

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 ПРАКТИЧЕСКОЙ РАБОТЕ №2.9
дисциплины «Программирование на Python»

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

(подпись)

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

(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

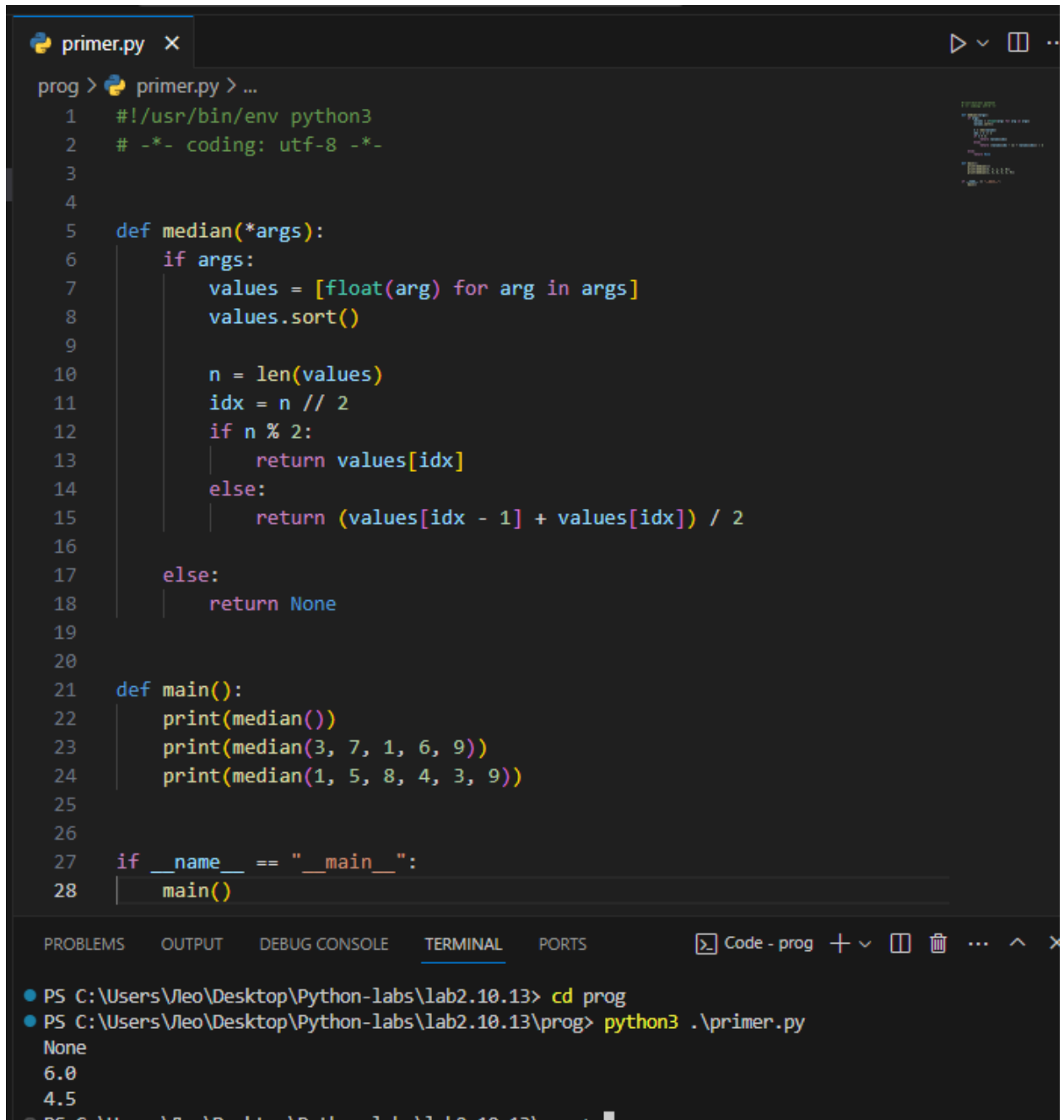
Ставрополь, 2023 г.

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.

