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 № 7
«Работа с файловой системе с использованием модуля pathlib»

по дисциплине «Основы программной инженерии»

Выполнила:
Первых Дарья
Александровна, 2 курс,
группа ПИЖ-б-о-20-1,

Проверил:
Доцент кафедры инфокоммуникаций,
Воронкин Р.А.

Ставрополь, 2022 г

ВЫПОЛНЕНИЕ

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


import collections
import pathlib


print(collections.Counter(p.suffix for p in pathlib.Path.cwd().iterdir()))
```

```
main
C:\Users\podar\study\anaconda\envs\LR7\python.exe "C:/Users/podar/study/PyChar
Counter({'': 1, '.py': 1})
```

Рисунок 1 – Подсчёт файлов

```python
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}')


tree(pathlib.Path.cwd())
```

```
tree()
main
C:\Users\podar\study\anaconda\envs\LR7\python.exe "C:/Users/po
+ C:\Users\podar\study\PyCharm Community Edition 2021.2.3\LR7
 + .idea
  + .gitignore
  + inspectionProfiles
   + profiles_settings.xml
   + Project_Default.xml
  + LR7.iml
  + misc.xml
  + modules.xml
  + workspace.xml
 + main.py
 + primer1.py
```

Рисунок 2 – Дерево каталогов

```python
# -*- coding: utf-8 -*-


from datetime import datetime
import pathlib


time, file_path = max((f.stat().st_mtime, f) for f in
                      pathlib.Path.cwd().iterdir())
print(datetime.fromtimestamp(time), file_path)
```

```
main
C:\Users\podar\study\anaconda\envs\LR7\python.exe "C:/Users/podar/study/PyCharm Community Edition 2021.2.3/LR7/main.py"
2022-04-14 22:46:51.091919 C:\Users\podar\study\PyCharm Community Edition 2021.2.3\LR7\main.py
```

Рисунок 3 – Результат нахождения последнего изменённого файла

