

**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 №2.8

Дисциплина: «Программирование на Python»

Тема: «Работа с функциями в языке Python»

Выполнил: студент 2 курса

группы ИВТ-б-о-21-1

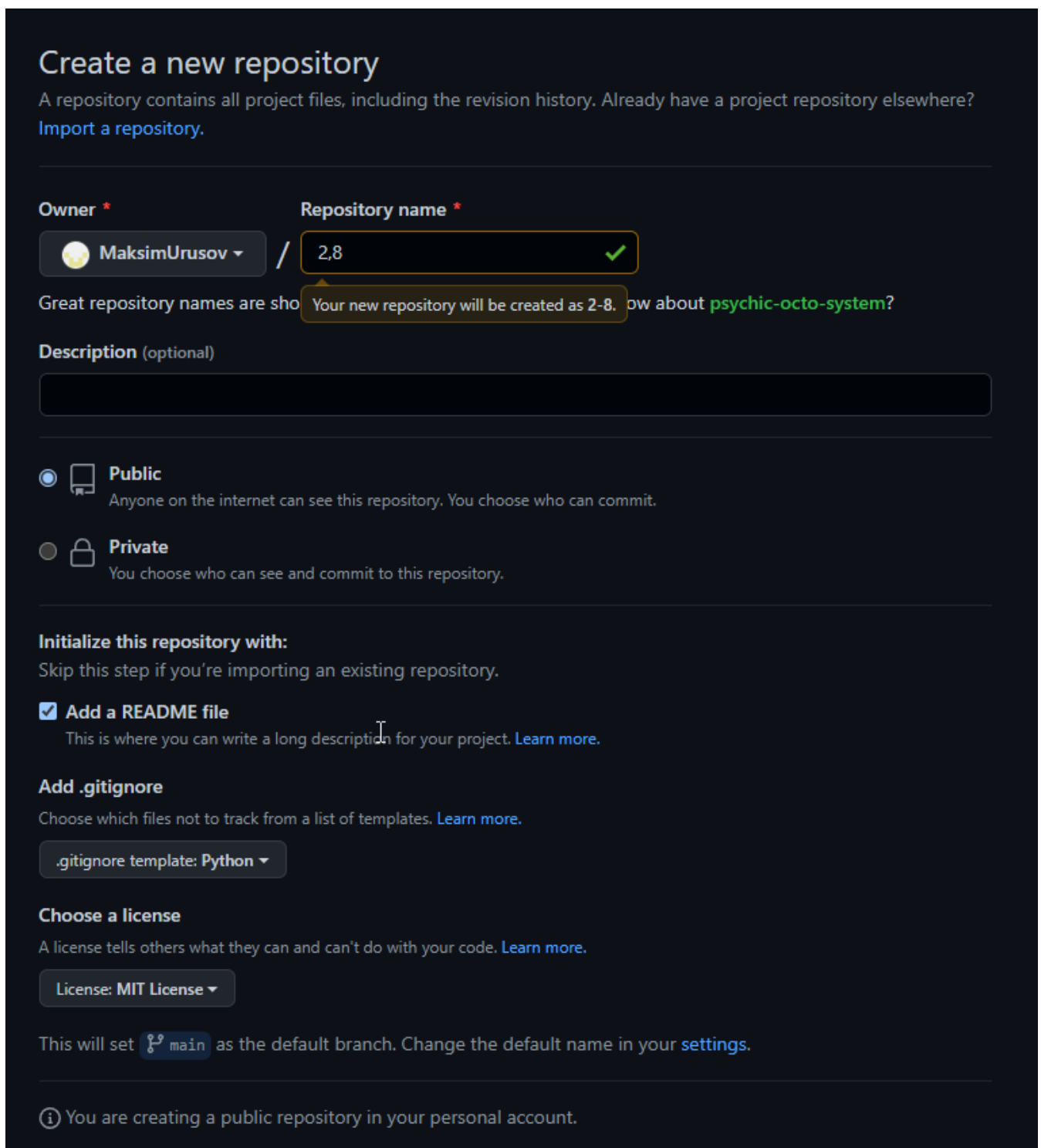
Урусов Максим Андреевич

Ставрополь 2022

Выполнение работы:

1. Создал репозиторий в GitHub «rep 2.6» в который добавил .gitignore, который дополнил правила для работы с IDE PyCharm с ЯП Python, выбрал лицензию MIT, клонировал его на лок. сервер и организовал в соответствии с моделью ветвления git-flow.