Порядок выполнения работы:

1. Написал программу (primer1.py), в которой проработал пример лабораторной работы:



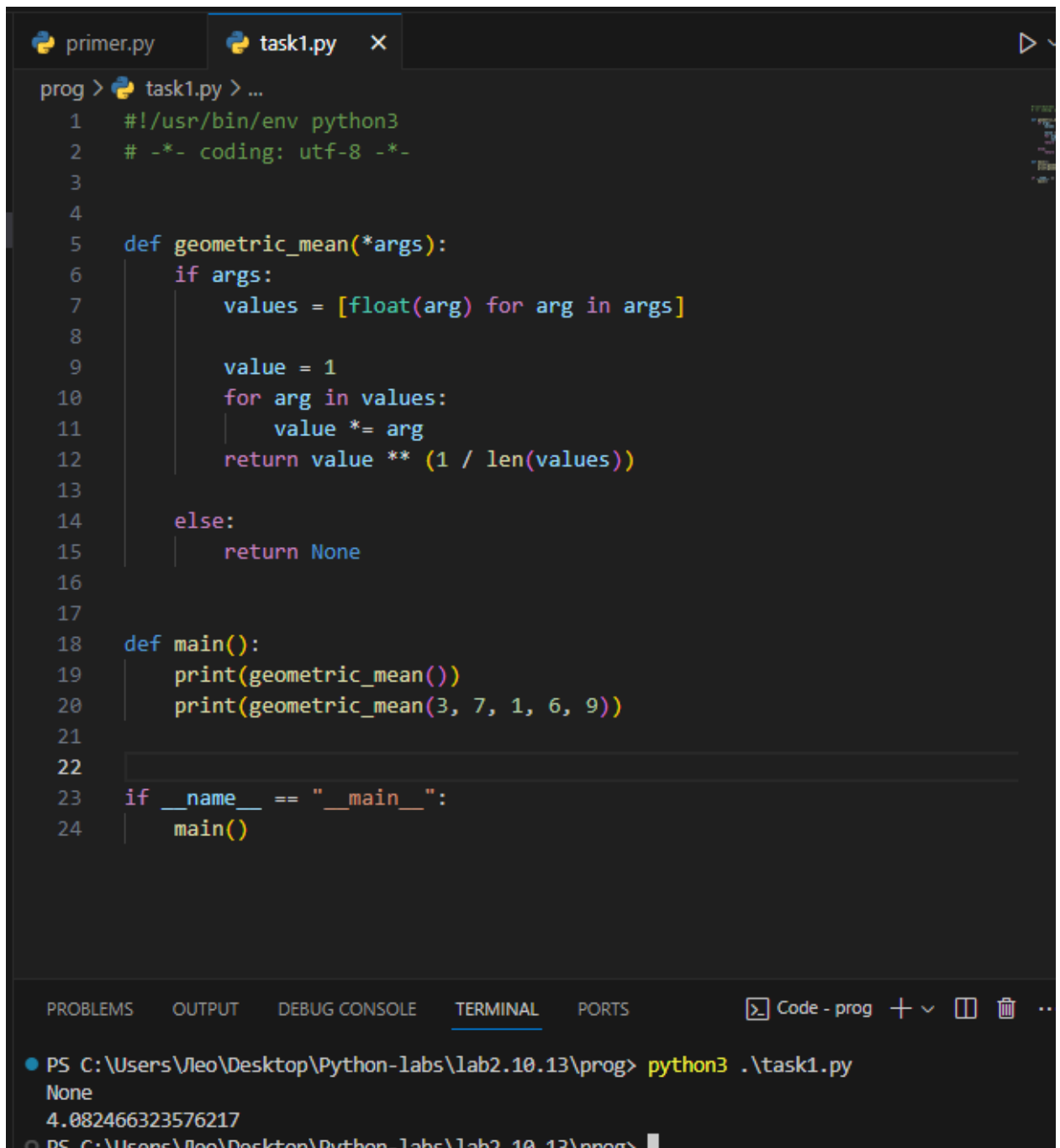
```
primer.py x
prog > primer.py > ...
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4
5  def median(*args):
6      if args:
7          values = [float(arg) for arg in args]
8          values.sort()
9
10         n = len(values)
11         idx = n // 2
12         if n % 2:
13             return values[idx]
14         else:
15             return (values[idx - 1] + values[idx]) / 2
16
17     else:
18         return None
19
20
21 def main():
22     print(median())
23     print(median(3, 7, 1, 6, 9))
24     print(median(1, 5, 8, 4, 3, 9))
25
26
27 if __name__ == "__main__":
28     main()
```

