

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ
АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО
ОБРАЗОВАНИЯ «СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»
ИНСТИТУТ ЦИФРОВОГО РАЗВИТИЯ**

Отчет по лабораторной работе № 2.19

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

по дисциплине «Основы программной инженерии»

Выполнила:
Образцова Мария Дмитриевна,
2 курс, группа ПИЖ-б-о-21-1,
Проверил:
Доцент кафедры инфокоммуникаций,
Воронкин Р.А.

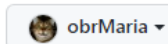
Ставрополь, 2023 г.

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

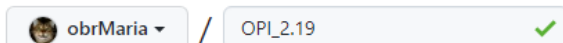
Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Owner *



Repository name *



Great repository names are short and memorable. Need inspiration? How about [probable-octo-fishstick?](#)

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** ▼

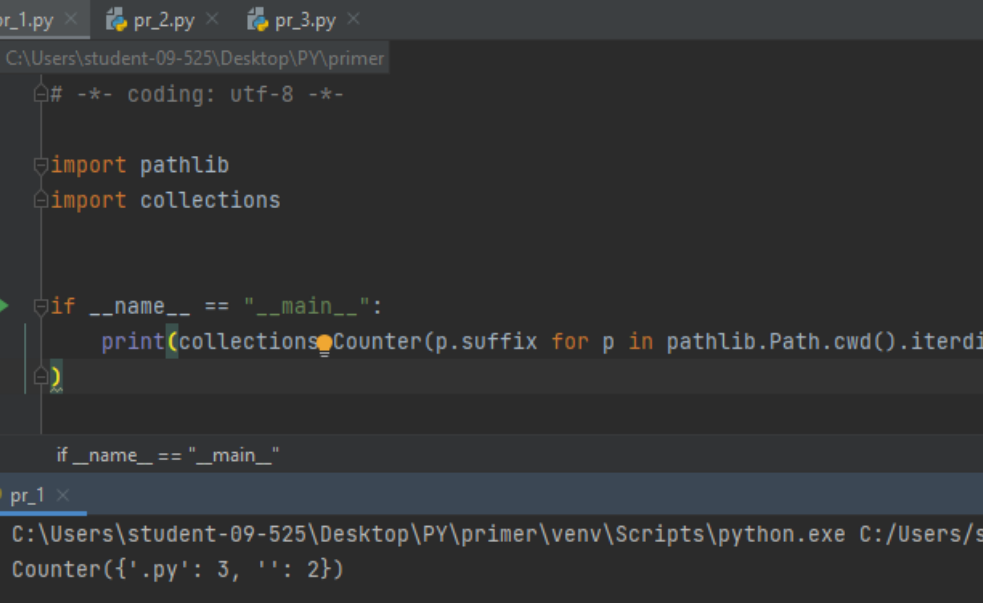
This will set  **main** as the default branch. Change the default name in your [settings](#).

 You are creating a public repository in your personal account.

Create repository

2. Проработать примеры лабораторной работы

Подсчет файлов



```
primer > pr_1.py

# -*- coding: utf-8 -*-

import pathlib
import collections

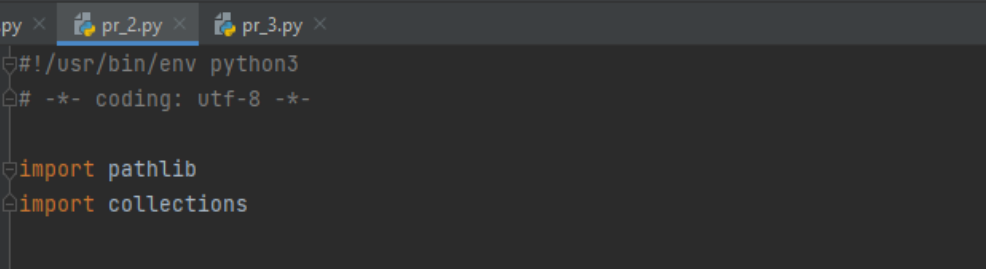
if __name__ == "__main__":
    print(collections.Counter(p.suffix for p in pathlib.Path.cwd().iterdir()))

if __name__ == "__main__":
```