Рисунок 1.1 Создание репозитория



Create a new repository


A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)


Owner * **Repository name ***

 MaksimUrusov / 

Great repository names are short and lowercase. Your new repository will be created as 2-8. How about [psychic-octo-system?](#)

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.](#)

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

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.

```
C:\Users\adamkh\Desktop\3 семестр\Программирование на Python\2.8>git clone https://github.com/AdamKh/2.8.git
Cloning into '2.8'...
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (4/4), done.
```

Рисунок 1.2 Клонирование репозитория

```
C:\Users\adamkh\Desktop\3 семестр\Программирование на Python\2.8>git flow init
Initialized empty Git repository in C:/Users/adamkh/Desktop/3 семестр/Программирование на Python/2.8/.git/
No branches exist yet. Base branches must be created now.
Branch name for production releases: [master]
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/adamkh/Desktop/3 семестр/Программирование на Python/2.8/.git/hooks]
```

Рисунок 1.3 Организация репозитория в соответствии с моделью ветвления

git-flow



```
.gitignore – Блокнот
Файл Правка Формат Вид Справка
.idea/
# Created by https://www.toptal.com/developers/gitignore/api/python,pycharm
# Edit at https://www.toptal.com/developers/gitignore?templates=python,pycharm

### PyCharm ###
# Covers JetBrains IDEs: IntelliJ, RubyMine, PhpStorm, AppCode, PyCharm, CLion, Android Studio,
# Reference: https://intellij-support.jetbrains.com/hc/en-us/articles/206544839

# User-specific stuff
.idea/**/workspace.xml
.idea/**/tasks.xml
.idea/**/usage.statistics.xml
.idea/**/dictionaries
.idea/**/shelf

# AWS User-specific
.idea/**/aws.xml
```

Рисунок 1.4 Изменение .gitignore

2. Создал проект PyCharm в папке репозитория и проработал примеры.

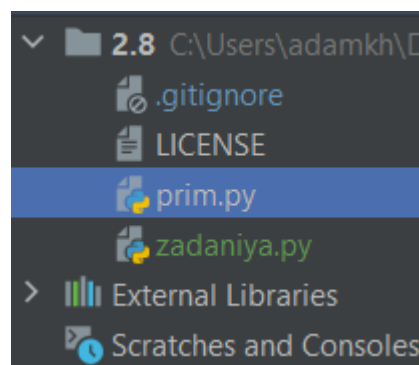


Рисунок 2.1 Создание проекта prim в PyCharm

```
>>> add
Фамилия и инициалы: asd
Должность: asd
Год поступления: 1234
>>> list
```

№	Ф.И.О.	Должность	Год
1	asd	asd	1234

Рисунок 2.2 Рез-т выполнения программы

3. Выполнил задания.

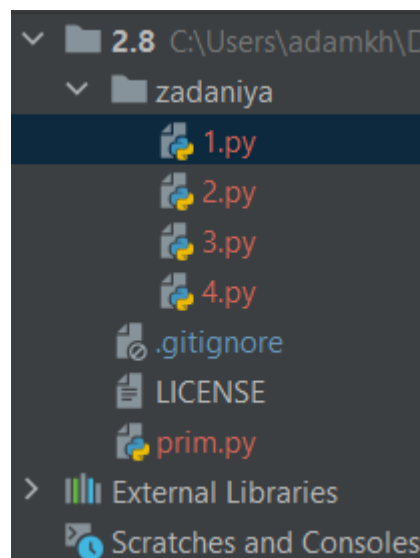


Рисунок 3.1 Создание проекта zadaniya в PyCharm

```
Введите число: 12
Число положительное

Process finished with exit code 0
```