PROBLEMS OUTPUT DEBUG CONSOLE **TERMINAL** PORTS Code - prog + -

```
PS C:\Users\leo\Desktop\Python-labs\lab2.10.13> cd prog
PS C:\Users\leo\Desktop\Python-labs\lab2.10.13\prog> python3 .\primer.py
None
6.0
4.5
PS C:\Users\leo\Desktop\Python-labs\lab2.10.13\prog>
```

Рисунок 1 – Результат работы программы primer1.py

2. Создал файл (task1.py) в которой решил задачу: решить поставленную задачу: написать функцию, вычисляющую среднее геометрическое своих аргументов.



```
primer.py task1.py X
prog > task1.py > ...
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4
5  def geometric_mean(*args):
6      if args:
7          values = [float(arg) for arg in args]
8
9          value = 1
10         for arg in values:
11             value *= arg
12         return value ** (1 / len(values))
13
14     else:
15         return None
16
17
18 def main():
19     print(geometric_mean())
20     print(geometric_mean(3, 7, 1, 6, 9))
21
22
23 if __name__ == "__main__":
24     main()
```

PROBLEMS OUTPUT DEBUG CONSOLE **TERMINAL** PORTS Code - prog + - [] [X] ..

```
PS C:\Users\leo\Desktop\Python-labs\lab2.10.13\prog> python3 .\task1.py
None
4.082466323576217
PS C:\Users\leo\Desktop\Python-labs\lab2.10.13\prog>
```

Рисунок 2 – Результат работы программы task1.py

3. Создал файл (task2.py) в которой решил задачу: решить поставленную задачу: написать функцию, вычисляющую среднее гармоническое своих аргументов



```
primer.py task1.py task2.py X
prog > task2.py > garmonic_mean
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4
5  def garmonic_mean(*args):
6      if args:
7          values = [float(arg) for arg in args]
8
9          reciprocal_sum = sum(1 / arg for arg in values)
10         return len(args) / reciprocal_sum
11
12     else:
13         return None
14
15
16 def main():
17     print(garmonic_mean())
18     print(garmonic_mean(3, 7, 1, 6, 9))
19
20
21 if __name__ == "__main__":
22     main()

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\Leo\Desktop\Python-labs\lab2.10.13\prog> python3 .\task2.py
None
2.850678733031674
```

Рисунок 2 – Результат работы программы task2.py

Индивидуальное задание (17 Вариант): 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.

```
primer.py task1.py ind.py x
prog > ind.py > sum_of_int_parts
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4
5  def sum_of_int_parts(*args):
6      ...
7      Функция расчета целых частей после последнего отрицательного.
8      ...
9      if args:
10         values = [float(arg) for arg in args]
11
12         found_negative = False
13         total_sum = 0
14         for arg in values:
15             if arg < 0:
16                 found_negative = True
17                 total_sum = 0
18             elif found_negative:
19                 total_sum += int(arg)
20         return total_sum
21
22     else:
23         return None
24
25
26 def main():
27     ...
28     Главная функция программы.
29     ...
30     print(sum_of_int_parts())
31     print(sum_of_int_parts(3, -7, -1, 6, 9))
32     print(sum_of_int_parts(1, -5, 8, 4, 3, 9))
33
34
35 if __name__ == "__main__":
36     main()
```

PROBLEMS OUTPUT DEBUG CONSOLE **TERMINAL** PORTS

```
PS C:\Users\leo\Desktop\Python-labs\lab2.10.13\prog> python3 .\ind.py
None
15
24
PS C:\Users\leo\Desktop\Python-labs\lab2.10.13\prog>
```

Вывод: в ходе выполнения работы были приобретены навыки по работе с функциями с переменным числом параметров при написании программ с помощью языка программирования Python версии 3.x.

Контрольные вопросы

1. Какие аргументы называются позиционными в Python?

Позиционные аргументы в Python – это аргументы, которые передаются в функцию в определенном порядке. Они соответствуют порядку параметров в определении функции.

2. Какие аргументы называются именованными в Python?

Именованные аргументы в Python - это аргументы, которые передаются в функцию с указанием их имени. Они позволяют передавать аргументы в любом порядке.

3. Для чего используется оператор * ?

Оператор * в Python используется для распаковки итерируемых объектов, таких как списки или кортежи, в аргументы функции. Он также может быть использован для распаковки словарей в именованные аргументы функции.

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

*args и **kwargs – это общепринятые названия для аргументов функций, которые позволяют передавать переменное количество позиционных и именованных аргументов соответственно. *args используется для передачи переменного количества позиционных аргументов, а **kwargs - для передачи переменного количества именованных аргументов.