Run: pr_1 x

```
C:\Users\student-09-525\Desktop\PY\primer\venv\Scripts\python.exe C:/Users/student-09-525/Desktop/PY/primer/venv/Scripts/python.exe C:/Users/student-09-525/Desktop/PY/primer/pr_1.py
Counter({''.py': 3, '': 2})

Process finished with exit code 0
```



The screenshot shows a code editor with a dark theme. The top menu bar includes 'Edit', 'View', 'Navigate', 'Code', 'Refactor', 'Run', 'Tools', 'VCS', 'Window', and 'Help'. The editor has three tabs: 'pr_1.py', 'pr_2.py' (active), and 'pr_3.py'. The active tab displays a Python script with the following content:

```
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4 import pathlib
5 import collections
6
7
8 if __name__ == "__main__":
9     print(collections.Counter(p.suffix for p in pathlib.Path.cwd().glob('*.py')))
```

The left sidebar shows a file explorer with a folder named 'primer' containing a file 'pr_2.py'. The bottom panel, titled 'Run: pr_2', shows the execution output:

```
C:\Users\student-09-525\Desktop\PY\primer\venv\Scripts\python.exe C:/Users/student-09-525
Counter({'py': 3})

Process finished with exit code 0
```

Показать дерево каталогов

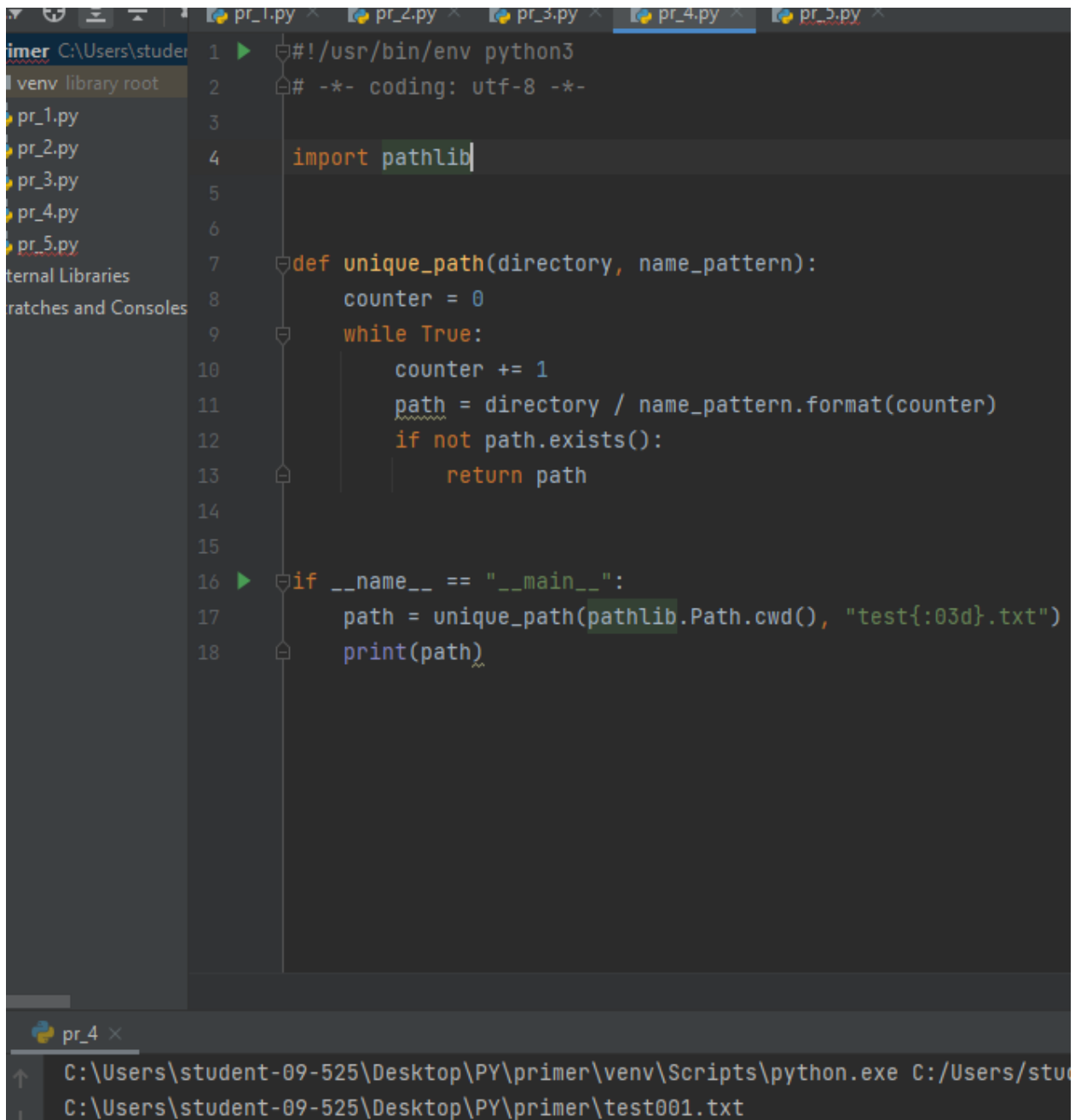
```
pr_3.py
pr_1.py x pr_2.py x pr_3.py x
1  ▶  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  import pathlib
5  import collections
6
7  def tree(directory):
8      print(f'+ {directory}')
9      for path in sorted(directory.rglob('*')):
10         depth = len(path.relative_to(directory).parts)
11         spacer = ' ' * depth
12         print(f'{spacer}+ {path.name}')
13
14  print(tree(pathlib.Path.cwd()))
```

