

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное
образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Кафедра инфокоммуникаций

Отчет по лабораторной работе № 14

Работа в файловой системе в Python3 с использованием модуля pathlib

По дисциплине «Технологии программирования и алгоритмизация»

Выполнил студент группы ИВТ-б-о-20-1

Бобров Н. В. « » _____ 20__ г.

Подпись студента _____

Работа защищена « » _____ 20__ г.

Проверил Воронкин Р. А. _____

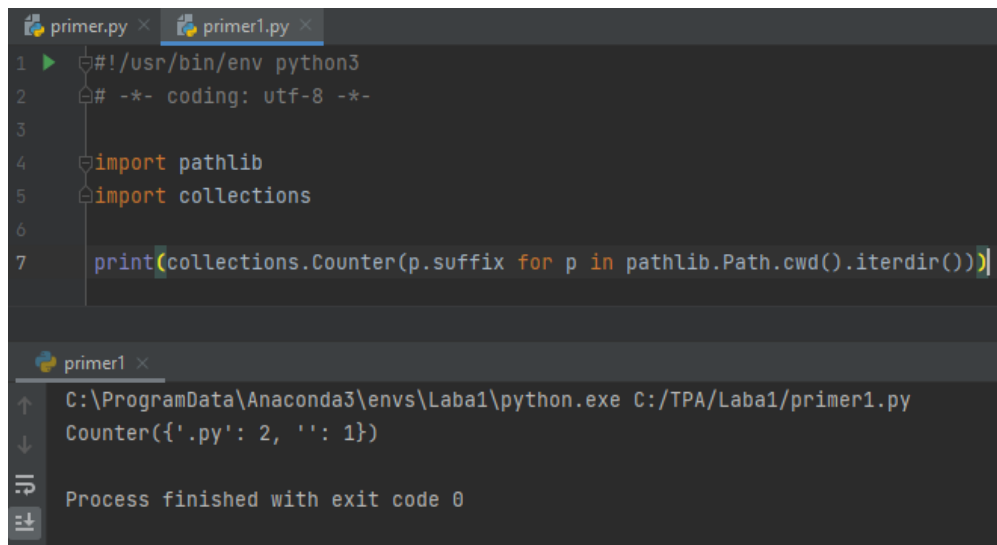
(подпись)

Ставрополь 2022

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

Ход работы:

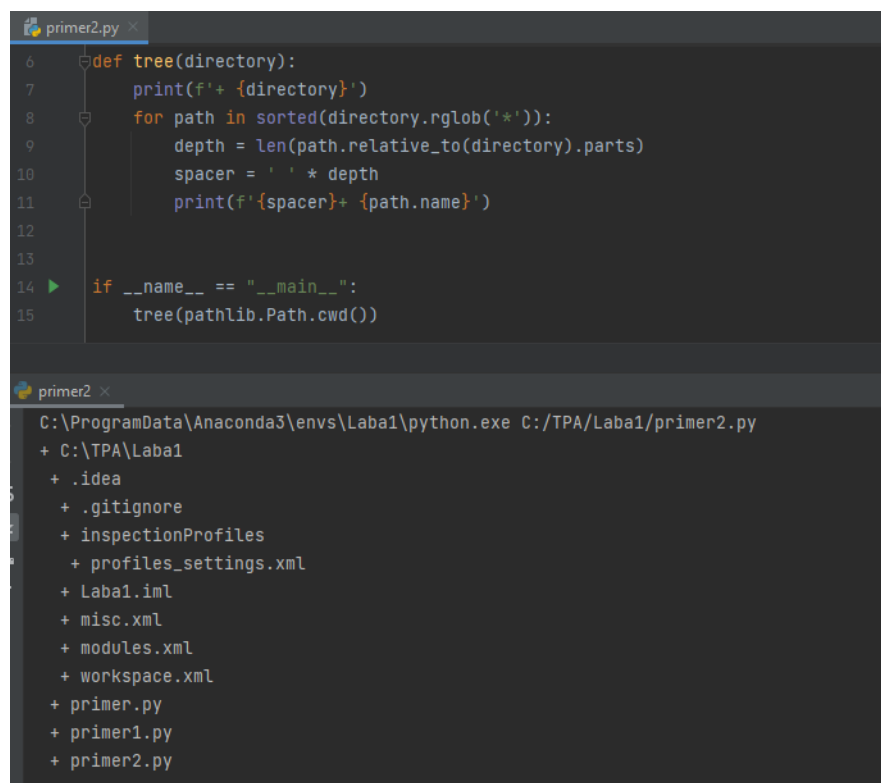
1. Создал общедоступный репозиторий и клонировал его на локальный сервер.
2. Ознакомившись с теоретическим материалом, выполнил примеры, создав для них отдельный модуль.



```
primer.py x primer1.py x
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4 import pathlib
5 import collections
6
7 print(collections.Counter(p.suffix for p in pathlib.Path.cwd().iterdir()))
```

```
primer1 x
C:\ProgramData\Anaconda3\envs\Laba1\python.exe C:/TPA/Laba1/primer1.py
Counter({' .py': 2, '': 1})
Process finished with exit code 0
```

