

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 №2.13

Дисциплина: «Программирование на Python»

Тема: «Модули и пакеты»

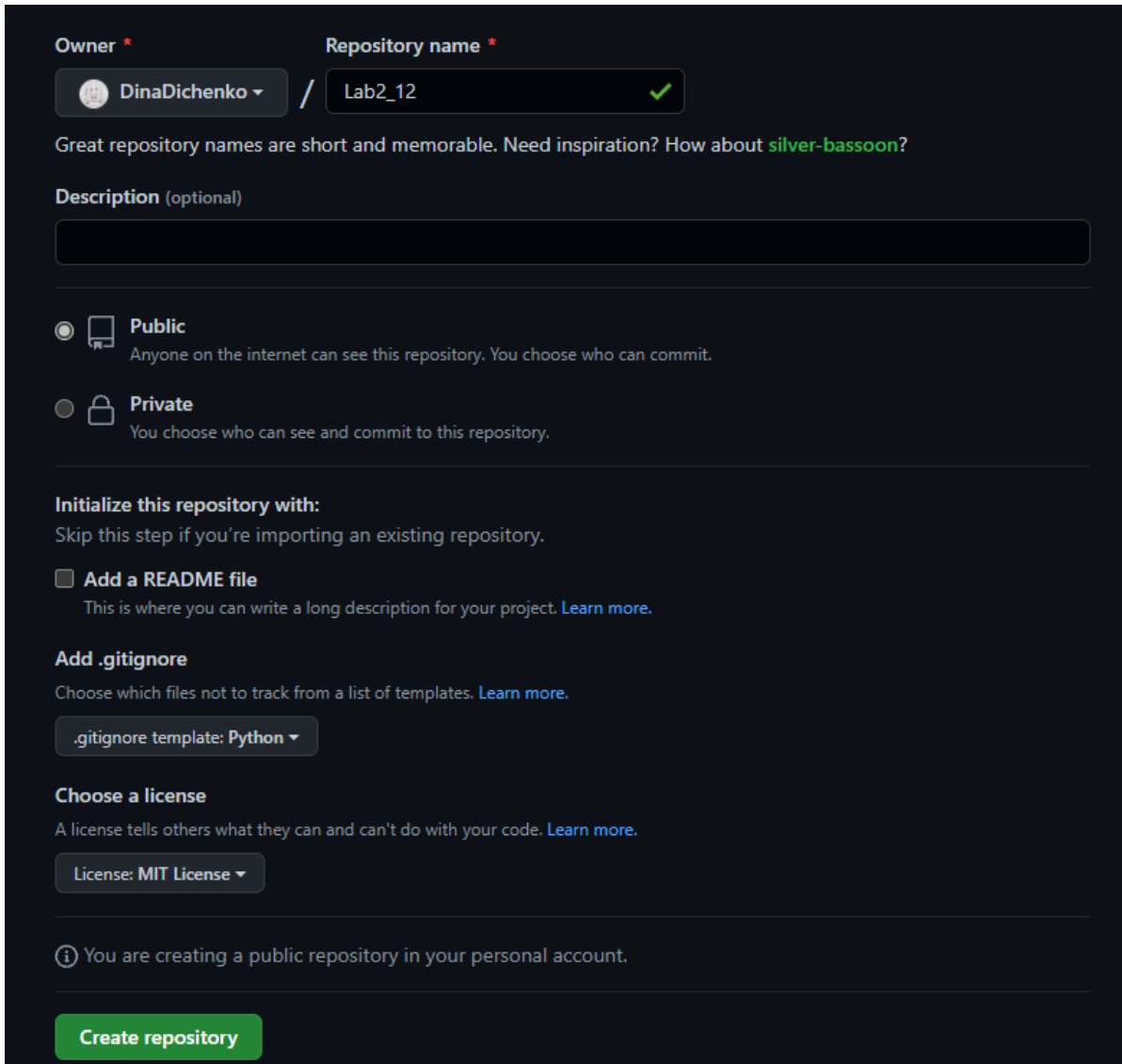
Вариант 8

Выполнила: студентка 2
курса, группы ИВТ-б-о-21-1
Диченко Дина Алексеевна

Ставрополь 2022

Цель работы: приобретение навыков по работе с модулями и пакетами языка программирования Python версии 3х.

Практическая часть:



The screenshot shows the GitHub 'Create new repository' page. At the top, the 'Owner' is set to 'DinaDichenko' and the 'Repository name' is 'Lab2_12', which is marked as valid with a green checkmark. Below this, a message suggests that great repository names are short and memorable, with an example 'silver-bassoon?'. A 'Description (optional)' text area is present. Under the 'Visibility' section, 'Public' is selected, indicating that anyone on the internet can see the repository. The 'Initialize this repository with:' section includes a checkbox for 'Add a README file' and a dropdown for '.gitignore template: Python'. The 'Choose a license' section has a dropdown for 'License: MIT License'. A note at the bottom states, 'You are creating a public repository in your personal account.' A green 'Create repository' button is at the bottom left.

Owner * Repository name *

DinaDichenko / Lab2_12 ✓

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

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. Создание репозитория

```
Cloning into 'Lab2_13'...
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 2. Клонирование репозитория

```
# 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, WebStorm and Rider
# Reference: https://intellij-support.jetbrains.com/nc/en-us/articles/206544839
.idea/
.idea

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

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

# Generated files
.idea/**/contentModel.xml

# Sensitive or high-churn files
.idea/**/dataSources/
.idea/**/dataSources.ids
.idea/**/dataSources.local.xml
.idea/**/sqlDataSources.xml
```

Рисунок 3. Изменение файла .gitignore

```
C:\Users\super\Desktop\Dina\ВУЗ\Программирование на python\Lab2_13>git flow init

Which branch should be used for bringing forth production releases?
- develop
- main
Branch name for production releases: [main] main

Which branch should be used for integration of the "next release"?
- develop
Branch name for "next release" development: [develop] dev
Fatal: Local branch 'dev' does not exist.