pr_3 x

C:\Users\student-09-525\Desktop\PY\primer\venv\Scripts\python.exe
+ C:\Users\student-09-525\Desktop\PY\primer
+ .idea
+ .gitignore
+ inspectionProfiles
+ profiles_settings.xml
+ misc.xml
+ modules.xml
+ primer.iml
+ workspace.xml
+ pr_1.py
+ pr_2.py
+ pr_3.py
+ venv
+ .gitignore
+ Lib
+ site-packages
+ __pycache__
+ _virtualenv.cpython-37.pyc

Найти последний измененный файл

Создать уникальное имя файла



```
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  import pathlib
5
6
7  def unique_path(directory, name_pattern):
8      counter = 0
9      while True:
10         counter += 1
11         path = directory / name_pattern.format(counter)
12         if not path.exists():
13             return path
14
15
16  if __name__ == "__main__":
17      path = unique_path(pathlib.Path.cwd(), "test{:03d}.txt")
18      print(path)
```

The screenshot shows a Python IDE with a file explorer on the left containing files pr_1.py through pr_5.py. The main editor displays the code above. The terminal at the bottom shows the command: `C:\Users\student-09-525\Desktop\PY\primer\venv\Scripts\python.exe C:/Users/stud` and the output: `C:\Users\student-09-525\Desktop\PY\primer\test001.txt`.

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`.

```

C:\Users\M\Desktop\OPI_2.19\PY\ind>python ind_1.py add ind_1.json --name="maria" --group
="123" --marks="3 5 4 5 5"
[{'name': 'maria', 'group': '123', 'marks': [3, 5, 4, 5, 5]}]

C:\Users\M\Desktop\OPI_2.19\PY\ind>python ind_1.py display ind_1.json
+-----+-----+-----+-----+
| № | Ф.И.О. | Группа | Оценки |
+-----+-----+-----+-----+
| 1 | maria | 123 | 3,5,4,5,5 |
+-----+-----+-----+-----+

```

Рисунок – Результат работы программы

150-500 (C:) > Пользователи > M			Поиск в: M
Имя	Дата изменения	Тип	
ind_1	28.04.2023 0:57	Исходный	

