

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 №2.15

Дисциплина: «Программирование на Python»

Тема: «Работа с файлами в языке Python»

Вариант 8

Выполнила: студентка 2 курса,

группы ИВТ-б-о-21-1

Диченко Дина Алексеевна

Ставрополь 2022

**Цель работы:** приобретение навыков по работе с текстовыми файлами при написании программ с помощью языка программирования Python версии 3.x, изучение основных методов модуля os для работы с файловой системой, получение аргументов командной строки.

## Практическая часть:

### 1. Создала репозиторий.

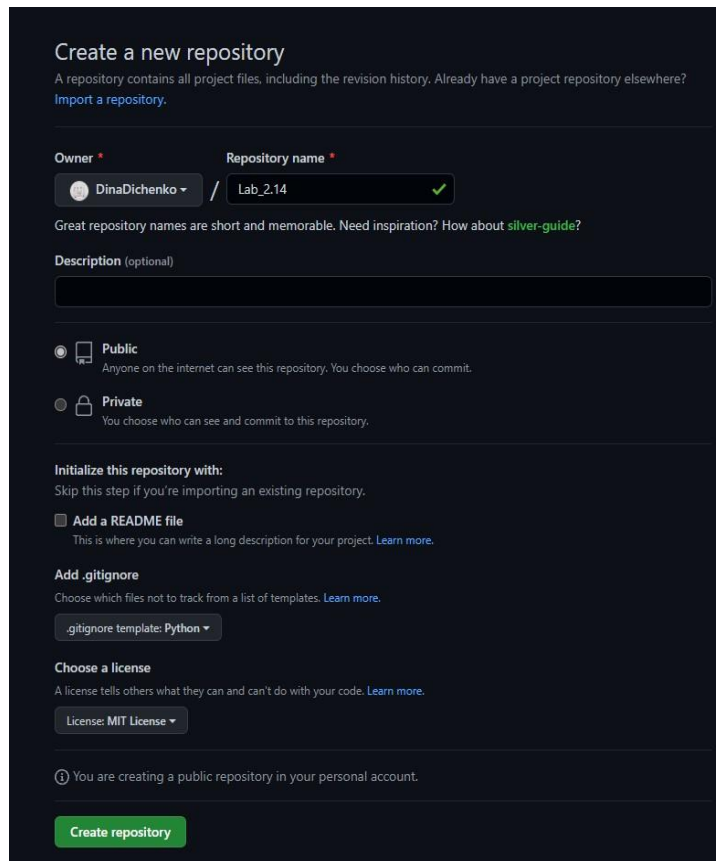
The image shows the GitHub 'Create a new repository' page. At the top, it says 'Create a new repository' and provides a brief explanation of what a repository is. Below this, there are two main input fields: 'Owner' and 'Repository name'. The 'Owner' field is set to 'DinaDichenko' and the 'Repository name' field is set to 'Lab\_2.14'. There is a green checkmark next to the repository name. Below these fields, there is a section for 'Description (optional)' with a text input area. Further down, there are two radio buttons for 'Public' and 'Private'. The 'Public' option is selected. Below this, there is a section for 'Initialize this repository with:' which includes a checkbox for 'Add a README file' and a dropdown menu for '.gitignore template' set to 'Python'. There is also a section for 'Choose a license' with a dropdown menu set to 'MIT License'. At the bottom, there is a green 'Create repository' button. A note at the bottom states 'You are creating a public repository in your personal account.'

Рисунок 1. Создание репозитория 2.

### Клонировала репозиторий.

```
C:\Users\super\Desktop\Dina\ВУЗ\Программирование на python>git clone https://github.com/DinaDichenko/Lab_2.14.git
Cloning into 'Lab_2.14'...
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.
C:\Users\super\Desktop\Dina\ВУЗ\Программирование на python>
```