```python
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
```

```
main
C:\Users\podar\study\anaconda\envs\LR7\python.exe "C:/Users/podar/study/
C:\Users\podar\study\PyCharm Community Edition 2021.2.3\LR7\test001.txt
```

Рисунок 4 – Создание уникального имени файла

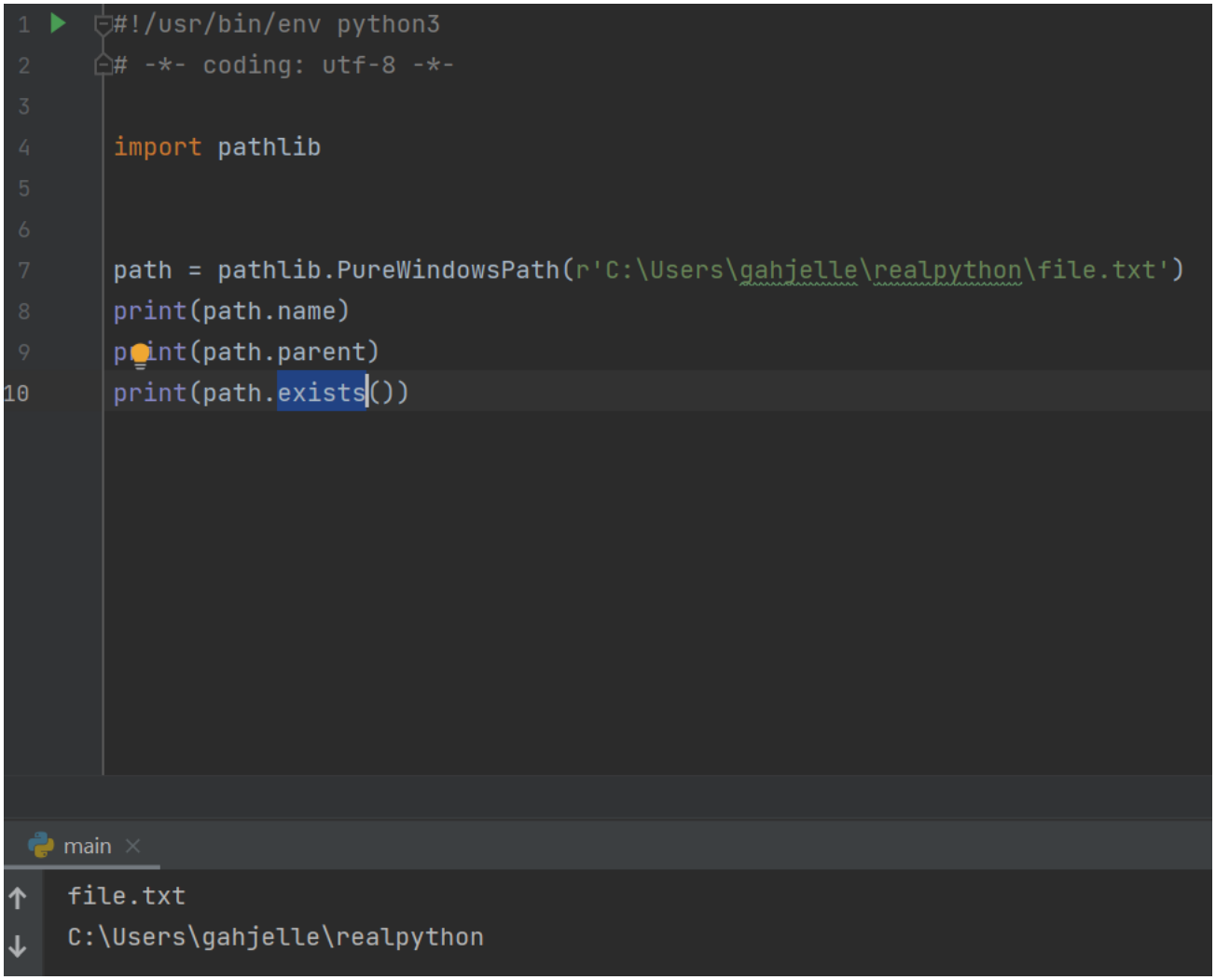

Рисунок 5 – Работа с файлами

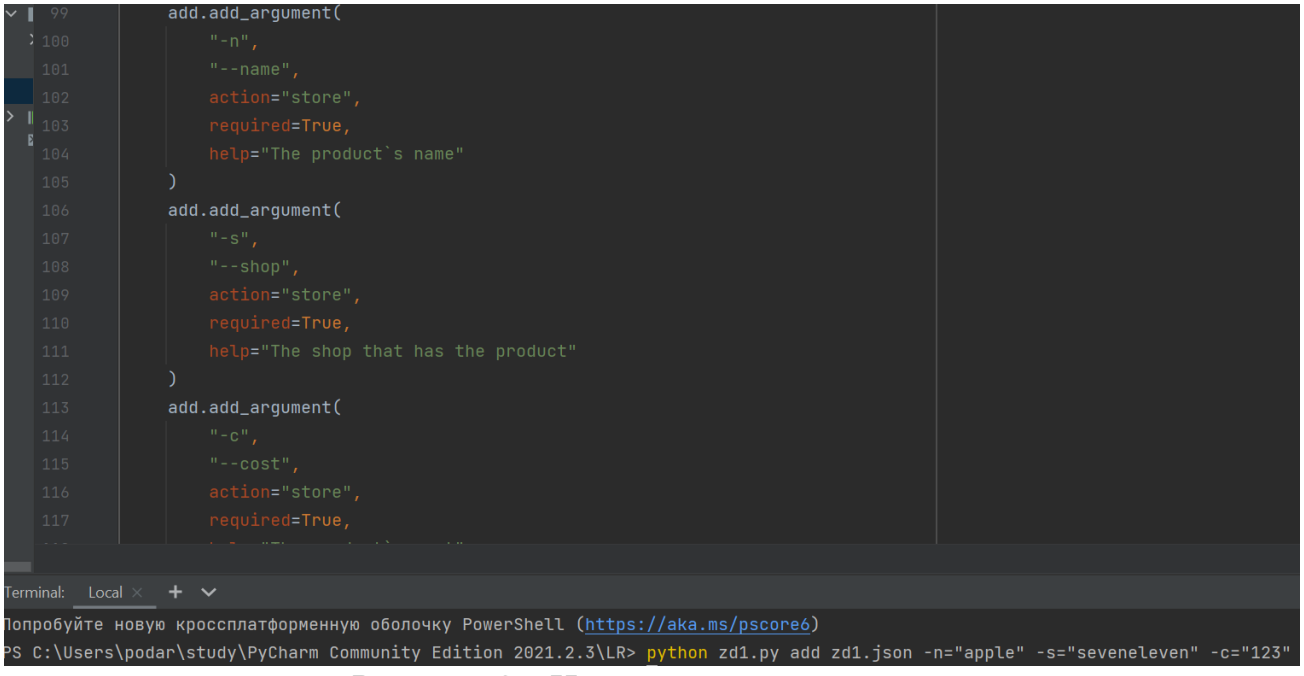

Рисунок 6 – Индивидуальное задание

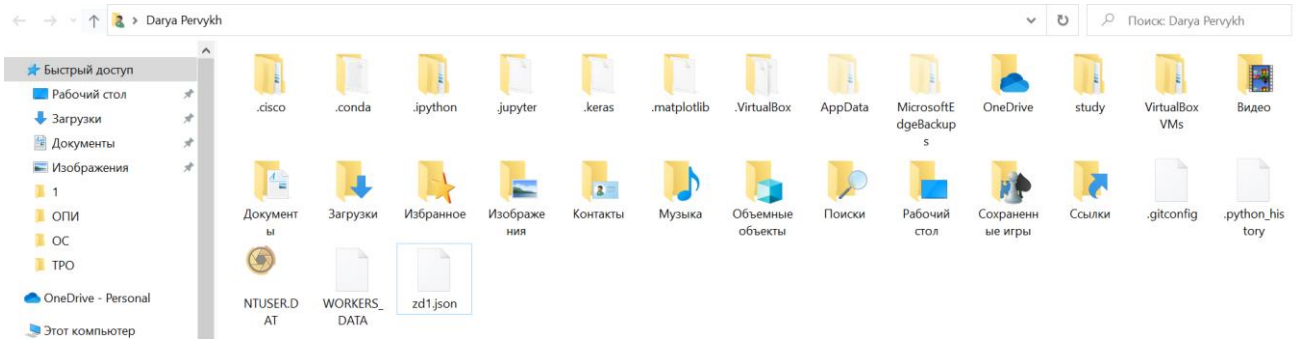

Рисунок 7 – Проверка наличия файла в домашнем каталоге

