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,

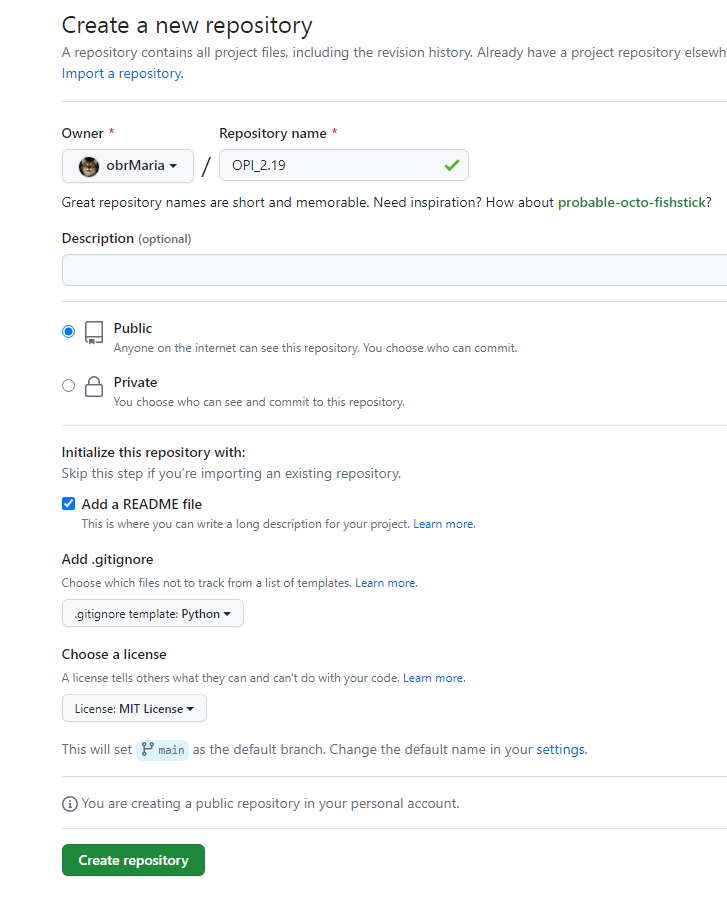
2 курс, группа ПИЖ-б-о-21-1, Проверил:

Доцент кафедры инфокоммуникаций, Воронкин Р.А.

Ставрополь, 2023 г.

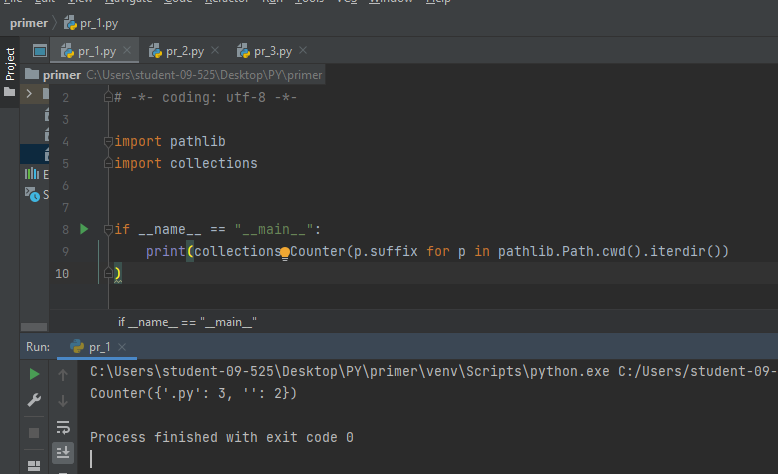
1. Создал общедоступный репозиторий на GitHub, в котором будет

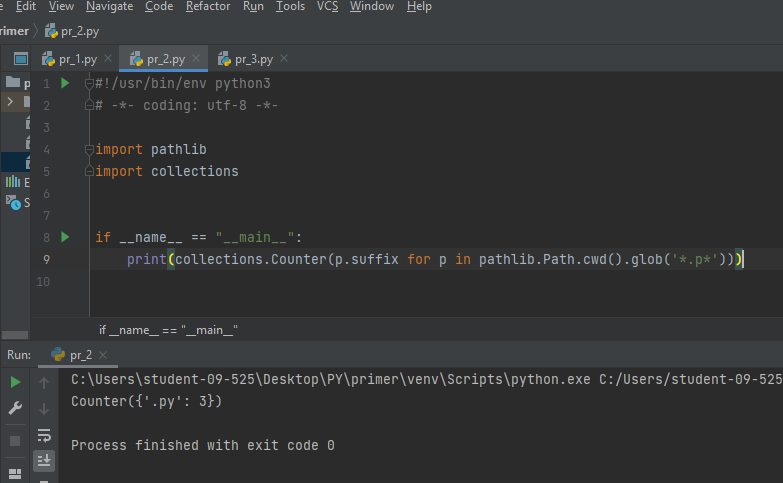
использована лицензия MIT и язык программирования Python.



