

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ**

Федеральное государственное автономное образовательное

учреждение высшего образования

**«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ
УНИВЕРСИТЕТ»**

Институт цифрового

развития Кафедра информационных систем и

технологий

Отчет по лабораторной работе №16.

Дисциплина: «Основы программной инженерии»

Выполнил:

Студент группы ПИЖ-

б-о-22-1, направление

подготовки: 09.03.04

«Программная инженерия»

ФИО: Гуртовой Ярослав

Дмитриевич

Проверил:

Богданов С.С

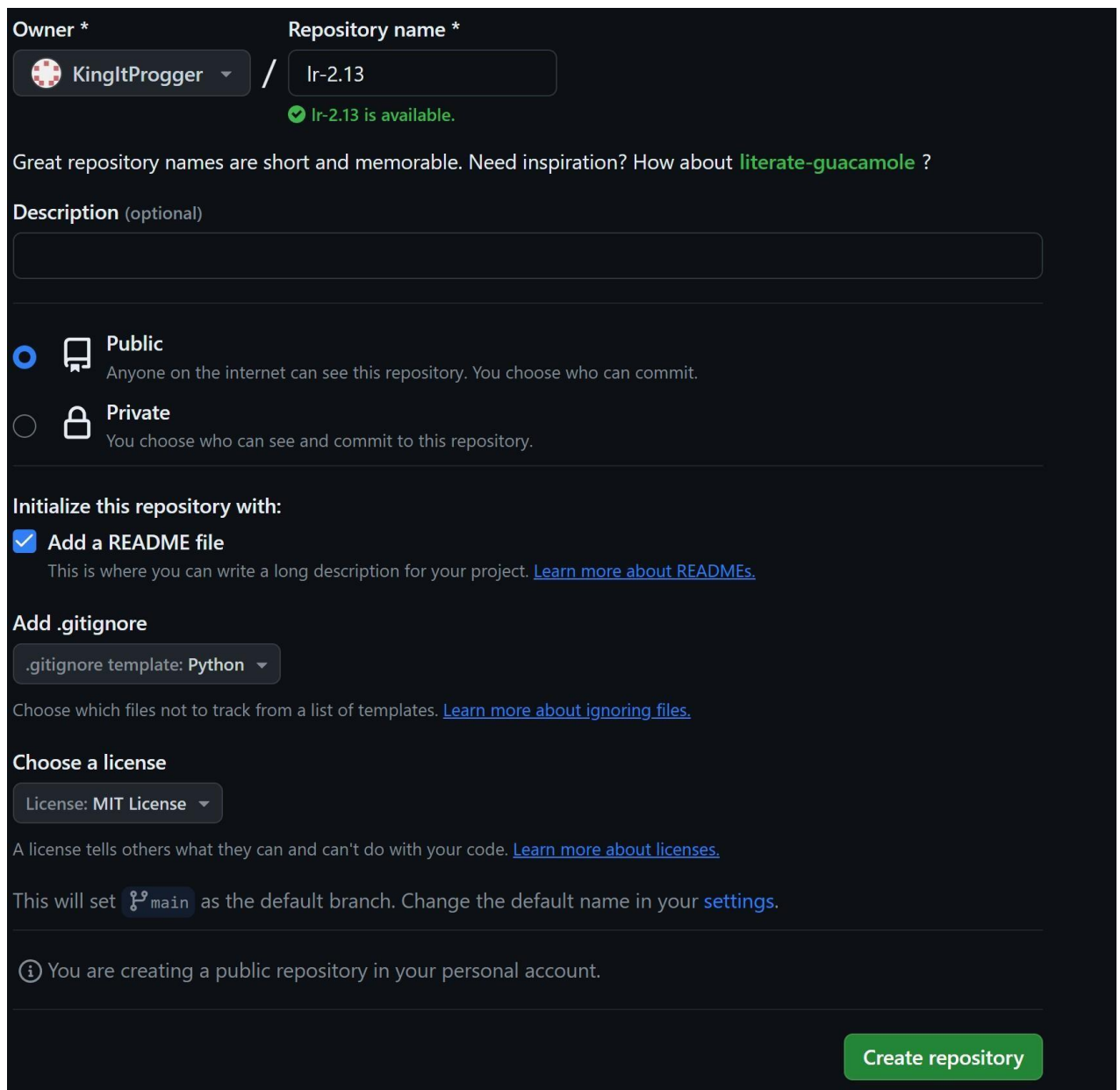
Ставрополь 2024

Тема: Модули и пакеты

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

Выполнение работы:

1. Изучил теоретический материал работы.
2. Создал репозиторий на git.hub.