```
PS C:\Users\podar\study\PyCharm Community Edition 2021.2.3\LR> python zd1.py display zd1.json
+------+--------------------------------+----------------------+-----------------+
|  №   |      Наименование товара       |  Название магазина   |    Стоимость    |
+------+--------------------------------+----------------------+-----------------+
|    1 | apple                          | seveneleven          | 123             |
+------+--------------------------------+----------------------+-----------------+
```

Рисунок 8 – Результат работы программы

```
        if limit_to_directories:
            contents = [d for d in dir_path.iterdir() if d.is_dir()]
        else:
            contents = list(dir_path.iterdir())
        pointers = [tee] * (len(contents) - 1) + [last]
        for pointer, path in zip(pointers, contents):
            if path.is_dir():
                yield prefix + pointer + path.name
```

```
main
C:\Users\podar\study\anaconda\envs\LR\python.exe "C:/Users/podar/study/
This program is showing files and directories on C: drive or CWD
Where do you want to work? h - home, c - CWD: h
Type the directory name: study
Enter the level of search if needed (-1) if don't: -1
Do you want to print only the directories? d - yes: no
study
├── !python
│   ├── DLLs
│   │   ├── libcrypto-1_1.dll
│   │   ├── libffi-7.dll
│   │   ├── libssl-1_1.dll
│   │   ├── py.ico
│   │   ├── pyc.ico
```

Рисунок 9 – Вывод дерева каталогов

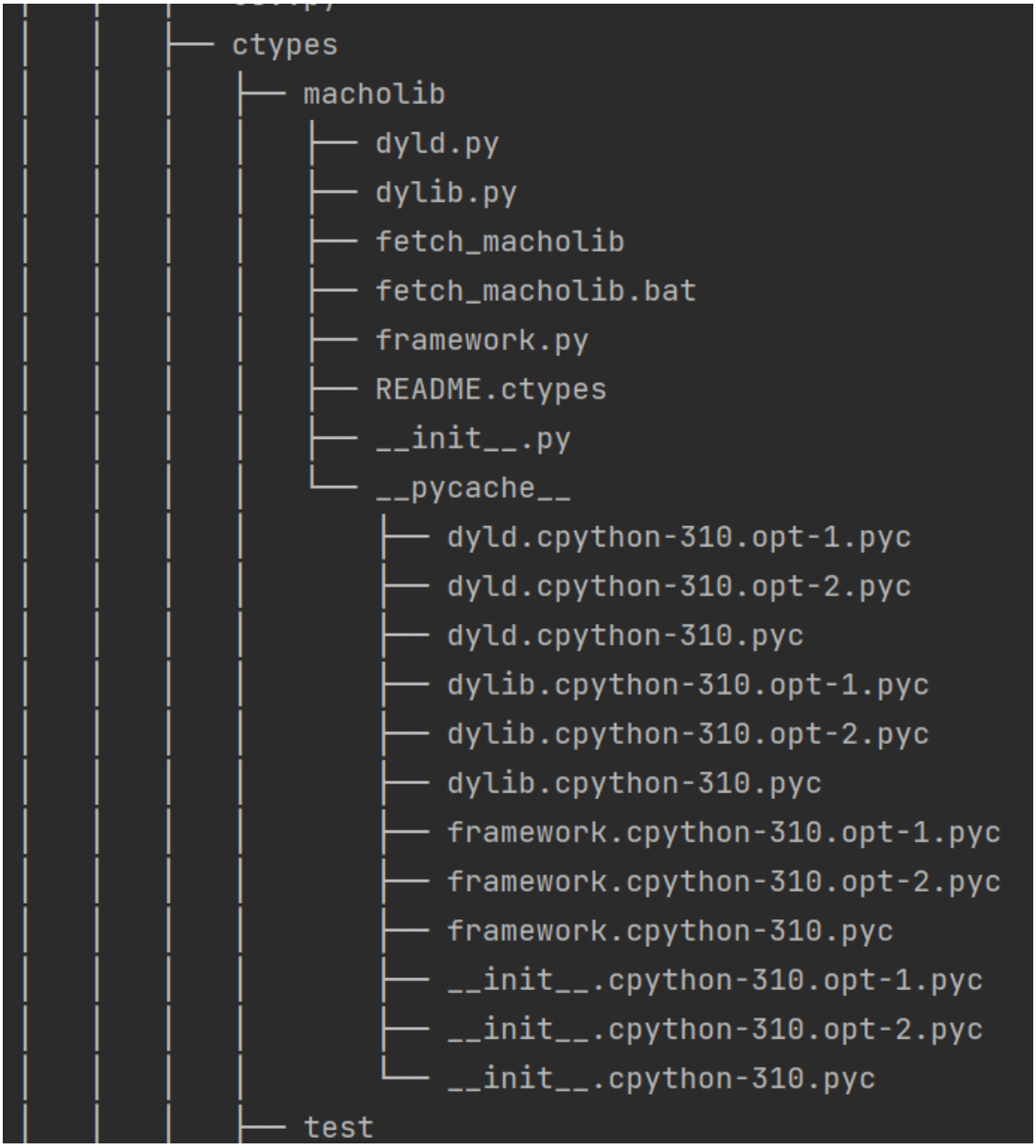

Рисунок 10 – Вывод дерева каталогов

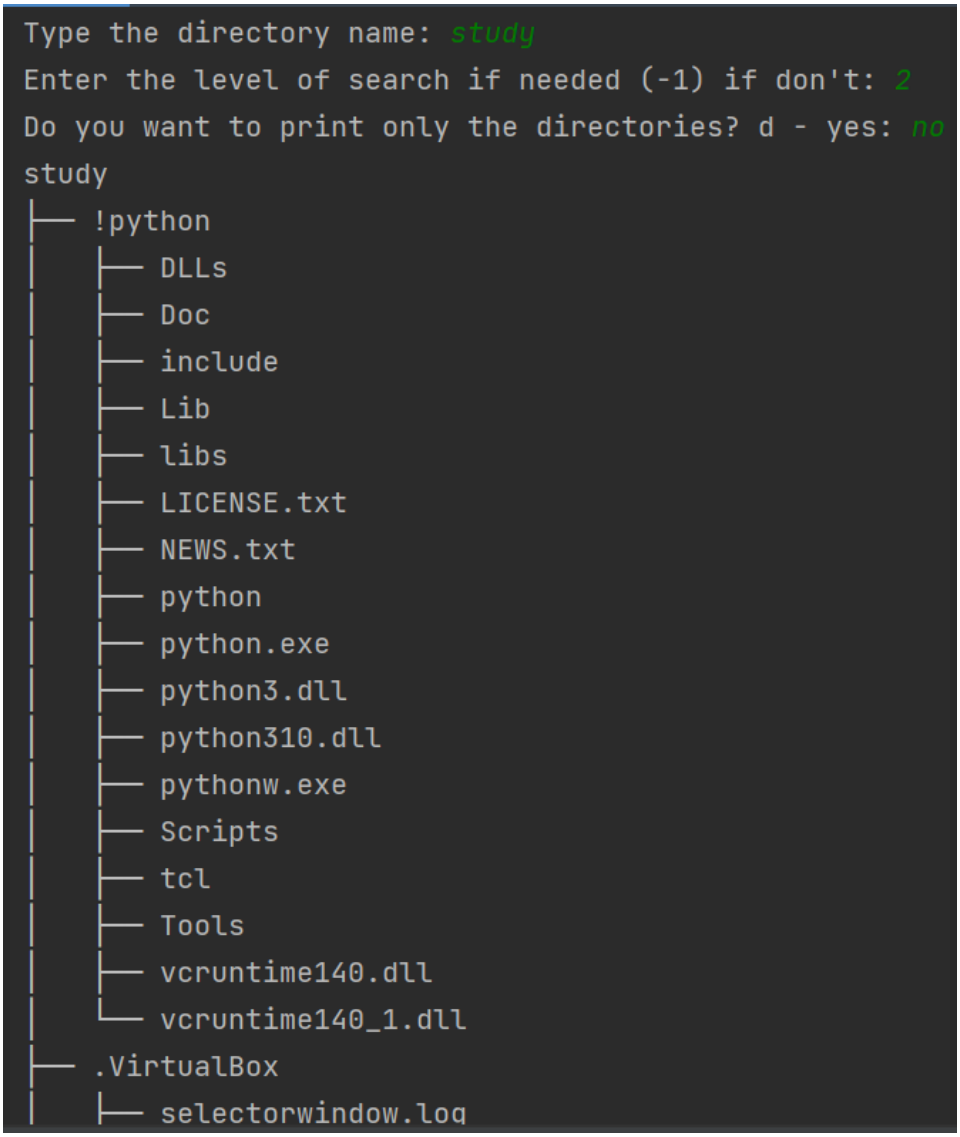

Рисунок 11 – Вывод дерева каталогов

ВОПРОСЫ

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

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

path.rsplit('\\', 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')

path.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 collection.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.