

**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

Хашиев Адам Мухарбекович

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.

Owner \*      Repository name \*

AdamKh / 2.8 ✓

Great repository names are short and memorable. Need inspiration? How about **super-tribble**?

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 ▾

*i* You are creating a public repository in your personal account.

**Create repository**

Рисунок 1.1 Создание репозитория

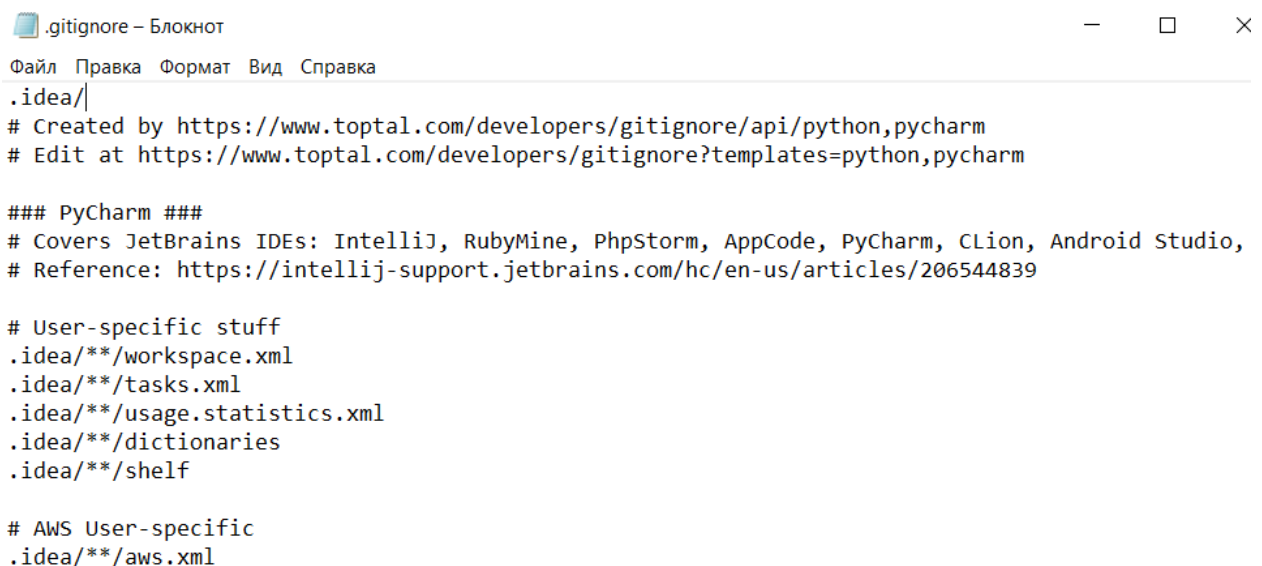
```
C:\Users\adamkh\Desktop\3 семестр\Программирование на Python\2.8>git clone https://github.com/AdamKh/2.8.git
Cloning into '2.8'...
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 4 (delta 0), reused 0 (delta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



```
.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.

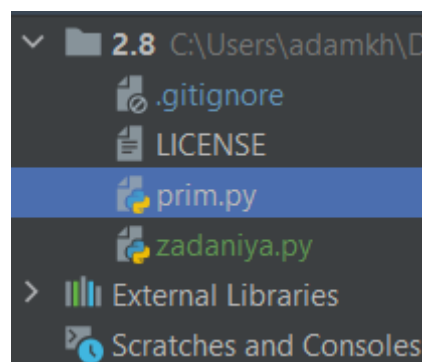


Рисунок 2.1 Создание проекта prim в PyCharm

```
>>> 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.

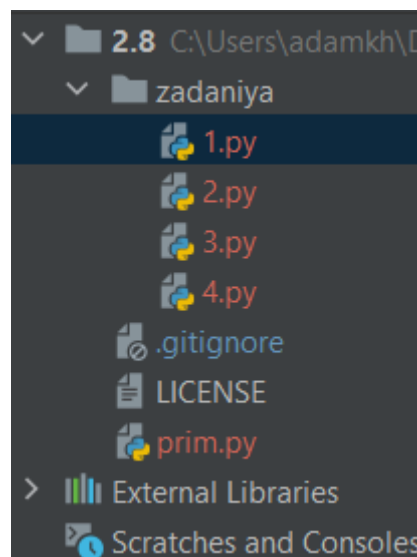


Рисунок 3.1 Создание проекта zadaniya в PyCharm

```
Введите число: 12
Число положительное

Process finished with exit code 0
```

Рисунок 3.2 Рез-т выполнения программы 1

4. (15 вариант). Выполнил индивидуальное задание.