The screenshot shows the GitHub 'Create repository' form. At the top, the 'Owner' is 'KingItProgger' and the 'Repository name' is 'lr-2.13'. A green checkmark indicates 'lr-2.13 is available.' Below this, a suggestion for 'literate-guacamole' is shown. The 'Description' field is empty. The 'Public' option is selected, with a note that anyone can see and commit. The 'Private' option is also visible. Under 'Initialize this repository with:', the 'Add a README file' checkbox is checked. Below this, the '.gitignore' template is set to 'Python'. The 'Choose a license' section shows 'MIT License' selected. At the bottom, a note states 'You are creating a public repository in your personal account.' and a green 'Create repository' button is visible.

Owner * / Repository name *

KingItProgger / lr-2.13

✓ lr-2.13 is available.

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

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:

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

Add .gitignore

.gitignore template: Python

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

Choose a license

License: MIT License

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

This will set `main` as the default branch. Change the default name in your [settings](#).

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

Create repository

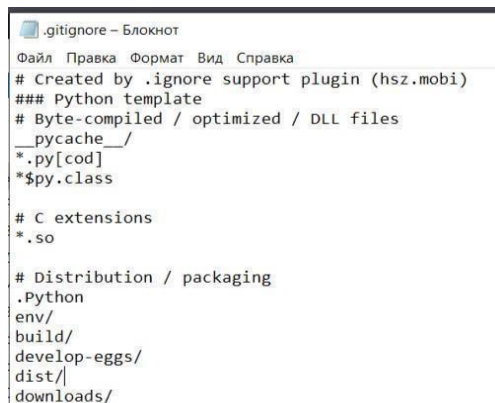
Рисунок 1. Создание репозитория

3. Клонировал созданный репозиторий

```
C:\Users\User\Desktop\учеба\4 сем\python> git clone https://github.com/KingItProgger/lr-2.13.git
Cloning into 'lr-2.13'...
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

4. Дополнил файл .gitignore необходимыми правилами



```
.gitignore - Блокнот
Файл Правка Формат Вид Справка
# Created by .ignore support plugin (hsz.mobi)
### Python template
# Byte-compiled / optimized / DLL files
__pycache__/
*.py[cod]
*$py.class

# C extensions
*.so

# Distribution / packaging
.Python
env/
build/
develop-eggs/
dist/
downloads/
```

Рисунок 3 – .gitignore для IDE PyCharm

5. Организовал репозиторий в соответствии с моделью ветвления git-flow

```
C:\Users\User\Desktop\учеба\4 сем\python\lr-2.13> git checkout -b develop
Switched to a new branch 'develop'
```

Рисунок 4. Создание ветки develop

6. Выполнил 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.

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

def create_greeting_template(template):
    def inner_function(last_name, first_name):
        formatted_template = template.replace('%F%', last_name).replace('%N%', first_name)
        return formatted_template
    return inner_function
```

Рисунок 7. Модуль greetings.py с необходимыми функциями

```
from greetings import create_greeting_template

if __name__ == "__main__":
    n = input("Введите вашу фамилию: ")
    l = input("Введите ваше имя: ")

    # Создаем замыкание с шаблоном
    greeting_template = create_greeting_template("Уважаемый %F%, %N%! Вы делаете работу по замыканиям функций.")

    # Вызываем внутреннюю функцию замыкания и отображаем результат
    result = greeting_template(n, l)
    print(result)
```

Рисунок 8. Файл general.py

```
C:\Users\User\AppData\Local\Programs\Python\Python311\python.exe "C:/Users/User/Desktop/ОПН/ЛР 2.13/general.py"
Введите вашу фамилию: Гуртовой
Введите ваше имя: Ярослав
Уважаемый Гуртовой , Ярослав! Вы делаете работу по замыканиям функций.

Process finished with exit code 0
```

Рисунок 9. Результаты работы программы

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.

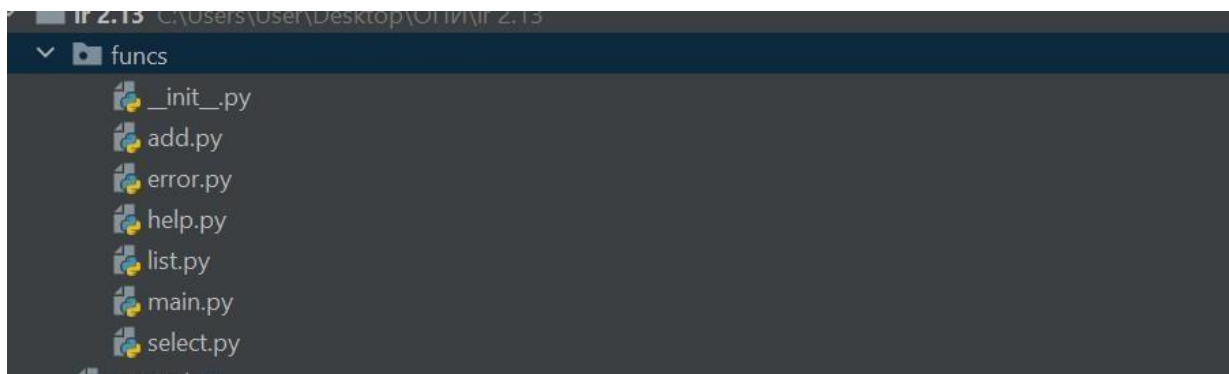


Рисунок 10. Пакетная организация задачи

```
from .add import *
from .list import *
from .help import *
from .error import *
from .select import *
from .main import *

__all__ = ['add', 'list', 'help', 'error', 'select', 'main']
```

