

**РОССИЙСКОЙ ФЕДЕРАЦИИ**  
**Федеральное государственное автономное**  
**образовательное учреждение высшего образования**  
**«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»**

**Кафедра инфокоммуникаций**

**«Работа с файловой системой в Python3 с использованием модуля  
pathlib»**

**Отчет по лабораторной работе № 2.19**  
**по дисциплине «Программирование на Python»**

Выполнил студент группы ИВТ-б-о-22-1

Пушкин Никита.

« » \_\_\_\_\_ 2023г.

Подпись студента \_\_\_\_\_

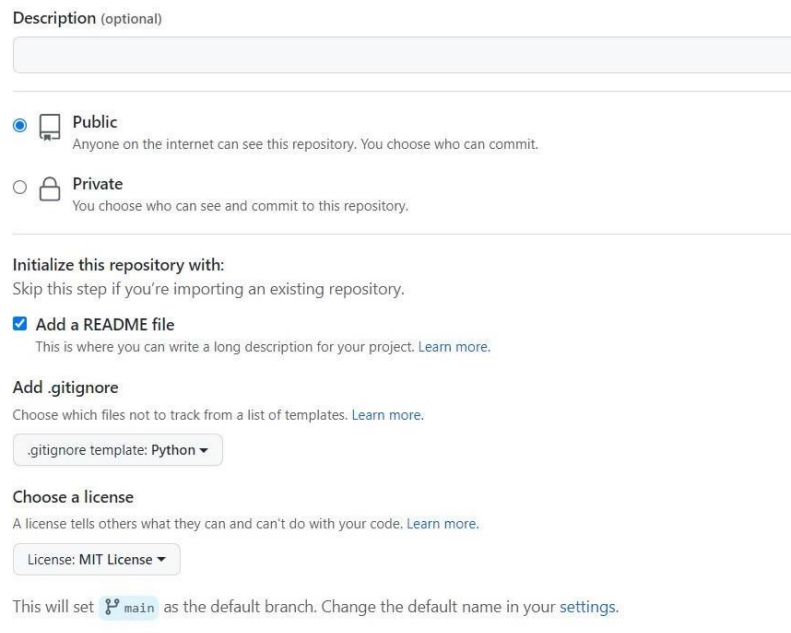
Работа защищена « » \_\_\_\_\_ 20\_\_г.

Проверил Воронкин Р.А. \_\_\_\_\_  
(подпись)

**Цель работы:** приобретение навыков по работе с файловой системой с помощью библиотеки `pathlib` языка программирования Python версии 3.x.

**Порядок выполнения работы:**

1. Создал общедоступный репозиторий на GitHub, в котором использована лицензия MIT и язык программирования Python.



The screenshot shows the GitHub repository creation interface. At the top is a text input field for the repository description, labeled 'Description (optional)'. Below this are two radio button options: 'Public' (selected) and 'Private'. The 'Public' option is accompanied by the text 'Anyone on the internet can see this repository. You choose who can commit.' The 'Private' option is accompanied by 'You choose who can see and commit to this repository.' Below these options is a section titled 'Initialize this repository with:' with the instruction 'Skip this step if you're importing an existing repository.' Under this section, the 'Add a README file' checkbox is checked, with a link to 'Learn more.' Below this is a section titled 'Add .gitignore' with the instruction 'Choose which files not to track from a list of templates. Learn more.' A dropdown menu shows '.gitignore template: Python'. Below this is a section titled 'Choose a license' with the instruction 'A license tells others what they can and can't do with your code. Learn more.' A dropdown menu shows 'License: MIT License'. At the bottom, it states 'This will set main as the default branch. Change the default name in your settings.'

Рисунок 1 - Создание репозитория

2. Выполните клонирование созданного репозитория.

```
C:\Users\User>cd C:\Users\User\Desktop\2 кypc Python\lab 22
C:\Users\User\Desktop\2 кypc Python\lab 22>git clone https://github.com/aikanyshkaukanbekova/2.19.git
Cloning into '2.19'...
remote: Enumerating objects: 8, done.
remote: Counting objects: 100% (8/8), done.
remote: Compressing objects: 100% (7/7), done.
remote: Total 8 (delta 1), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (8/8), 4.51 KiB | 1.13 MiB/s, done.
Resolving deltas: 100% (1/1), done.
C:\Users\User\Desktop\2 кypc Python\lab 22>
```