```

help - список всех команд
>>> рудз
>>> Неизвестная команда рудз
help
Список команд:

add - добавить человека;
list - вывести список людей;
help - список всех команд;
exit - завершить работу с программой.
>>> add
Фамилия: asd
Имя: qwe
Знак Зодиака: ert
Введите дату выпуска (dd/mm/yyyy)
11/11/1111
>>> list
+-----+-----+-----+-----+
| № |          Фамилия и имя          |      Знак Зодиака      |   Дата рождения   |
+-----+-----+-----+-----+
|  1 |      asd          qwe          |      ert              |   1111-11-11      |
+-----+-----+-----+-----+
>>> |

```

Рисунок 4.1 Вывод программы индивидуального задания

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

```

C:\Users\adamkh\Desktop\3 семестр\Программирование на Python\2.8\2.8>git add .
C:\Users\adamkh\Desktop\3 семестр\Программирование на Python\2.8\2.8>git commit -m "added pr"
[main 1f54bf3] added pr
7 files changed, 557 insertions(+), 3 deletions(-)
create mode 100644 ind.py
create mode 100644 prim.py
create mode 100644 zadaniya/1.py
create mode 100644 zadaniya/2.py
create mode 100644 zadaniya/3.py
create mode 100644 zadaniya/4.py
C:\Users\adamkh\Desktop\3 семестр\Программирование на Python\2.8\2.8>git push
Enumerating objects: 12, done.
Counting objects: 100% (12/12), done.
Delta compression using up to 8 threads
Compressing objects: 100% (10/10), done.
Writing objects: 100% (10/10), 6.83 KiB | 2.28 MiB/s, done.
Total 10 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), done.
To https://github.com/AdamKh/2.8.git
 a98eb31..1f54bf3  main -> main

```

Рисунок 4.1 коммит и пуш изменений

```

C:\Users\adamkh\Desktop\3 семестр\Программирование на Python\2.8\2.8>git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
C:\Users\adamkh\Desktop\3 семестр\Программирование на Python\2.8\2.8>git merge develop
Already up to date.

```

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

```
C:\Users\adamkh\Desktop\3 семестр\Программирование на Python\2.8\2.8>git push --set-upstream origin develop
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'develop' on GitHub by visiting:
remote:   https://github.com/AdamKh/2.8/pull/new/develop
remote:
To https://github.com/AdamKh/2.8.git
 * [new branch]      develop -> develop
branch 'develop' set up to track 'origin/develop'.
```

Рисунок 4.3 Пуш изменений на удаленный сервер

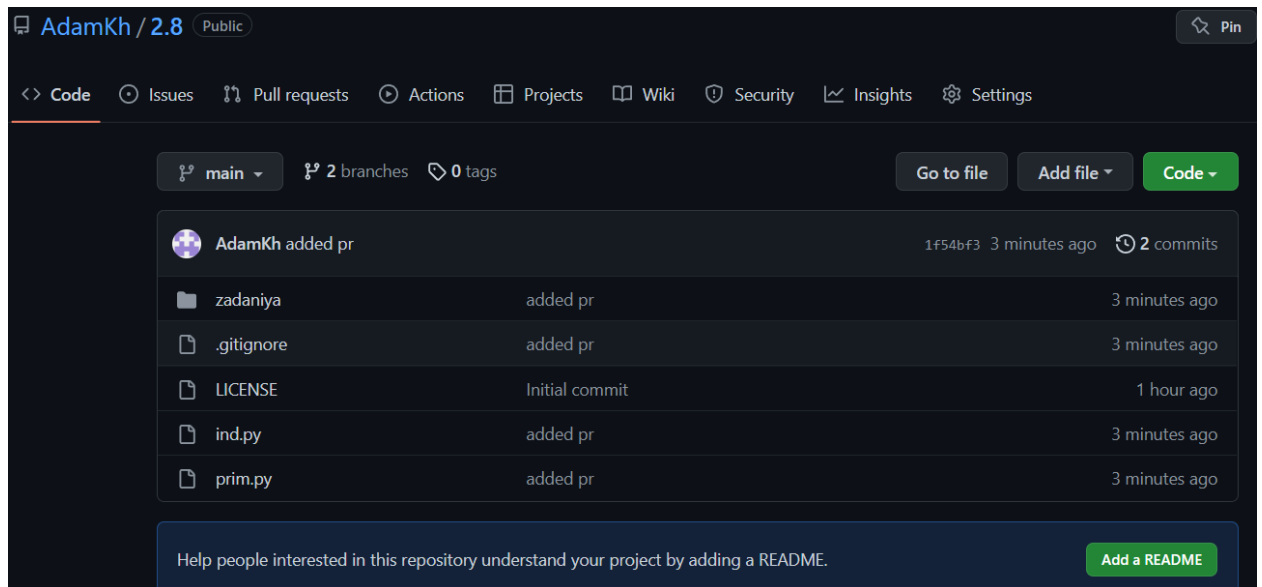


Рисунок 4.4 Изменения на удаленном сервере

### 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.