

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ
ФЕДЕРАЦИИ**
**Федеральное государственное автономное
образовательное учреждение высшего образования
«СЕВЕРОКАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»**

Кафедра инфокоммуникаций

Институт цифрового развития

ОТЧЁТ

по лабораторной работе №4

Дисциплина: «Программирование на Python»

Тема: «Функции с переменным числом параметров в Python»

Выполнил: студент 2 курса

группы ИВТ-б-о-21-1

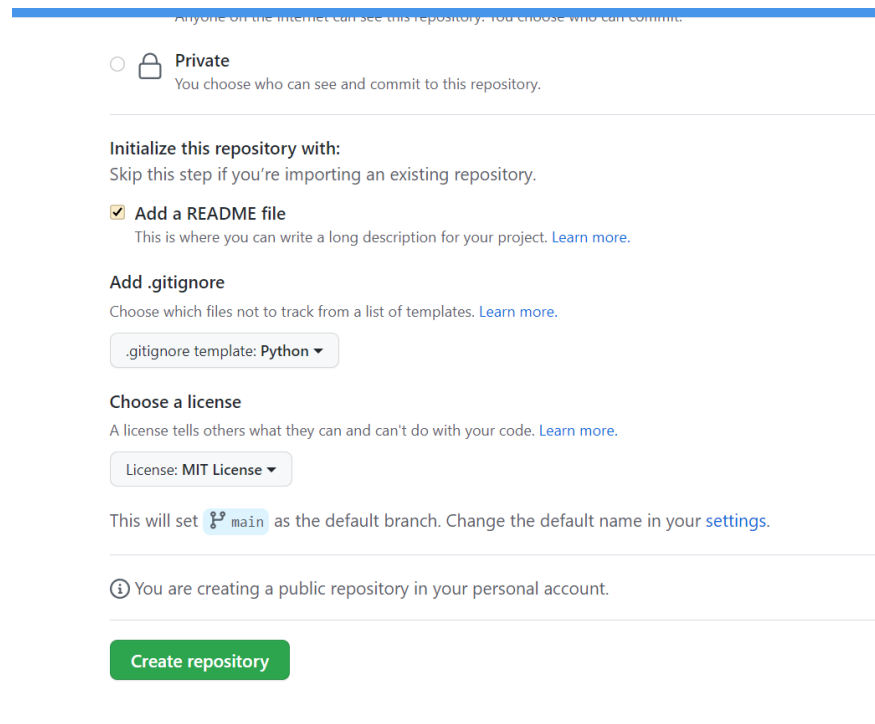
Уланбекова Айканыш Уланбековна

Ставрополь 2022


Цель работы: 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. Создать общедоступный репозиторий на GitHub, в котором будет использована лицензия MIT и язык программирования Python.



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:
Skip this step if you're importing an existing repository.


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


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

.gitignore template: **Python** ▼

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

License: **MIT License** ▼

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

Рисунок 1. Создание репозитория

2. Выполните клонирование созданного репозитория.

```
C:\Users\User>cd C:\Users\User\Desktop\2 курс Python\lab 13
C:\Users\User\Desktop\2 курс Python\lab 13>git clone https://github.com/aikanyshkauanbekova/lab13.git
Cloning into 'lab13'...
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.
C:\Users\User\Desktop\2 курс Python\lab 13>_
```

Рисунок 2. Клонирование репозитория

3. Дополните файл .gitignore необходимыми правилами для работы с IDE PyCharm.

