

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ
ФЕДЕРАЦИИ**

**Федеральное государственное автономное образовательное учреждение
высшего образования**

«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития

Кафедра информационных систем и технологий

Отчет по лабораторной работе №13.

Дисциплина: «Основы программной инженерии»

Выполнил:

Студент группы ПИЖ-б-о-22-1,

направление подготовки: 09.03.04

«Программная инженерия»

ФИО: Гуртовой Ярослав Дмитриевич

Проверил:

Богданов С.С

Ставрополь 2023

Тема: Лабораторная работа 2.10 Функции с переменным число 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.


Выполнение работы:


1. Изучил теоретический материал работы.
2. Создал репозиторий на git.hub.

Рисунок 1 – создание репозитория

3. Клонировал репозиторий.


Required fields are marked with an asterisk (*).


Owner *  KingItProgger / Repository name *

 lr-2.10 is available.

Great repository names are short and memorable. Need inspiration? How about [improved-disco](#) ?

Description (optional)

☒  **Public**
Anyone on the internet can see this repository. You choose who can commit.

☐  **Private**
You choose who can see and commit to this repository.

Initialize this repository with:


☒ **Add a README file**
This is where you can write a long description for your project. [Learn more about READMEs.](#)


Add .gitignore

Choose which files not to track from a list of templates. [Learn more about ignoring files.](#)

Choose a license

A license tells others what they can and can't do with your code. [Learn more about licenses.](#)

This will set  **main** as the default branch. Change the default name in your [settings](#).

 You are creating a public repository in your personal account.

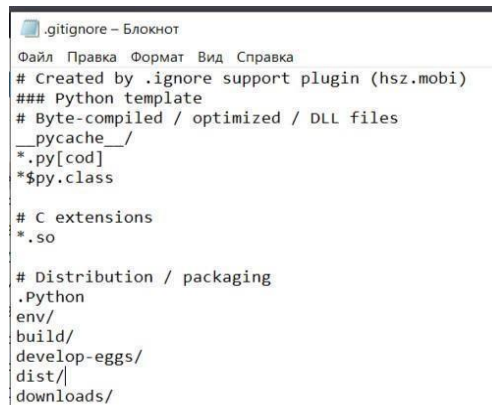
[Create repository](#)

4.

```
PS C:\Users\User\Desktop\учеба\Зсемак\змій\lr_2.10> git clone http
Cloning into 'lr-2.10'...
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 5 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (5/5), done.
PS C:\Users\User\Desktop\учеба\Зсемак\змій\lr_2.10> cd lr-2.10
PS C:\Users\User\Desktop\учеба\Зсемак\змій\lr_2.10\lr-2.10>
```

Рисунок 2 – клонирование репозитория 4.

Дополнить файл gitignore необходимыми правилами.



```
.gitignore - Блокнот
Файл Правка Формат Вид Справка
# Created by .ignore support plugin (hsz.mobi)
### Python template
# Byte-compiled / optimized / DLL files
__pycache__/
*.py[cod]
*$py.class

# C extensions
*.so

# Distribution / packaging
.Python
env/
build/
develop-eggs/
dist/
downloads/
```

Рисунок 3 – .gitignore для IDE PyCharm

5. Организовать свой репозиторий в соответствии с моделью ветвления git-flow.

```
PS C:\Users\User\Desktop\учеба\Зсемак\змій\lr_2.10\lr-2.10> git checkout -b develop
Switched to a new branch 'develop'
PS C:\Users\User\Desktop\учеба\Зсемак\змій\lr_2.10\lr-2.10>
```

Рисунок 4 – создание ветки develop

6. Проработал примеры из методички

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

def median(*args):
    if args:
        values = [float(arg) for arg in args]
        values.sort()
        n = len(values)
        idx = n // 2
        if n % 2:
            return values[idx]
        else:
            return (values[idx - 1] + values[idx]) / 2
    else:
        return None

if __name__ == "__main__":
    print(median())
    print(median(3, 7, 1, 6, 9))
    print(median(1, 5, 8, 4, 3, 9))
```

Рисунок 5 – пример 1

```
C:\Users\User\AppData\Local\Programs\Python\Python311\python.exe "C:/Users/User/Desktop/ОПИ/ІІІ 2.10/ex1.py"
None
6.0
4.5

Process finished with exit code 0
```

Рисунок 6 – пример выполнения примера 1

7. Решить поставленную задачу: написать функцию, вычисляющую среднее геометрическое своих аргументов. Если функции передается пустой список аргументов, то она должна возвращать значение None.

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

import math

def medGeom(*a):
    if a:
        n = len(a)
        y = 1
        for i in a:
            y *= i

        s = math.pow(y, 1/n)
        return s

    else:
        return None

if __name__ == "__main__":
    p = list(int(i) for i in input("Введите значения: ").split())
    result = medGeom(*p)
    print(result)
```

Рисунок 7 – задание 8

```
C:\Users\User\AppData\Local\Programs\Python\Python311\python.exe "C:/Users/User/Desktop/ОПИ/лр 2.10/task1.py"
Введите значения: 2 3 4
2.8844991406148166

Process finished with exit code 0
```

Рисунок 8 – пример выполнения 8 задания

8. Решить поставленную задачу: написать функцию, вычисляющую среднее гармоническое своих аргументов. Если функции передается пустой список аргументов, то она должна возвращать значение None .

