

**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 №2.11

Дисциплина: «Функции с переменным числом параметров в Python»

Тема: «Замыкания в языке Python»

Выполнил: студент 1 курса

группы ИВТ-б-о-21-1

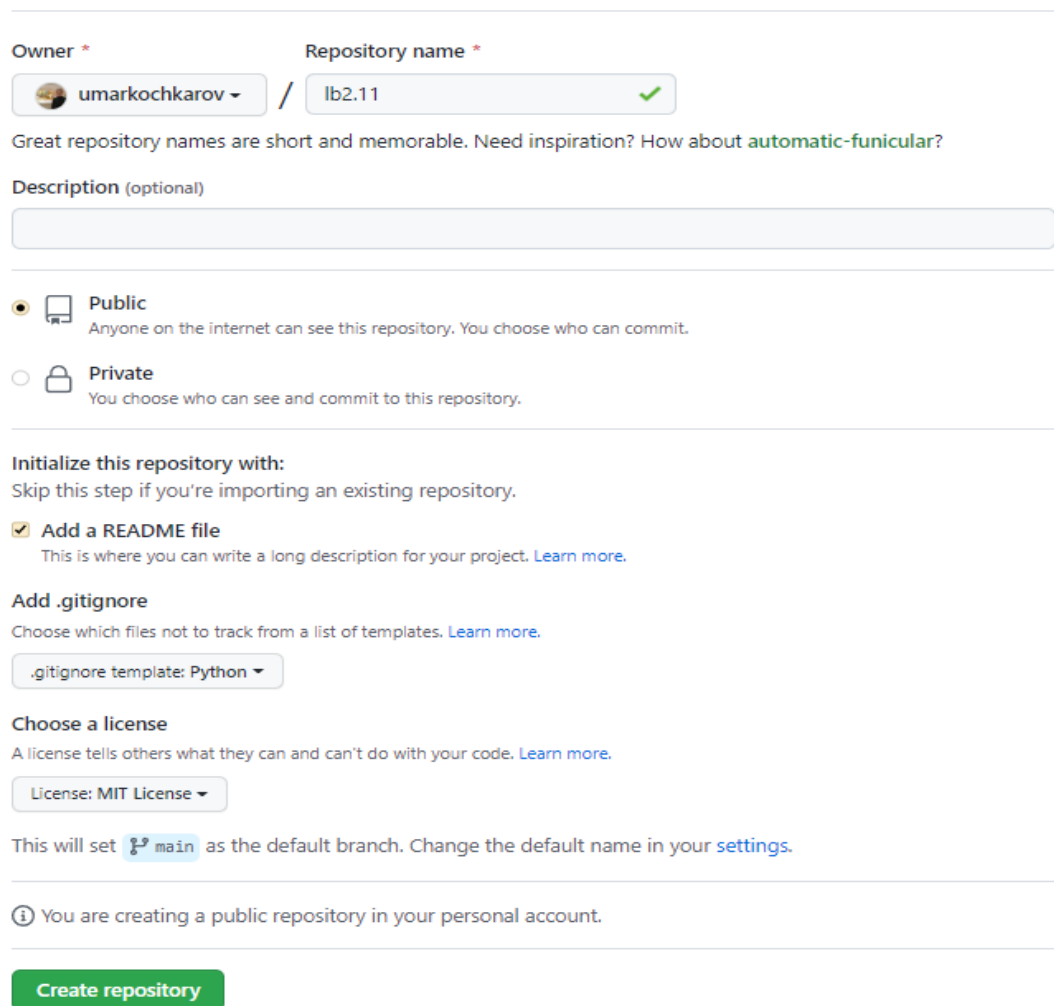
Кочкаров Умар Ахматович

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.

Ход работы

1. Создать общедоступный репозиторий с лицензией MIT и языком Python.



Owner * Repository name *

umarkochkarov / lb2.11 ✓

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

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:
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](#).

Create repository

Рисунок 1. Создание репозитория

2. Клонировать репозиторий на ПК:

```
erken@LAPTOP-ESTC60GF MINGW64 ~/Desktop/python/Ла62.11
$ git clone https://github.com/umarkochkarov/lb2.11.git
Cloning into 'lb2.11'...
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.
```

Рисунок 2. Клонирование репозитория

3. Организовать репозиторий в соответствии с моделью ветвления git-flow.

```
erken@LAPTOP-ESTC60GF MINGW64 ~/Desktop/python/Ла62.11/1b2.11 (main)
$ 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/erken/Desktop/python/Ла62.11/1b2.11/.git/hooks]
```

Рисунок 3. Организация репозитория в соответствии с моделью git-flow

4. Проработка примеров из лабораторной работы

```
1 ▶ 1 #!/usr/bin/env python3
2 2 # -*- coding: utf-8 -*-
3
4
5 5 def mul(a):
6 6     def helper(b):
7 7         return a * b
8 8     return helper
9
10
11 ▶ 11 if __name__ == '__main__':
12 12     print(mul(5)(2))
```

PC Run - 2.11

Run: 1 ×

C:\Users\erken\Desktop\python\Примеры\2.11\venv\Scripts\python.exe 10

Рисунок 4. Результат выполнения программы 1

```
1 ▶ #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4
5 def fun1(a):
6     x = a * 3
7
8     def fun2(b):
9         nonlocal x
10        return b + x
11    return fun2
12
13
14 ▶ if __name__ == "__main__":
15     test_fun = fun1(4)
16     print(test_fun(7))
```

PC Run - 2.11

Run: 2 ×

C:\Users\erken\Desktop\python\Примеры\2.11\venv\S
19

Рисунок 5. Результат выполнения программы 2

5. Выполнение индивидуального задания

```
1 ▶ #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4 def vn():
5     def fun(a, b):
6         return "Для значений а, б функция f(a,b) = ", a*b/2
7     return fun
8
9 ▶ if __name__ == '__main__':
10     A = float(input("Введите а: "))
11     B = float(input("Введите б: "))
12     f = vn()
13     print(f(A,B))
```

PC Run - 2.11

Run: main ×

C:\Users\erken\Desktop\python\Примеры\2.11\venv\Scripts\python.exe C:/User
Введите а: 10
Введите б: 22
('Для значений а, б функция f(a,b) = ', 110.0)

Рисунок 6. Результат выполнения программы индивидуального задания

Ответы на контрольные вопросы

1. Что такое замыкание?

Замыкание — это функция, в теле которой присутствуют ссылки на переменные, объявленные вне тела этой функции.

2. Как реализованы замыкания в языке программирования Python?

Замыканием в языке Python называется функция, вложенная в другую функцию и использующая переменные внешней функции.

3. Что подразумевает под собой область видимости Local?

Переменные с областью видимости Local (локальные переменные) могут быть использованы только внутри того блока кода, где она была объявлена.

4. Что подразумевает под собой область видимости Enclosing?

Для вложенных функций переменные из функции более высокого уровня

имеют данную область видимости.

5. Что подразумевает под собой область видимости Global?

Область видимости Global означает, что данная переменная может быть использована (видна) во всём модуле (файле с расширением .py).

6. Что подразумевает под собой область видимости Build-in?

Это переменный уровень интерпретатора. Для их использования не нужно импортировать модули.