Рисунок 2 - Клонирование репозитория

3. Организуйте свой репозиторий в соответствии с моделью ветвления git-flow.

```

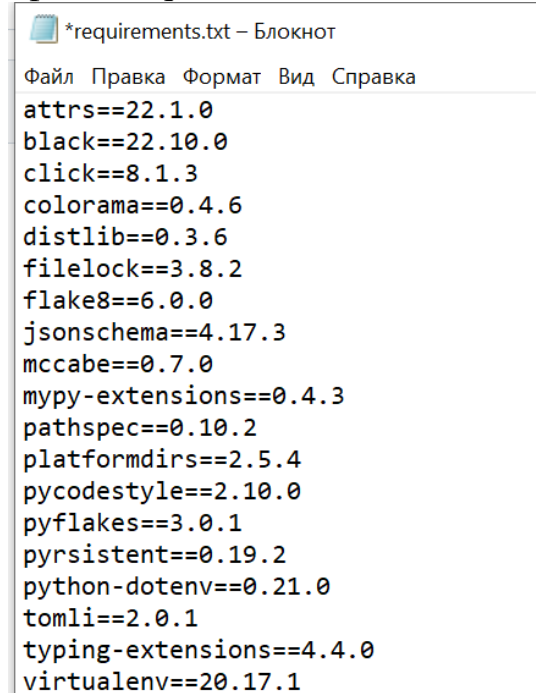
C:\Users\User\Desktop\2 курс Python\lab 10\lab-10>git flow init
which branch should be used for bringing forth production releases?
- main
Branch name for production releases: [main]
Branch name for "next release" development: [develop]

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/User/Desktop/2 курс Python/lab 10/lab-10/.git/hooks]
C:\Users\User\Desktop\2 курс Python\lab 10\lab-10>

```

Рисунок 3 - Ветвление по модели git-flow

#### 4. Формирование файла requirements.txt.



```

attrs==22.1.0
black==22.10.0
click==8.1.3
colorama==0.4.6
distlib==0.3.6
filelock==3.8.2
flake8==6.0.0
jsonschema==4.17.3
mccabe==0.7.0
mypy-extensions==0.4.3
pathspec==0.10.2
platformdirs==2.5.4
pycodestyle==2.10.0
pyflakes==3.0.1
pyrsistent==0.19.2
python-dotenv==0.21.0
tomli==2.0.1
typing-extensions==4.4.0
virtualenv==20.17.1

```

Рисунок 4 - Файл requirements.txt

#### 5. Проработать примеры лабораторной работы.

```

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

import pathlib
import collections

if __name__ == "__main__":
    print(collections.Counter(p.suffix for p in pathlib.Path.cwd().iterdir()))

```

Рисунок 5 – Результат выполнения примера 1

```

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

import pathlib

def tree(directory):
    print(f'+ {directory}')
    for path in sorted(directory.rglob('*')):
        depth = len(path.relative_to(directory).parts)
        spacer = ' ' * depth
        print(f'{spacer}+ {path.name}')

```

```
if __name__ == "__main__":
    tree(pathlib.Path.cwd())
```

Рисунок 6 - Результат выполнения примера 2

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

import pathlib
from datetime import datetime

if __name__ == "__main__":
    directory = pathlib.Path.cwd()
    time, file_path = max((f.stat().st_mtime, f) for f in directory.iterdir())
    print(datetime.fromtimestamp(time), file_path)
```

Рисунок 7 - Результат выполнения примера 3

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

import pathlib

def unique_path(directory, name_pattern):
    counter = 0
    while True:
        counter += 1
        path = directory/name_pattern.format(counter)
        if not path.exists():
            return path

if __name__ == "__main__":
    path = unique_path(pathlib.Path.cwd(), 'test{:03d}.txt')
    print(path)
```

Рисунок 8 - Результат выполнения примера 4

## 6. Выполнить индивидуальные задания.

### Задание 1

Для своего варианта лабораторной работы 2.17 добавьте возможность хранения файла данных