Рисунок – JSON-файл в домашнем каталоге

Задание 2

Разработайте аналог утилиты tree в Linux. Используйте возможности модуля argparse для управления отображением дерева каталогов файловой системы. Добавьте дополнительные уникальные возможности в данный программный продукт.

```

C:\Users\M\Desktop\OPI_2.19\PY\ind>python ind_2.py E:\4_семестр\экс_практика
E:\4_семестр\экс_практика
экс_практика/
Новый текстовый документ.txt
делаю/
    TZ.docx
    чье-то/
        Руководство программиста5 .doc
        рук-во програм.pdf
        тз.pdf
    рук-во программиста.docx
сдаю/
    ОбразцоваМД_ТЗ.docx
документы трофимовой/
    Стандартизация разработки программных средств.pdf
    Эксплуатационная учебная практика_дневник.docx
Новая папка/
    Эксплуатационная учебная практика_индивидуальное задание.docx
    Эксплуатационная учебная практика_титул_содержание.docx
    Эксплуатационная учебная практика_дневник.docx
    2021_Эксплуатационная практика_4 семестр_задания.docx

C:\Users\M\Desktop\OPI_2.19\PY\ind>

```

ВОПРОСЫ ДЛЯ ЗАЩИТЫ РАБОТЫ

1. Какие существовали средства для работы с файловой системой до Python 3.4?

До Python 3.4 работа с путями файловой системы осуществлялась либо с помощью методов строк:

```
path.split('\\', maxsplit=1)[0]
```

либо с помощью модуля `os.path` :

```
os.path.isfile(os.path.join(os.path.expanduser('~'), 'realpython.txt'))
```

2. Что регламентирует PEP 428?

Данный PEP предлагает включить в стандартную библиотеку модуль стороннего разработчика – `pathlib`. Включение предлагается под предварительной меткой, как описано в PEP 411. Поэтому изменения в API могут быть сделаны либо в рамках процесса PEP, либо после принятия в стандартную библиотеку (и до тех пор, пока предварительная метка не будет снята).

Цель этой библиотеки - предоставить простую иерархию классов для работы с путями файловой системы и обычными операциями, которые пользователи выполняют над ними.

3. Как осуществляется создание путей средствами модуля `pathlib`?

Все, что вам действительно нужно знать, это класс `pathlib.Path`. Есть несколько разных способов создания пути. Прежде всего, существуют `classmethods` наподобие `.cwd()` (текущий рабочий каталог) и `.home()` (домашний каталог вашего пользователя):

```
import pathlib
pathlib.Path.cwd()
```

Вывод: `PosixPath('/home/gahjelle/realpython/')`

Путь также может быть явно создан из его строкового представления:

```
pathlib.Path(r'C:\Users\gahjelle\realpython\file.txt')
```

Вывод: `WindowsPath('C:/Users/gahjelle/realpython/file.txt')` Объединение путей: с помощью «\» или `.joinpath()` `pathlib.Path.home().joinpath('python', 'scripts', 'test.py')` `PosixPath('/home/gahjelle/python/scripts/test.py')`

4. Как получить путь дочернего элемента файловой системы с помощью модуля `pathlib`?

```
path = pathlib.Path('test.md').resolve()  
PosixPath('/home/gahjelle/realpython/test.md')
```

5. Как получить путь к родительским элементам файловой системы с помощью модуля `pathlib`?

```
path.parent
```

6. Как выполняются операции с файлами с помощью модуля `pathlib`?

Чтение и запись файлов

Традиционно для чтения или записи файла в Python использовалась встроенная функция `open()`. Это все еще верно, поскольку функция `open()` может напрямую использовать объекты `Path`. Следующий пример находит все заголовки в файле Markdown и печатает их:

```
path = pathlib.Path.cwd() / 'test.md' with open(path, mode='r') as fid:  
    headers = [line.strip() for line in fid if line.startswith('#')]  
print('\n'.join(headers))
```

Для простого чтения и записи файлов в библиотеке `pathlib` есть несколько удобных методов:

`.read_text()` : открыть путь в текстовом режиме и вернуть содержимое в виде строки.

`.read_bytes()` : открыть путь в двоичном/байтовом режиме и вернуть содержимое в виде строки байтов.

`.write_text()` : открыть путь и записать в него строковые данные.

`.write_bytes()` : открыть путь в двоичном/байтовом режиме и записать в него данные.

7. Как можно выделить компоненты пути файловой системы с помощью модуля `pathlib`?