Рисунок 2. Клонирование репозитория

### 3. Организовала свой репозиторий в соответствие с моделью ветвления git-flow.

```

C:\Users\super\Desktop\Dina\ВУЗ\Программирование на python\Lab_2.14>git checkout develop
Switched to branch 'develop'

C:\Users\super\Desktop\Dina\ВУЗ\Программирование на python\Lab_2.14>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] develop

How to name your supporting branch prefixes?
Feature branches? [feature/] fea
Bugfix branches? [bugfix/] bug
Release branches? [release/] rel
Hotfix branches? [hotfix/] hot
Support branches? [support/] sup
Version tag prefix? [] pre
Hooks and filters directory? [C:/Users/super/Desktop/Dina/ВУЗ/Программирование на python/Lab_2.14/.git/hooks] hook

C:\Users\super\Desktop\Dina\ВУЗ\Программирование на python\Lab_2.14>_

```

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

#### 4. Изменила файл .gitignore/

```

# Edit at https://www.toptal.com/developers/gitignore?templates=python,visualstudio

### Python ###
# Byte-compiled / optimized / DLL files
__pycache__/_
*.py[co]
*$py.class
.idea/
.idea

# C extensions
*.so
.idea/
# Distribution / packaging
.Python
build/
develop-eggs/
dist/
downloads/
eggs/
.eggs/
lib/
lib64/
parts/
sdist/
var/
wheels/
share/python-wheels/
*.egg-info/
.installed.cfg
*.egg
MANIFEST

# PyInstaller
# Usually these files are written by a python script from a template
# before PyInstaller builds the exe, so as to inject date/other infos into it.
*.manifest
*.spec

# Installer logs
pip-log.txt
pip-delete-this-directory.txt

# Unit test / coverage reports
htmlcov/
.tox/
.nox/
.coverage
.coverage.*

```

Рисунок 4. Изменение файла .gitignore

5. Проработала примеры.
6. Выполнила индивидуальные задания.

Задание 1. Написать программу, которая считывает текст из файла и выводит на экран только цитаты, то есть предложения, заключенные в кавычки.

```
цитата тут
Мастера и Маргаритты
— Вы не Достоевский, — сказала гражданка, сбиваемая с толку Коровьевым.
— Ну, почем знать, почем знать, — ответил тот.
— Достоевский умер, — сказала гражданка, но как-то не очень уверенно.
— Протестую, — горячо воскликнул Бегемот. — Достоевский бессмертен!
>>> |
```

Рисунок 5. Результат выполнения индивидуального задания 1

Задание 2. Как вы знаете, в языке Python для создания комментариев в коде используется символ `#`. Комментарий начинается с этого символа и продолжается до конца строки – без возможности остановить его раньше. В данном упражнении вам предстоит написать программу, которая будет удалять все комментарии из исходного файла с кодом на языке Python. Пройдите по всем строкам в файле на предмет поиска символа `#`. Обнаружив его, программа должна удалить все содержимое, начиная с этого символа и до конца строки. Для простоты не будем рассматривать ситуации, когда знак решетки встречается в середине строки. Сохраните новое содержимое в созданном файле. Имена файла источника и файла назначения должны быть запрошены у пользователя. Удостоверьтесь в том, что программа корректно обрабатывает возможные ошибки при работе с обоими файлами.

```

C:\Users\student-09-525\Downloads\Lab2_15\prog>python individual2.py ind2.tht
Ошибка при доступе к файлу.
C:\Users\student-09-525\Downloads\Lab2_15\prog>python individual2.py ind2.txt
Аа : 68
Бб : 13
Вв : 50
Гг : 14
Дд : 22
Ее : 79
Ёё : 7
Жж : 4
Зз : 21
Ии : 102
Йй : 26
Кк : 36
Лл : 54
Мм : 31
Нн : 67
Оо : 115
Пп : 32
Рр : 39
Сс : 46
Тт : 69
Уу : 36
Хх : 5
Цц : 7
Чч : 54
Шш : 3
Щщ : 3
Ыы : 14
Ьь : 16
Ээ : 5
Юю : 2
Яя : 14
C:\Users\student-09-525\Downloads\Lab2_15\prog>

```

Рисунок 6. Результат выполнения индивидуального задания 2

7. Самостоятельно подберите или придумайте задачу для работы с изученными функциями модуля os.

---

```

Имя текущего пользователя: student-09-525
Текущая дериктория: C:\Users\student-09-525\Downloads\Lab2_15\prog

```

Рисунок 7. Результат выполнения индивидуального задания 3

### Контрольные вопросы:

1. Как открыть файл в языке Python только для чтения?