Рисунок 11. Файл `__init.py__`

```
add - добавить рейс;
list - вывести список рейсов;
select <тип> - вывод на экран пунктов назначения и номеров рейсов для данного
help - отобразить справку;
exit - завершить работу с программой.
>>> add
Название пункта назначения рейса? moscow
Номер рейса? 777
Тип самолета? boeing
>>> add
Название пункта назначения рейса? milan
Номер рейса? 99
Тип самолета? aeroflot
>>> list
+-----+-----+-----+-----+
| 1 | milan |          | 99 |          | aeroflot |
| 2 | moscow |          | 777 |          | boeing |
+-----+-----+-----+-----+
>>> |
```

Рисунок 12. Результат работы программы

7. Зафиксировал изменения в репозитории

```

10 files changed, 167 insertions(+)
create mode 100644 pycharm/funcs/__init__.py
create mode 100644 pycharm/funcs/add.py
create mode 100644 pycharm/funcs/error.py
create mode 100644 pycharm/funcs/help.py
create mode 100644 pycharm/funcs/list.py
create mode 100644 pycharm/funcs/main.py
create mode 100644 pycharm/funcs/select.py
create mode 100644 pycharm/general.py
create mode 100644 pycharm/greetings.py
create mode 100644 pycharm/task2.py

C:\Users\User\Desktop\учеба\4 сем\python\lr-2.13>

```

Рисунок 13. Фиксация изменений

8. Слил ветки

```

C:\Users\User\Desktop\учеба\4 сем\python\lr-2.13>git merge develop
Updating f1ad260..d6059a7
Fast-forward
 pycharm/funcs/__init__.py | 8 +++++
 pycharm/funcs/add.py      | 18 ++++++
 pycharm/funcs/error.py    | 7 +++++
 pycharm/funcs/help.py     | 13 ++++++
 pycharm/funcs/list.py     | 26 ++++++
 pycharm/funcs/main.py     | 52 ++++++
 pycharm/funcs/select.py   | 20 ++++++
 pycharm/general.py        | 12 +++++
 pycharm/greetings.py      | 9 +++++
 pycharm/task2.py          | 2 ++
10 files changed, 167 insertions(+)
create mode 100644 pycharm/funcs/__init__.py
create mode 100644 pycharm/funcs/add.py
create mode 100644 pycharm/funcs/error.py
create mode 100644 pycharm/funcs/help.py
create mode 100644 pycharm/funcs/list.py
create mode 100644 pycharm/funcs/main.py
create mode 100644 pycharm/funcs/select.py
create mode 100644 pycharm/general.py
create mode 100644 pycharm/greetings.py
create mode 100644 pycharm/task2.py

C:\Users\User\Desktop\учеба\4 сем\python\lr-2.13>

```

Рисунок 14. Слияние веток

Вывод: были приобретены навыки по работе с модулями и пакетами языка программирования Python версии 3.x.

Контрольные вопросы:

1) Что является модулем языка Python?

Модуль - это файл, содержащий код на языке Python и предназначенный для использования другими программами на Python.

2) Какие существуют способы подключения модулей в языке Python?

Абсолютный импорт. При абсолютном импорте используется полный путь к желаемому (импортируемому) модулю.

Относительный импорт. Для относительного импорта используется синтаксис `from .<модуль/пакет> import <объект_импорта>`.

3) Что является пакетом языка Python?

Пакет представляет собой набор модулей Python: в то время как модуль представляет собой отдельный файл Python, пакет представляет собой каталог модулей Python, содержащий дополнительный `__init__.py` файл, чтобы отличить пакет от каталога, который просто случайно содержит кучу скриптов Python.

4) Каково назначение файла `__init__.py` ?

В общем случае файл `__init__.py` предназначен для выполнения действий по инициализации пакета, создания пространства имен для каталога и реализации поведения инструкций `from *` (то есть `from ... import *`), когда они используются для импортирования каталогов: Инициализация пакета. Когда интерпретатор Python импортирует каталог в первый раз он автоматически запускает программный код файла `__init__.py` этого каталога.

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

Python `__all__` — это переменная, которую можно установить в файле `__init__.py` пакета. Переменная `__all__` представляет собой список строк, определяющих символы, импортируемые программой. Объекты, начинающиеся с подчеркивания или не упомянутые в `__all__`, если `__all__` присутствует, не являются скрытыми; в идеале их можно увидеть и получить к ним доступ, если вы знаете их имена. `__all__` сообщает семантически «публичные» имена модуля.