
25     # Покупки в формате "название_продукта=цена"
26     result = calculate_total_cost(apple=1.0, banana=0.5, orange=1.2, bread=2.0)
27
28     print(f"Итоговая сумма с учетом налога и скидки: {result}")
```

The terminal window at the bottom shows the execution of the script:

```
PS C:\Users\Admin\Python-13> & 'C:\Users\Admin\AppData\Local\Programs\Python\Python312\python.exe' 'C:\Users\Admin\.vscode\extensions\ms-python.python-2023.22.1\pythonFiles\lib\python\debugpy\adapter\..\..\debugpy\launcher' '53514' '-.' 'C:\Users\Admin\Python-13\mine.py'
Итоговая сумма с учетом налога и скидки: 4.9115
PS C:\Users\Admin\Python-13>
```

Рисунок 5. Результат работы программы из придуманного задания

Контрольные вопросы

1. Какие аргументы называются позиционными в Python?

В Python аргументы называются позиционными, если они передаются функции в том же порядке, в котором они определены в функции.

В функцию также можно передать переменное количество позиционных аргументов. Это делается с помощью оператора `*` перед именем аргумента в определении функции.

2. Какие аргументы называются именованными в Python?

В Python аргументы называются именованными, если они передаются функции с указанием имени аргумента, за которым следует значение аргумента.

В функцию также можно передать переменное количество именованных аргументов. Это делается с помощью оператора `**` перед именем аргумента в определении функции.

3. Для чего используется оператор `*` ?

Оператор `*` чаще всего ассоциируется у людей с операцией умножения, но в Python он имеет и другой смысл. Этот оператор позволяет «распаковывать» объекты, внутри которых хранятся некие элементы.

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

Итак, мы знаем о том, что оператор «звёздочка» в Python способен «вытаскивать» из объектов составляющие их элементы. Знаем мы и о том, что существует два вида параметров функций. А именно, `*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: в результате выполнения работы были приобретены навыки по работе с функциями с переменным числом параметров при написании программ с помощью языка программирования Python версии 3.x