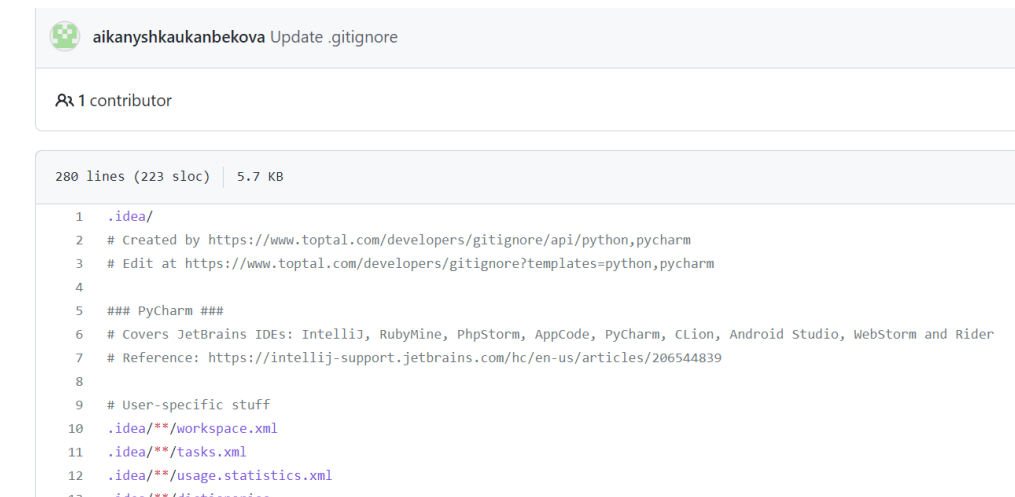


Рисунок 3. Дополнение файла .gitignore

4.Организуйте свой репозиторий в соответствии с моделью ветвления git-flow.

```
C:\Users\User\Desktop\2 кypc Python\lab 10\lab-10>git flow init
which branch should be used for bringing forth production releases?
- main
Branch name for production releases: [main]
Branch name for "next release" development: [develop]

How to name your supporting branch prefixes?
Feature branches? [feature/]
Bugfix branches? [bugfix/]
Release branches? [release/]
Hotfix branches? [hotfix/]
Support branches? [support/]
Version tag prefix? []
Hooks and filters directory? [C:/Users/User/Desktop/2 кypc Python/lab 10\lab-10/.git/hooks]
C:\Users\User\Desktop\2 кypc Python\lab 10\lab-10>
```

Рисунок 4. Организован модель ветвления git flow

5.Проработайте пример лабораторной работы. Создайте для него отдельный модуль языка Python. Зафиксируйте изменения в репозитории.

```
#!/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 лаб работы

6. Решите задачу:

8. Решить поставленную задачу: написать функцию, вычисляющую среднее геометрическое своих аргументов a_1, a_2, \dots, a_n

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

9. Решить поставленную задачу: написать функцию, вычисляющую среднее гармоническое своих аргументов a_1, a_2, \dots, a_n

$$\frac{n}{H} = \sum_{k=1}^n \frac{1}{a_k}. \quad (2)$$

```

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

import math

def sr_geom(*args):
    if args:
        values = [float(arg) for arg in args]
        n = len(values)
        pr = 1
        k = 0
        for k in range(n):
            pr *= values[k]
            k += 1
        g = math.pow(pr, 1 / k)
        return g
    else:
        return None

def sr_garm(*args):
    if args:
        values = [float(arg) for arg in args]
        n = len(values)
        s = 0
        for k in range(n):
            s += 1/values[k]
        g = n / s
        return g
    else:
        return None

def intro(**kwargs):
    for key, value in kwargs.items():

```

```

        print("{} is {}".format(key, value))

if __name__ == "__main__":
    print(sr_geom(1, 2, 3, 4))
    print(sr_garm(1, 2, 3, 4))

    print('\n')
    intro(
        Firstname="Sita",
        Lastname="Sharma",
        Phone=1234567890,
        Age=22
    )

    print('\n')
    intro(
        Firstname="John",
        Lastname="Wood",
        Email="john@mail.com",
        Country="Wakanda",
        Phone=9876543210,
        Age=25
    )

```

Рисунок 6. Выполненные задания

7. Индивидуальное задание

Вариант 9. Сумму модулей аргументов, расположенных после первого аргумента, равного нулю.

```

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

def sm(*args):
    if args:
        values = [float(arg) for arg in args]
        print(sum(map(abs, values[values.index(0) + 1:])))
    else:
        return None

if __name__ == "__main__":
    print(sm())
    print(sm(3, 0, 0, -6, 9))
    print(sm(1, 0, 0, 4, 3, 9))

```

Рисунки 7. Выполненное индивидуальное задание

8. Сделала коммит, выполнил слияние с веткой main, и запустил изменения в уд. репозиторий.

```
C:\Users\User\Desktop\2 курс Python\lab 13\lab13>git add .  
C:\Users\User\Desktop\2 курс Python\lab 13\lab13>git commit -m "new"  
[main 08c8074] new  
4 files changed, 252 insertions(+), 3 deletions(-)  
create mode 100644 indiv.py  
create mode 100644 primer.py  
create mode 100644 zadanie.py  
  
C:\Users\User\Desktop\2 курс Python\lab 13\lab13>git push  
Enumerating objects: 8, done.  
Counting objects: 100% (8/8), done.  
Delta compression using up to 8 threads  
Compressing objects: 100% (6/6), done.  
Writing objects: 100% (6/6), 3.74 KiB | 3.74 MiB/s, done.  
Total 6 (delta 0), reused 0 (delta 0), pack-reused 0  
To https://github.com/aikanyshkaukanbekova/lab13.git  
cd219cb..08c8074 main -> main  
  
C:\Users\User\Desktop\2 курс Python\lab 13\lab13>
```

Рисунок 8. Сохранения

Контрольные вопросы:

1. Какие аргументы называются позиционными в Python?

Это аргументы, передаваемые в вызов в определённой последовательности (на определённых позициях), без указания их имён. Элементы объектов, поддерживающих итерирование, могут использоваться в качестве позиционных аргументов, если их распаковать при помощи *.

2. Какие аргументы называются именованными в Python?

Это аргументы, передаваемые в вызов при помощи имени (идентификатора), либо словаря с его распаковкой при помощи **.

3. Для чего используется оператор *?

Функция также может принимать переменное количество позиционных аргументов, тогда перед именем ставится *.

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

Каждая из этих конструкций используется для распаковки аргументов соответствующего типа, позволяя вызывать функции со списком аргументов переменной длины.