в домашнем каталоге пользователя. Для выполнения операций с файлами необходимо

использовать модуль `pathlib`.

```
import sys
import json
from pathlib import Path

def select(line, humans):
    # Функция выбора человека по дате рождения
    nom = input('Введите дату рождения: ')
    count = 0
    print(line)
    print(
```

```
f| {"№":^4} | {"Ф.И.О.":^20} | {"знак зодиака":^15} | {"Дата рождения":^16} |")
print(line)
```

```
for i, num in enumerate(humans, 1):
    if nom == num.get('daytime', ""):
        count += 1
    print(
        '| {:<4} | {:<20} | {:<15} | {:<16} |'.format(
            count,
            num.get('name', ""),
            num.get('zodiac', ""),
            num.get('daytime', 0)))
print(line)
```

```
if count == 0:
    print("Таких людей нет")
```

```
def table(line, humans):
    # Функция вывода списка людей
    print(line)
    print(
        '| {:^4} | {:^20} | {:^15} | {:^16} |'.format(
            "№",
            "Ф.И.О.",
            "Знак зодиака",
            "Дата рождения"))
    print(line)
    for i, num in enumerate(humans, 1):
        print(
            '| {:<4} | {:<20} | {:<15} | {:<16} |'.format(
                i,
                num.get('name', ""),
                num.get('zodiac', ""),
                num.get('daytime', 0)
            )
        )
    print(line)
```

```
def add(humans):
    # Функция добавления новых людей
    daytime = input('Введите дату рождения: ')
    zodiac = input('Введите знак зодиака: ')
    name = input('Введите Ф.И.О.: ')
    air = {
        'zodiac': zodiac,
        'name': name,
        'daytime': daytime
    }
```

```
humans.append(air)
save_to_json(Path.home() / 'humans.json', humans) # Сохраняем данные в домашнем каталоге
if len(humans) > 1:
    humans.sort(key=lambda x: x.get('daytime', ""))
```

```
def save_to_json(file_path, data):
    with open(file_path, 'w', encoding='utf-8') as file: # Добавил encoding
        json.dump(data, file, ensure_ascii=False) # Для сохранения символов не входящих в ASCII
```

```

def load_from_json(file_path):
    try:
        with open(file_path, 'r', encoding='utf-8') as file: # Добавил encoding
            return json.load(file)
    except FileNotFoundError:
        return []

def main():
    # Основная функция программы
    humans = load_from_json(Path.home() / 'humans.json') # Загружаем данные из домашнего каталога
    print('Список команд: \n exit - Завершить работу'
          '\n add - Добавить человека \n'
          '\n list - Показать список людей'
          '\n select - Выбрать знак зодиака по дате рождения')
    line = '+-{}-+-{}-+-{}-+-{}-+'.format(
        '-' * 4,
        '-' * 20,
        '-' * 15,
        '-' * 16
    )
    while True:
        com = input('Введите команду: ').lower()
        if com == 'exit':
            break
        elif com == "add":
            add(humans)
        elif com == 'list':
            table(line, humans)
        elif com == 'select':
            select(line, humans)
        else:
            print(f'Неизвестная команда {com}', file=sys.stderr)

if __name__ == '__main__':
    main()

```

### 1. Что регламентирует PEP 428?

Модуль Pathlib – Объектно-ориентированные пути файловой системы

### 2. Как осуществляется создание путей средствами модуля pathlib?

Есть несколько разных способов создания пути. Прежде всего, существуют classmethods наподобие `.cwd()` (текущий рабочий каталог) и `.home()` (домашний каталог вашего пользователя)

### 3. Как получить путь дочернего элемента файловой системы с помощью модуля pathlib?

При помощи метода `resolve()`.

#### **4. Как получить путь к родительским элементам файловой системы с помощью модуля pathlib?**

При помощи свойства `parent`.

#### **5. Как выполняются операции с файлами с помощью модуля pathlib?**

- перемещение;
- удаление файлов;
- подсчёт файлов;
- найти последний изменённый файл;
- создать уникальное имя файла;
- чтение и запись файлов.

#### **6. Как можно выделить компоненты пути файловой системы с помощью модуля pathlib?**

`.name`  
`.parent`  
`.stem`  
`.suffix`  
`.anchor`

#### **7. Как выполнить перемещение и удаление файлов с помощью модуля pathlib?**

`.replace()` – метод перемещения файлов  
`.unlink()` – метод удаления файлов

## 8. Как выполнить подсчет файлов в файловой системе?

Метод `.iterdir()`

## 9. Как отобразить дерево каталогов файловой системы?

```
def tree(directory):  
    print(f'+ {directory}')  
    for path in sorted(directory.rglob('*')):  
        depth = len(path.relative_to(directory).parts)  
        spacer = ' ' * depth  
        print(f'{spacer}+ {path.name}')
```

## 10. Как создать уникальное имя файла?

```
def unique_path(directory, name_pattern):  
    counter = 0  
    while True:  
        counter += 1  
        path = directory/name_pattern.format(counter)  
        if not path.exists():  
            return path  
    path = unique_path(pathlib.Path.cwd(), 'test{:03d}.txt')
```

## 11. Каковы отличия в использовании модуля `pathlib` для различных операционных систем?

Ранее мы отмечали, что когда мы создавали экземпляр `pathlib.Path`, возвращался либо объект `WindowsPath`, либо `PosixPath`. Тип объекта будет зависеть от операционной системы, которую вы используете. Эта функция позволяет довольно легко писать кроссплатформенный код. Можно явно запросить `WindowsPath` или `PosixPath`, но вы будете ограничивать свой код только этой системой без каких-либо преимуществ. Такой конкретный путь не может быть использован в другой системе

**Вывод:** были приобретены навыки по работе с файловой системой с помощью библиотеки `pathlib` языка программирования Python версии 3.x.