Рисунок 1 – Результат выполнения первого примера



```
primer2.py x
6 def tree(directory):
7     print(f'+ {directory}')
8     for path in sorted(directory.rglob('*')):
9         depth = len(path.relative_to(directory).parts)
10        spacer = '  ' * depth
11        print(f'+ {spacer}{path.name}')
12
13
14 if __name__ == "__main__":
15     tree(pathlib.Path.cwd())
```

```
primer2 x
C:\ProgramData\Anaconda3\envs\Laba1\python.exe C:/TPA/Laba1/primer2.py
+ C:\TPA\Laba1
+ .idea
+ .gitignore
+ inspectionProfiles
+ profiles_settings.xml
+ Laba1.iml
+ misc.xml
+ modules.xml
+ workspace.xml
+ primer.py
+ primer1.py
+ primer2.py
```

Рисунок 2 – Результат выполнения второго примера

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

unique_path() > while True

primer x
C:\ProgramData\Anaconda3\envs\Lab1\python.exe C:/TPA/Lab1/primer.py
C:\TPA\Lab1\test001.txt

Process finished with exit code 0
```

Рисунок 3 – Результат выполнения третьего примера

```
Python Console x
Python 3.10.0 | packaged by conda-forge | (default, Nov 10 2021,
>>> import pathlib
>>> path = pathlib.PureWindowsPath(r'C:\TPA\Lab1\primer3.py')
>>> path.name
'primer3.py'
>>> path.parent
PureWindowsPath('C:/TPA/Lab1')
```

Рисунок 4 – Результат выполнения четвертого примера

3. Приступил к выполнению индивидуальных заданий своего варианта.
4. Добавил возможность хранения файла json в домашнем каталоге пользователя.

```
103 def save_students(file_name, students):
104     """
105     Сохранить данные о студенте
106     """
107     with open(file_name, "w", encoding="utf-8") as fout:
108         json.dump(students, fout, ensure_ascii=False, indent=4)
109     directory = pathlib.Path.cwd().joinpath(file_name)
110     directory.replace(pathlib.Path.home().joinpath(file_name))
```

Рисунок 5 – Возможность хранения файла в домашнем каталоге

Этот компьютер > Windows (C:) > Пользователи > nbobr

Имя	Дата изменения	Тип	Размер
.conda	28.01.2022 22:03	Папка с файлами	
.continuum	28.01.2022 22:02	Папка с файлами	
.VirtualBox	03.02.2022 12:42	Папка с файлами	
AppData	28.01.2022 6:59	Папка с файлами	
Documents	28.01.2022 7:06	Папка с файлами	
laba2.19	07.02.2022 10:59	Папка с файлами	
labwork_4.1	06.02.2022 16:10	Папка с файлами	
OneDrive	05.02.2022 1:56	Папка с файлами	
Видео	31.01.2022 9:44	Папка с файлами	
Загрузки	19.02.2022 18:22	Папка с файлами	
Избранное	28.01.2022 7:02	Папка с файлами	
Контакты	28.01.2022 7:02	Папка с файлами	
Музыка	28.01.2022 7:02	Папка с файлами	
Объемные объекты	28.01.2022 7:02	Папка с файлами	
Поиски	28.01.2022 7:04	Папка с файлами	
Рабочий стол	28.01.2022 14:44	Папка с файлами	
Сохраненные игры	28.01.2022 7:02	Папка с файлами	
Ссылки	28.01.2022 7:02	Папка с файлами	
.condarc	28.01.2022 22:02	Файл "CONDARC"	1 КБ
.gitconfig	28.01.2022 14:46	Файл "GITCONFIG"	1 КБ
NTUSER.DAT	17.02.2022 16:27	Файл "DAT"	2 304 КБ
students.json	19.02.2022 15:38	Файл "JSON"	1 КБ

Рисунок 6 – Домашняя папка пользователя

5. Затем приступил к выполнению второго задания.

Условие: 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.

6. Написал код для реализации задачи.

```

task2.py x
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4
5  import argparse
6  import pathlib
7  import colorama
8  from colorama import Fore, Style
9
10
11 def tree(directory):
12     print(Fore.RED + f'>>> {directory}')
13     for path in sorted(directory.rglob('*')):
14         depth = len(path.relative_to(directory).parts)
15         spacer = ' ' * depth
16         print(Fore.GREEN + Style.BRIGHT + f'{spacer} >> {path.name}')
17         for new_path in sorted(directory.joinpath(path).glob('*')):
18             depth = len(new_path.relative_to(directory.joinpath(path)).parts)
19             spacer = '\t' * depth
20             print(Fore.BLUE + f'{spacer} > {new_path.name}')
21
22

```

Рисунок 7 – Функция def tree