Рисунок 3.2 Рез-т выполнения программы 1

4. (16 вариант). Выполнил индивидуальное задание.

```

help - список всех команд
>>> add
Фамилия: fa
Имя: hd
Знак Зодиака: rak
Введите дату рождения (dd/mm/yyyy)
29/06/2003
>>> list
+-----+-----+-----+-----+
| № |          Фамилия и имя          |      Знак Зодиака      |   Дата рождения   |
+-----+-----+-----+-----+
|  1 | fa          hd          |      rak      | 2003-06-29      |
+-----+-----+-----+-----+
>>> select rak
+-----+-----+-----+-----+
| № |          Фамилия и имя          |      Знак Зодиака      |   Дата рождения   |
+-----+-----+-----+-----+
|  1 | fa          hd          |      rak      | 2003-06-29      |
+-----+-----+-----+-----+
>>> add
Фамилия: fa
Имя: hd
Знак Зодиака: ribi
Введите дату рождения (dd/mm/yyyy)
17/03/2000
>>> list
+-----+-----+-----+-----+
| № |          Фамилия и имя          |      Знак Зодиака      |   Дата рождения   |
+-----+-----+-----+-----+
|  1 | fa          hd          |      ribi      | 2000-03-17      |
|  2 | fa          hd          |      rak      | 2003-06-29      |
+-----+-----+-----+-----+
>>> select rak
+-----+-----+-----+-----+
| № |          Фамилия и имя          |      Знак Зодиака      |   Дата рождения   |
+-----+-----+-----+-----+
|  1 | fa          hd          |      rak      | 2003-06-29      |
+-----+-----+-----+-----+
>>> exit

```

Рисунок 4.1 Вывод программы индивидуального задания

5. Сделал коммит, выполнил слияние с веткой main, и запустил изменения в уд. репозиторий.

```
D:\учёба\пнп\2.8>git push origin develop
Enumerating objects: 16, done.
Counting objects: 100% (16/16), done.
Delta compression using up to 8 threads
Compressing objects: 100% (13/13), done.
Writing objects: 100% (13/13), 6.97 KiB | 6.97 MiB/s, done.
Total 13 (delta 3), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (3/3), done.
remote:
remote: Create a pull request for 'develop' on GitHub by visiting
remote:   https://github.com/LenkaGolovach/2.8/pull/new/develop
remote:
To https://github.com/LenkaGolovach/2.8.git
 * [new branch]      develop -> develop
```

Рисунок 4.1 коммит и пуш изменений

```
D:\учёба\пнп\2.8>git checkout main
Switched to branch 'main'
Your branch is ahead of 'origin/main' by 2 commits.
  (use "git push" to publish your local commits)

D:\учёба\пнп\2.8>git merge develop
Updating 362c36b..12c0d9c
Fast-forward
 ind.py | 135 ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
 1 file changed, 135 insertions(+)
 create mode 100644 ind.py
```

Рисунок 4.2 Переход на ветку main и слияние ветки main с develop

```
D:\учёба\пнп\2.8>git push
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/LenkaGolovach/2.8.git
6ff1b94..12c0d9c main -> main
```

Рисунок 4.3 Пуш изменений на удаленный сервер

MaksimUrusov ind1			a097346 3 days ago	5 commits
doc	3		8 days ago	
.gitignore	test		8 days ago	
I N D.py	ind		3 days ago	
LICENSE	Initial commit		9 days ago	
README.md	Initial commit		9 days ago	
primer.py	test		8 days ago	
zadanie.py	test		8 days ago	

Рисунок 4.4 Изменения на удаленном сервере

Контр. вопросы и ответы на них:

Ответы на вопросы:

1. Каково назначение функций в языке программирования Python?

Главной задачей функций в Python, как и в других языках программирования, является сокращение объёма кода и его структуризация. В функции, как правило, выносятся те части кода, которые выполняются в программе многократно.

2. Каково назначение операторов def и return?

Оператор def необходим для определения функции. После него идёт название самой функции, передаваемые в функцию параметры и само тело функции. Оператор return служит для возвращения результата выполнения функции в основную программу, где эта функция была вызвана.

3. Каково назначение локальных и глобальных переменных при написании функций Python?

Локальные переменные существуют только внутри функции. В другой части программы как-либо вызывать или изменить их невозможно.

Глобальные напротив – существуют во всей программе.

4. Как вернуть несколько значений из функции Python?

После оператора `return` необходимо записать все возвращаемые переменные через запятую, а при вызове функции нужно задать необходимое количество переменных. Куда будут возвращены параметры.

5. Какие существуют способы передачи значений в функцию?

По ссылке и по значению.

6. Как задать значение аргументов функции по умолчанию?

Нужно в скобках передаваемых параметров присвоить им значение.

7. Каково назначение lambda-выражений в языке Python?

Lambda-выражения – это небольшие функции, которые вызываются в программе один раз.

8. Как осуществляется документирование кода согласно PEP257?

Если пояснение функции содержит одну строку, то достаточно двух кавычек с каждой стороны строки. Пример: `"""Пояснение"""`. Если это многострочное пояснение, то необходимо три кавычки с каждой стороны. Пояснение находится в теле функции, сразу после её объявления.