C:\Users\super\Desktop\Dina\ВУЗ\Программирование на python\Lab2_13>git flow init

Which branch should be used for integration of the "next release"?
- develop
Branch name for "next release" development: [develop] develop

How to name your supporting branch prefixes?
Feature branches? [feature/] fra
Bugfix branches? [bugfix/] bug
Release branches? [release/] rea
Hotfix branches? [hotfix/] hot
Support branches? [support/] sup
Version tag prefix? [] ver
Hooks and filters directory? [C:/Users/super/Desktop/Dina/ВУЗ/Программирование на python/Lab2_13/.git/hooks] hook
```

Рисунок 4. Организация репозитория в соответствии с git-flow

Проработала примеры:

```
PS C:\Users\super> & C:/Users/super/AppData/Local/Programs/Python/Python39-64/Scripts/python.exe C:/Users/super/Desktop/Dina/ВУЗ/Программирование на python/Lab2_13/prog/primer1.py
-0.9999987317275395
PS C:\Users\super>
```

Рисунок 5. Результат работы примера 1

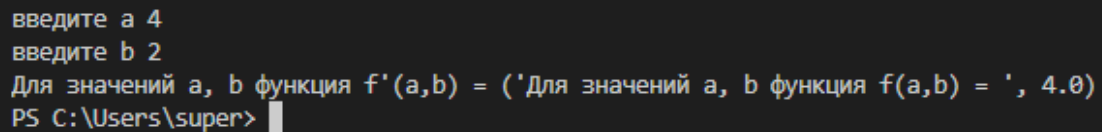
```
PS C:\Users\super> & C:/Users/super/AppData/Local/Programs/Python/Python39-64/Scripts/python.exe C:/Users/super/Desktop/Dina/ВУЗ/Программирование на python/Lab2_13/prog/primer2.py
24
PS C:\Users\super>
```

Рисунок 6. Результат работы примера 2

Выполнила индивидуальное задание:

Задание 1

Выполнить индивидуальное задание лабораторной работы 2.11, оформив все функции программы в виде отдельного модуля. Разработанный модуль должен быть подключен в основную программу с помощью одного из вариантов команды `import`. Номер варианта уточнить у преподавателя.



```
введите a 4
введите b 2
Для значений a, b функция f'(a,b) = ('Для значений a, b функция f(a,b) = ', 4.0)
PS C:\Users\super>
```

Рисунок 7. Результат работы индивидуального задания 1

Задание 2

Выполнить индивидуальное задание лабораторной работы 2.8, оформив все классы программы в виде отдельного пакета. Разработанный пакет должен быть подключен в основную программу с помощью одного из вариантов команды `import`. Настроить соответствующим образом переменную `__all__` в файле `__init__.py` пакета. Номер варианта уточнить у преподавателя.

```

>>> add
Название пункта назначения? lil
Номер поезда? 1
Введите время отправления (чч:мм)
12:34
>>> add
Название пункта назначения? ner
Номер поезда? 2
Введите время отправления (чч:мм)
16:08
>>> list
+-----+-----+-----+-----+
| No |      Название пункта      |      Номер поезда      |      Время      |
+-----+-----+-----+-----+
|  1 | lil                        |  1                      | 12:34:00        |
|  2 | ner                        |  2                      | 16:08:00        |
+-----+-----+-----+-----+
>>> select
Введите номер поезда: 1
Название пункта: lil
Время отправления: 12:34:00
>>> select 5
Введите номер поезда: 5
Поезда с таким номером нет
>>> help
Список команд:

add - добавить поезд;
list - вывести список поездов;
select <номер> - запросить поезд по номеру;
help - отобразить справку;
exit - завершить работу с программой.
>>> exit
PS C:\Users\super>

```

Рисунок 8. Результат работы индивидуального задания 2

Контрольные вопросы:

1. Что является модулем языка Python?

Под модулем в Python понимается файл с расширением .py. Модули предназначены для того, чтобы в них хранить часто используемые функции, классы, константы и т. п. Можно условно разделить модули и программы: программы предназначены для непосредственного запуска, а модули для импортирования их в другие программы. Стоит заметить, что модули могут быть написаны не только на языке Python, но и на других языках (например C).

2. Какие существуют способы подключения модулей в языке Python?

За один раз можно импортировать сразу несколько модулей, для этого их нужно перечислить через запятую после слова `import`. Если вы хотите задать псевдоним для модуля в вашей программе, можно воспользоваться вот таким синтаксисом: `import имя_модуля import имя_модуля1, имя_модуля2`. Используя любой из вышеперечисленных подходов, при вызове функции из импортированного модуля, вам всегда придется указывать имя модуля (или псевдоним). Для того, чтобы этого избежать делайте импорт через конструкцию `from ... import`. Для импортирования нескольких функций из модуля, можно перечислить их имена через запятую. Импортируемому объекту можно задать псевдоним. `import имя_модуля as новое_имя`.

3. Что является пакетом языка Python?

Пакет в Python – это каталог, включающий в себя другие каталоги и модули, но при этом дополнительно содержащий файл `__init__.py`. Пакеты используются для формирования пространства имен, что позволяет работать с модулями через указание уровня вложенности (через точку).

4. Каково назначение файла `__init__.py` ?

Файл `__init__.py` нужен для объявления структуры пакета

5. Каково назначение переменной `__all__` файла `__init__.py` ?

В переменную `__all__` вносятся все модули пакета

Вывод: в результате выполнения работы были приобретены навыки по работе с модулями и пакетами языка программирования Python версии 3х.