```

23 def main(command_line=None):
24     colorama.init()
25     current = pathlib.Path.cwd()
26     file_parser = argparse.ArgumentParser(add_help=False)
27
28     # Создаем основной парсер командной строки
29     parser = argparse.ArgumentParser("tree")
30     parser.add_argument(
31         "--version",
32         action="version",
33         help="The main parser",
34         version="%(prog)s 0.1.0"
35     )
36
37     subparsers = parser.add_subparsers(dest="command")
38
39     # Создаем субпарсер для создания новой папки
40     create = subparsers.add_parser(
41         "mkdir",
42         parents=[file_parser]
43     )
44     create.add_argument(
45         "filename",
46         action="store"
47     )
48
49 if __name__ == "__main__":

```

Рисунок 8 – Функция def main

```
77     args = parser.parse_args(command_line)
78     if args.command == 'mkdir':
79         directory_path = current / args.filename
80         directory_path.mkdir()
81         tree(current)
82     elif args.command == "rmdir":
83         directory_path = current / args.filename
84         directory_path.rmdir()
85         tree(current)
86     elif args.command == "touch":
87         directory_path = current / args.filename
88         directory_path.touch()
89         tree(current)
90     elif args.command == "rm":
91         directory_path = current / args.filename
92         directory_path.unlink()
93         tree(current)
94     else:
95         tree(current)
96
97
98 ► if __name__ == "__main__":
99     main()
```

Рисунок 9 – Код программы

7. Сделал проверку разработанной программы через Anaconda PowerShell.

```
(base) PS C:\tpa\laba1> python task2.py
>>> C:\tpa\laba1
>> .idea
> .gitignore
> .name
> inspectionProfiles
> Laba1.iml
> misc.xml
> modules.xml
> workspace.xml
>> .gitignore
>> .name
>> inspectionProfiles
> profiles_settings.xml
>> profiles_settings.xml
>> Laba1.iml
>> misc.xml
>> modules.xml
>> workspace.xml
>> class
>> primer1.py
>> primer2.py
>> primer3.py
>> task1.py
>> task2.py
```

Рисунок 10 – Запуск без использования команд

```
(base) PS C:\tpa\laba1> python task2.py mkdir .abc
>>> C:\tpa\laba1
>> .abc
>> .idea
> .gitignore
> .name
> inspectionProfiles
> Laba1.iml
> misc.xml
> modules.xml
> workspace.xml
>> .gitignore
>> .name
>> inspectionProfiles
> profiles_settings.xml
>> profiles_settings.xml
```

Рисунок 11 – Создание каталога с помощью mkdir

```
(base) PS C:\tpa\laba1> python task2.py rmdir .abc
>>> C:\tpa\laba1
>> .idea
> .gitignore
> .name
> inspectionProfiles
> Laba1.iml
> misc.xml
> modules.xml
> workspace.xml
```

Рисунок 12 – Удаление каталога

```
(base) PS C:\tpa\laba1> python task2.py touch Bobrov.txt
>>> C:\tpa\laba1
>> .idea
> .gitignore
> .name
> inspectionProfiles
> Laba1.iml
> misc.xml
> modules.xml
> workspace.xml
>> .gitignore
>> .name
>> inspectionProfiles
> profiles_settings.xml
>> profiles_settings.xml
>> Laba1.iml
>> misc.xml
>> modules.xml
>> workspace.xml
>> Bobrov.txt
```

Рисунок 13 – Создание файла

```
(base) PS C:\tpa\laba1> python task2.py rm Bobrov.txt
>>> C:\tpa\laba1
>> .idea
> .gitignore
> .name
> inspectionProfiles
> Laba1.iml
> misc.xml
> modules.xml
> workspace.xml
>> .gitignore
>> .name
>> inspectionProfiles
> profiles_settings.xml
>> profiles_settings.xml
>> Laba1.iml
>> misc.xml
>> modules.xml
>> workspace.xml
>> primer1.py
>> primer2.py
>> primer3.py
```

Рисунок 14 – Удаление файла

Контрольные вопросы:

1. Какие существовали средства для работы с файловой системой до Python 3.4?
 - Методы строк, например `path.rsplit("\\", maxsplit=1)[0]`
 - Модуль `os.path`
2. Что регламентирует PEP 428?

Модуль Pathlib – Объектно-ориентированные пути файловой системы

3. 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)

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

При помощи метода `resolve()`.

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

При помощи свойства `parent`.

6. 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.

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

`.name`

`.parent`

`.stem`

`.suffix`

`.anchor`

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

`.replace()` – метод перемещения файлов

`.unlink()` – метод удаления файлов

9. Как выполнить подсчет файлов в файловой системе?

Метод `.iterdir()`

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?

```
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.

Вывод: в ходе выполнения лабораторной работы были приобретены навыки по работе с библиотеками `pathlib` и `colorama`, а также закреплены знания по использованию библиотеки `argparse`.