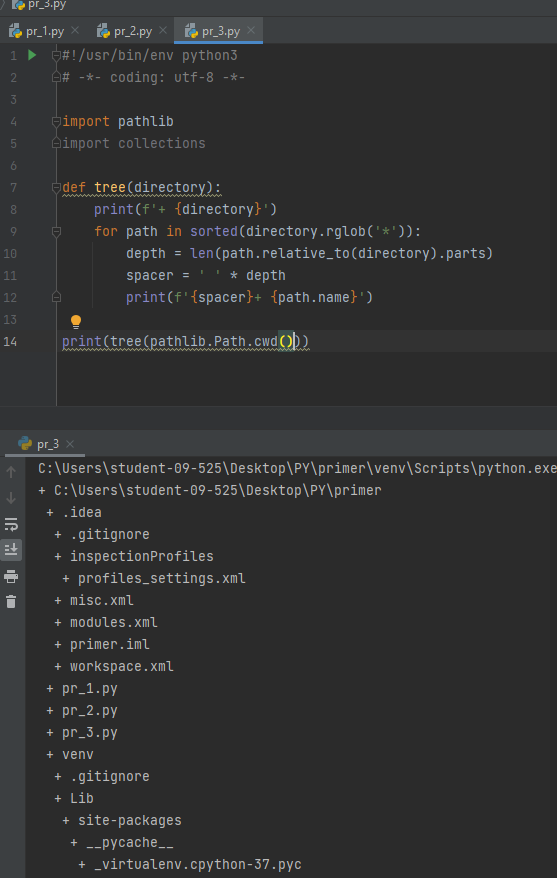
1. Проработать примеры лабораторной работы

Подсчет файлов

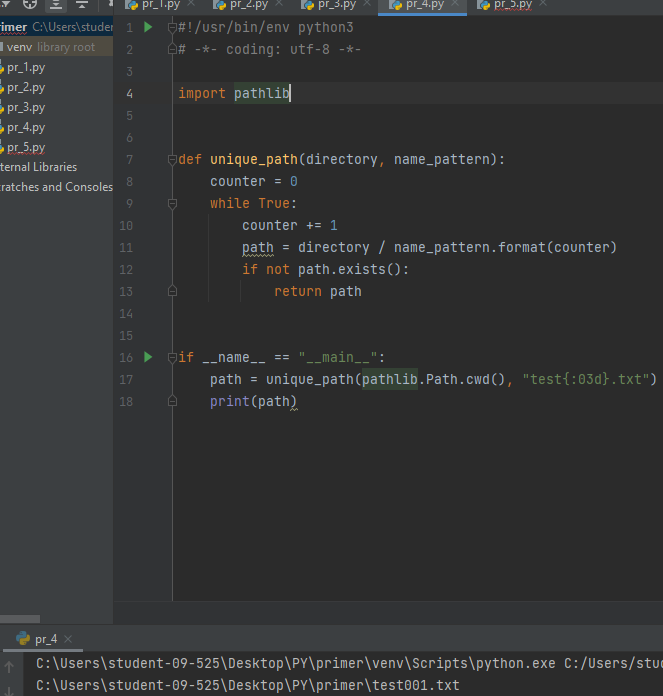




Показать дерево каталогов



Найти последний измененный файл

Создать уникальное имя файла

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.

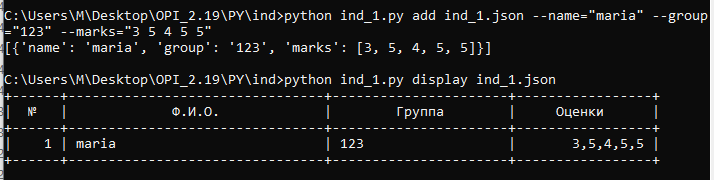


Рисунок – Результат работы программы

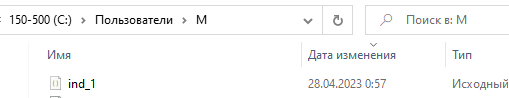
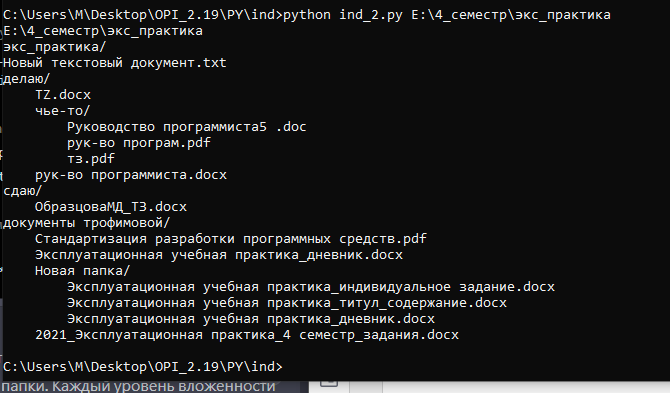


Рисунок – JSON-файл в домашнем каталоге

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.



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.rsplit('\\', maxsplit=1)[0]

либо с помощью модуля os.path : os.path.isfile(os.path.join(os.path.expanduser('~'), 'realpython.txt'))

1. 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.

1. 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')

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

path = pathlib.Path('test.md') path.resolve()

PosixPath('/home/gahjelle/realpython/test.md')

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

path.parent

1. 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.

1. 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

1. 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.

1. 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})

1. 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}')

1. 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')

1. 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.