Чтобы открыть файл для чтения, мы используем режим r. Для чтения мы воспользуемся функцией read(size), если параметр size не указан, функция вернет нам всю строку. file = open("text.txt", 'r', encoding = 'utf-8').

2. Как открыть файл в языке Python только для записи?

В Python открытие файлов выполняется с помощью функции open(), которой передается два аргумента - имя файла и режим. Файл может быть открыт в режиме чтения, записи, добавления.

3. Как прочитать данные из файла в языке Python?

Чтение данных из файла осуществляется с помощью методов `read(размер)` и `readline()`. Метод `read(размер)` считывает из файла определенное количество символов, переданное в качестве аргумента.

4. Как записать данные в файл в языке Python?

Запись данных в файл. Записать данные в файл можно с помощью метода `write()`.

5. Как закрыть файл в языке Python?

После того, как мы открыли файл, и выполнили все нужные операции, нам необходимо его закрыть. Для закрытия файла используется функция `close()`.

6. Изучите самостоятельно работу конструкции `with ... as`. Каково ее назначение в языке?

Конструкция `with ... as` используется для оборачивания выполнения блока инструкций менеджером контекста. ... Если в конструкции `with - as` было несколько выражений, то это эквивалентно нескольким вложенным конструкциям

7. Изучите самостоятельно документацию Python по работе с файлами. Какие помимо рассмотренных существуют методы записи/чтения информации из файла?

Один из самых распространенных способов вывести данные в Python – это напечатать их в консоли. Если вы находитесь на этапе изучения языка, такой способ является основным для того, чтобы быстро просмотреть результат своей работы

8. Какие существуют, помимо рассмотренных, функции модуля `os` для работы с файловой системой?

`os.chdir(path)` - смена текущей директории.

`os.chmod (path, mode, *, dir_fd=None, follow_symlinks=True)` - смена прав доступа к объекту (`mode` - восьмеричное число).

`os.chown (path, uid, gid, *, dir_fd=None, follow_symlinks=True)` - меняет id владельца и группы (Unix).

`os.getcwd()` - текущая рабочая директория.

`os.link (src, dst, *, src_dir_fd=None, dst_dir_fd=None, follow_symlinks=True)` - создаёт жёсткую ссылку.

`os.listdir (path=".")` - список файлов и директорий в папке. `os.mkdir (path, mode=0o777, *, dir_fd=None)` - создаёт директорию.

`OSError`, если директория существует.

`os.makedirs (path, mode=0o777, exist_ok=False)` - создаёт директорию, создавая при этом промежуточные директории.

`os.remove (path, *, dir_fd=None)` - удаляет путь к файлу.

`os.rename (src, dst, *, src_dir_fd=None, dst_dir_fd=None)` - переименовывает файл или директорию из `src` в `dst`.

`os.rename (old, new)` - переименовывает `old` в `new`, создавая промежуточные директории.

`os.replace (src, dst, *, src_dir_fd=None, dst_dir_fd=None)` - переименовывает из `src` в `dst` с принудительной заменой.

`os.rmdir (path, *, dir_fd=None)` - удаляет пустую директорию. `os.removedirs (path)` - удаляет директорию, затем пытается удалить родительские директории, и удаляет их рекурсивно, пока они пусты. `os.sync()` - записывает все данные на диск (Unix). `os.truncate (path, length)` - обрезает файл до длины `length`.

`os.utime (path, times=None, *, ns=None, dir_fd=None, follow_symlinks=True)` - модификация времени последнего доступа и изменения файла. Либо `times` - кортеж (время доступа в секундах, время изменения в секундах), либо `ns` - кортеж (время доступа в наносекундах, время изменения в наносекундах).

`os.walk (top, topdown=True, onerror=None, followlinks=False)` – генерация имён файлов в дереве каталогов, сверху вниз (если `topdown` равен `True`), либо снизу вверх (если `False`). Для каждого каталога функция `walk` возвращает кортеж (путь к каталогу, список каталогов, список файлов).

**Вывод:** в результате выполнения программы были приобретены навыки по работе с текстовыми файлами при написании программ с помощью языка

программирования Python версии 3.x, изучены основные методы модуля os для работы с файловой системой, получение аргументов командной строки.