Различные части пути удобно доступны как свойства. Основные примеры включают в себя:

`.name` : имя файла без какого-либо каталога

`.parent` : каталог, содержащий файл, или родительский каталог, если путь является каталогом

`.stem` : имя файла без суффикса

`.suffix` : расширение файла

`.anchor` : часть пути перед каталогами

8. Как выполнить перемещение и удаление файлов с помощью модуля `pathlib`?

Чтобы переместить файл, используйте `.replace()` . Обратите внимание, что если место назначения уже существует, `.replace()` перезапишет его. К сожалению, `pathlib` явно не поддерживает безопасное перемещение файлов. Чтобы избежать возможной перезаписи пути назначения, проще всего проверить, существует ли место назначения перед заменой:

```
if not destination.exists(): source.replace(destination)
```

Тем не менее, это оставляет дверь открытой для возможного состояния гонки. Другой процесс может добавить файл по пути `destination` между выполнением оператора `if` и метода `.replace()` . Если это вызывает озабоченность, более безопасный способ - открыть путь назначения для создания `exclusive` и явно скопировать исходные данные:

```
with destination.open(mode='xb') as fid:
```

```
    fid.write(source.read_bytes())
```

Приведенный выше код вызовет `FileExistsError` , если `destination` уже существует. Технически это копирует файл. Чтобы выполнить перемещение, просто удалите `source` после завершения копирования.

Когда вы переименовываете файлы, полезными методами могут быть

`.with_name()` и `.with_suffix()` . Они оба возвращают исходный путь, но с замененным именем или суффиксом соответственно.

```
path                                PosixPath('/home/gahjelle/realpython/test001.txt')
path.with_suffix('.py')            PosixPath('/home/gahjelle/realpython/test001.py')
path.replace(path.with_suffix('.py'))
```

Каталоги и файлы могут быть удалены с помощью `.rmdir()` и `.unlink()` соответственно.

9. Как выполнить подсчет файлов в файловой системе?

Есть несколько разных способов перечислить много файлов. Самым простым является метод `.iterdir()` , который перебирает все файлы в данном каталоге. В следующем примере комбинируется `.iterdir()` с классом `collections.Counter` для подсчета количества файлов каждого типа в текущем каталоге:

```
import collections
collections.Counter(p.suffix for p in pathlib.Path.cwd().iterdir())
Counter({'md': 2, 'txt': 4, 'pdf': 2, 'py': 1})
```

Более гибкие списки файлов могут быть созданы с помощью методов `.glob()` и `.rglob()` (рекурсивный глоб). Например, `pathlib.Path.cwd().glob('*txt')` возвращает все файлы с суффиксом `.txt` в текущем каталоге. Следующее только подсчитывает типы файлов, начинающиеся с `p` :

```
import collections
collections.Counter(p.suffix for p in pathlib.Path.cwd().glob('*p*'))
Counter({'pdf': 2, 'py': 1})
```

10. 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}')
```

11. Как создать уникальное имя файла?

Сначала укажите шаблон для имени файла с местом для счетчика.

Затем проверьте существование пути к файлу, созданного путем соединения каталога и имени файла (со значением счетчика). Если он уже существует, увеличьте счетчик и попробуйте снова:

```
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')
```

12. 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:

```
pathlib.WindowsPath('test.md')
```

`NotImplementedError: cannot instantiate 'WindowsPath' on your system` В некоторых случаях может потребоваться представление пути без

доступа к базовой файловой системе (в этом случае также может иметь смысл представлять путь Windows в системе, отличной от Windows, или наоборот). Это можно сделать с помощью объектов `PurePath`.

```
path = pathlib.PureWindowsPath(r'C:\Users\gahjelle\realpython\file.txt')
path.name
```

```
'file.txt' path.parent
```

```
PureWindowsPath('C:/Users/gahjelle/realpython') path.exists()
```

```
AttributeError: 'PureWindowsPath' object has no attribute 'exists'
```

Windows использует «\» , а Mac и Linux используют «/» в качестве разделителя. Это различие может привести к трудно обнаруживаемым ошибкам.