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

def midgeom(*a):
    if a:
        n = len(a)
        y = 0
        for i in a:
            y += 1/i
        return n/y

    else:
        return None

if __name__ == "__main__":
    p = list(int(i) for i in input("Введите значения: ").split())
    result = midgeom(*p)
    print(result)
```

Рисунок 9 – выполнение задания 9

```
C:\Users\User\AppData\Local\Programs\Python\Python311\python.exe "C:/Users/User/Desktop/ОПИ/1r 2.10/task2.py"
Введите значения: 8 10 46
9.324324324324323

Process finished with exit code 0
```

Рисунок 10 – результат выполнения задания 9

9. 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.

5. Сумму аргументов, расположенных до последнего положительного аргумента.

```

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

def sum_before_last_positive(*args):
    if args:
        l = 0
        for i in args:
            if i > 0:
                l += i
            else: break

        return l

    else:
        return None

if __name__ == '__main__':
    p = list(int(i) for i in input("Введите значения: ").split())
    result = sum_before_last_positive(*p)
    print(result)

```

Рисунок 11 – выполнение индивидуального задания

```

C:\Users\User\AppData\Local\Program
Введите значения: 1 3 4 -1 2
8

Process finished with exit code 0

```

Рисунок 12 – результат выполнения индивидуального задания

9. Зафиксировал все изменения в github в ветке develop.

```

PS C:\Users\User\Desktop\учеба\Зсемак\эмй\lr_2.10\lr-2.10> git add .
warning: in the working copy of 'pycharm/.idea/inspectionProfiles/profiles_settings.xml'
the next time Git touches it
PS C:\Users\User\Desktop\учеба\Зсемак\эмй\lr_2.10\lr-2.10> git commit -m "laba"
[develop 0bb37e2] laba
 9 files changed, 111 insertions(+)
 create mode 100644 pycharm/.idea/.gitignore
 create mode 100644 pycharm/.idea/inspectionProfiles/profiles_settings.xml
 create mode 100644 pycharm/.idea/lr 2.10.iml
 create mode 100644 pycharm/.idea/misc.xml
 create mode 100644 pycharm/.idea/modules.xml
 create mode 100644 pycharm/ex1.py
 create mode 100644 pycharm/my_tas.py
 create mode 100644 pycharm/task1.py
 create mode 100644 pycharm/task2.py
PS C:\Users\User\Desktop\учеба\Зсемак\эмй\lr_2.10\lr-2.10>

```

Рисунок 13 – фиксация изменений в ветку develop

10. Слил ветки.

```

PS C:\Users\User\Desktop\учеба\Зсемак\эмй\lr_2.10\lr-2.10> git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
PS C:\Users\User\Desktop\учеба\Зсемак\эмй\lr_2.10\lr-2.10> git merge develop
Updating 4394c86..0bb37e2
Fast-forward
 pycharm/.idea/.gitignore           | 3 +++
 .../.idea/inspectionProfiles/profiles_settings.xml | 6 +++++
 pycharm/.idea/lr 2.10.iml          | 8 ++++++++
 pycharm/.idea/misc.xml             | 4 ++++
 pycharm/.idea/modules.xml          | 8 ++++++++
 pycharm/ex1.py                     | 21 +++++++++++++++++++++
 pycharm/my_tas.py                   | 20 +++++++++++++++++++++
 pycharm/task1.py                    | 22 +++++++++++++++++++++
 pycharm/task2.py                    | 19 ++++++++++++++++++++
 9 files changed, 111 insertions(+)
 create mode 100644 pycharm/.idea/.gitignore
 create mode 100644 pycharm/.idea/inspectionProfiles/profiles_settings.xml
 create mode 100644 pycharm/.idea/lr 2.10.iml
 create mode 100644 pycharm/.idea/misc.xml
 create mode 100644 pycharm/.idea/modules.xml
 create mode 100644 pycharm/ex1.py
 create mode 100644 pycharm/my_tas.py
 create mode 100644 pycharm/task1.py
 create mode 100644 pycharm/task2.py
PS C:\Users\User\Desktop\учеба\Зсемак\эмй\lr_2.10\lr-2.10>

```


Контрольные вопросы:

1. Какие аргументы называются позиционными в Python?

Позиционные аргументы передаются в функцию в том порядке, в котором они объявлены в сигнатуре функции.

Значения, переданные в качестве аргументов, присваиваются параметрам в том порядке, в котором они объявлены в определении функции.

2. Какие аргументы называются именованными в Python?

Именованные аргументы передаются с указанием имени параметра и значения, которое вы хотите присвоить этому параметру.

Именованные аргументы могут быть переданы в любом порядке.

3. Для чего используется оператор * ?

Благодаря использованию * мы создаем список позиционных аргументов на основе того, что было передано функции при вызове. Также наоборот раскрываем список, раскладывая по элементам.

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

Конструкции *args и **kwargs в Python используются для передачи переменного числа аргументов в функцию. Они облегчают работу с функциями, которые могут принимать разное количество аргументов.

*args позволяет передавать переменное количество позиционных аргументов в функцию.

Звездочка (*) перед именем args означает, что все аргументы, следующие после *args, будут собраны в кортеж.

**kwargs позволяет передавать переменное количество именованных (ключевых) аргументов в функцию.

Звездочки с двумя знаками перед именем kwargs означают, что все переданные именованные аргументы будут собраны в словарь.