

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

**Отчет о лабораторной работе №2.19 по дисциплине основы
программной инженерии**

Выполнил:

Кожухов Филипп Денисович,
2 курс, группа ПИЖ-б-о-20-1,

Проверил:

Доцент кафедры
прикладной математики и
компьютерной
безопасности, Воронкин Р.А.

Отчет защищен с оценкой _____ Дата защиты _____

Ставрополь, 2022 г.

ВЫПОЛНЕНИЕ:

```
1 ▶ #!/usr/bin/env python3
2   # -*- coding: utf-8 -*-
3
4
5   import pathlib
6   import collections
7
8   if __name__ == "__main__":
9       print(
10          collections.Counter(
11              p.suffix for p in pathlib.Path.cwd().iterdir()
12          )
13      )
14
```

C:\Python37\python.exe C:/Users/student-09-525/PycharmProjects/OPI_7/Examples/e1.py
Counter({''.py': 4})

Process finished with exit code 0

```
1 ▶ #!/usr/bin/env python3
2   # -*- coding: utf-8 -*-
3
4
5   import pathlib
6
7
8   def tree(directory):
9       print(f'+ {directory}')
10      for path in sorted(directory.rglob('*')):
11          depth = len(path.relative_to(directory).parts)
12          spacer = ' ' * depth
13          print(f'{spacer}+ {path.name}')
14
15
16   if __name__ == "__main__":
17       tree(pathlib.Path.cwd())
18
```

+ C:\Users\student-09-525\PycharmProjects\OPI_7\Examples
+ e1.py
+ e2.py
+ e3.py
+ e4.py

Process finished with exit code 0

```

1 ▶ #!/usr/bin/env python3
2   # -*- coding: utf-8 -*-
3
4
5   from datetime import datetime
6   import pathlib
7
8
9   def last_changed(directory):
10      time, file_path = max(
11          (f.stat().st_mtime, f) for f in directory.iterdir()
12      )
13      print(datetime.fromtimestamp(time), file_path)
14
15
16 ▶ if __name__ == "__main__":
17     last_changed(pathlib.Path.cwd())
18
C:\Python37\python.exe C:/Users/student-09-525/PycharmProjects/OPI_7/Examples/e3.py
2022-05-19 13:26:57.776793 C:\Users\student-09-525\PycharmProjects\OPI_7\Examples\e4.py

Process finished with exit code 0

```

```

armProjects/OPI_7 1 ▶ #!/usr/bin/env python3
2   # -*- coding: utf-8 -*-
3
4
5   import pathlib
6
7
8   def unique_path(directory, name_pattern):
9       counter = 0
10      while True:
11          counter += 1
12          path = directory/name_pattern.format(counter)
13          if not path.exists():
14              return path
15
16
17 ▶ if __name__ == "__main__":
18     path = unique_path(pathlib.Path.cwd(), 'test{:03d}.txt')
19     print(path)
20
C:\Python37\python.exe C:/Users/student-09-525/PycharmProjects/OPI_7/Examples/e4.py
C:\Users\student-09-525\PycharmProjects\OPI_7\Examples\test001.txt

Process finished with exit code 0

```

Рисунки 1-4 - Примеры №1, 2, 3 и 4

```

1  ▶  #!/usr/bin/env python3
2      # -*- coding: utf-8 -*-
3
4      import ...
5
6
7
8
9      def get_flight(fls, dest, num, type):
10         """
11         Добавить данные о работнике.
12         """
13         fls.append(
14             {
15                 "flight_destination": dest,
16                 "flight_number": num,
17                 "airplane_type": type
18             }
19         )
20         return fls
21
22
23     def display_flights(flights):
24         """
25         Отобразить список рейсов
26         """
27         if flights:
28             line = '+-{}-+-{}-+-{}-+-{}+'.format(
29                 '-' * 4,
30                 '-' * 30,
31                 '-' * 20,
32                 '-' * 15
33             )
34             print(line)
35             print(
36                 '| {:^4} | {:^30} | {:^20} | {:^15} |'.format(
37                     "No",
38                     "Пункт назначения",
39                     "Номер рейса",
40                     "Тип самолета"
41                 )
42             )
43             print(line)
44             for idx, flight in enumerate(flights, 1):
45                 print(
46                     '| {:>4} | {:<30} | {:<20} | {:<15} |'.format(
47                         idx,
48                         flight.get('flight_destination', ''),
49                         flight.get('flight_number', ''),
50                         flight.get('airplane_type', 0)
51                     )
52                 )
53             print(line)
54
55         else:
56             print("Список рейсов пуст")
57
58
59     def select_flights(flights, airplane_type):
60         """
61         Выбрать рейсы самолётов заданного типа
62         """
63         count = 0
64         res = []
65         for flight in flights:
66             if flight.get('airplane_type') == airplane_type:

```

```

67         count += 1
68         res.append(flight)
69     if count == 0:
70         print("рейсы не найдены")
71
72     return res
73
74
75     def save_flights(file_name, fls):
76         """
77         Сохранить все записи полётов в файл JSON.
78         """
79         # Открыть файл с заданным именем для записи.
80         with open(file_name, "w", encoding="utf-8") as fout:
81             # Выполнить сериализацию данных в формат JSON.
82             # Для поддержки кириллицы установим ensure_ascii=False
83             json.dump(fls, fout, ensure_ascii=False, indent=4)
84
85
86     def load_flights(file_name):
87         """
88         Загрузить все записи полётов из файла JSON.
89         """
90         # Открыть файл с заданным именем для чтения.
91         with open(file_name, "r", encoding="utf-8") as fin:
92             return json.load(fin)
93
94
95     def main(command_line=None):
96         """
97         Главная функция программы
98         """
99         file_parser = argparse.ArgumentParser(add_help=False)
100         file_parser.add_argument(
101             "filename",
102             action="store",
103             help="The data file name"
104         )
105         parser = argparse.ArgumentParser("flights")
106         parser.add_argument(
107             "--version",
108             action="version",
109             version=f"%(prog)s 0.1.0"
110         )
111         subparsers = parser.add_subparsers(dest="command")
112         add = subparsers.add_parser(
113             "add",
114             parents=[file_parser],
115             help="Add a new flight"
116         )
117         add.add_argument(
118             "-fld",
119             "--flight_dest",
120             action="store",
121             required=True,
122             help="The flight destination"
123         )
124         add.add_argument(
125             "-n",
126             "--number",
127             action="store",
128             help="The flight number"
129         )
130         add.add_argument(





```

```

131         "-t",
132         "--type",
133         action="store",
134         required=True,
135         help="The airplane type"
136     )
137     _ = subparsers.add_parser(
138         "display",
139         parents=[file_parser],
140         help="Display all flights"
141     )
142     select = subparsers.add_parser(
143         "select",
144         parents=[file_parser],
145         help="Select the flights"
146     )
147     select.add_argument(
148         "-t",
149         "--type",
150         action="store",
151         required=True,
152         help="The required flight type"
153     )
154     args = parser.parse_args(command_line)
155     destination = pathlib.Path.home() / args.filename
156     is_dirty = False
157     if destination.exists():
158         flights = load_flights(destination)
159     else:
160         flights = []
161     if args.command == "add":
162         flights = get_flight(
163             flights,
164             args.flight_dest,
165             args.number,
166             args.type
167         )
168         is_dirty = True
169     elif args.command == "display":
170         display_flights(flights)
171     elif args.command == "select":
172         selected = select_flights(flights, args.type)
173         display_flights(selected)
174     if is_dirty:
175         save_flights(destination, flights)
176
177
178 ▶ if __name__ == '__main__':
179     main()
180
PS C:\Users\student-09-525\PycharmProjects\0PI_7\Individual> python ind1.py add backup.json --flight_dest="Monaco" --number="CQ231" --type="Military"
PS C:\Users\student-09-525\PycharmProjects\0PI_7\Individual>

PS C:\Users\student-09-525\PycharmProjects\0PI_7\Individual> python ind1.py display backup.json
+-----+-----+-----+-----+
| No | Пункт назначения | Номер рейса | Тип самолета |
+-----+-----+-----+-----+
| 1 | Monaco | CQ231 | Military |
+-----+-----+-----+-----+
PS C:\Users\student-09-525\PycharmProjects\0PI_7\Individual>

```

	Сохраненные игры	19.05.2022 11:28	Папка с файлами	
	Ссылки	19.05.2022 11:28	Папка с файлами	
	.gitconfig	19.05.2022 11:50	Файл "GITCONFIG"	1 КБ
	backup.json	19.05.2022 13:43	JSON File	1 КБ

Рисунки 5-10 - Индивидуальное задание №1

```

1 ▶ #!/usr/bin/env python3
2   # -*- coding: utf-8 -*-
3
4   from pathlib import Path
5   from itertools import islice
6
7   space = ' '
8   branch = '| '
9   tee = '|_ '
10  last = '|_ '
11
12
13  def tree(dir_path, level=-1, limit_to_directories=False,
14          length_limit=1000):
15      dir_path = Path(dir_path)
16      files = 0
17      directories = 0
18
19      def inner(dir_path, prefix='', level=-1):
20          nonlocal files, directories
21          if not level:
22              return
23          if limit_to_directories:
24              contents = [d for d in dir_path.iterdir() if d.is_dir()]
25          else:
26              contents = list(dir_path.iterdir())
27          pointers = [tee] * (len(contents) - 1) + [last]
28          for pointer, path in zip(pointers, contents):
29              if path.is_dir():
30                  yield prefix + pointer + path.name
31                  directories += 1
32                  extension = branch if pointer == tee else space
33                  yield from inner(path, prefix=prefix + extension,
34                                level=level - 1)
35              elif not limit_to_directories:
36                  yield prefix + pointer + path.name
37                  files += 1
38
39      print(dir_path.name)
40      iterator = inner(dir_path, level=level)
41      for line in islice(iterator, length_limit):
42          print(line)
43      if next(iterator, None):
44          print(f'... length_limit, {length_limit}, reached, counted:')
45      print(
46          f'\n{directories} directories' + (f', {files} files' if files else ''))
47
48
49 ▶ if __name__ == "__main__":
50     print("This program is showing files and directories on C: drive or CWD")
51     option = input("Where do you want to work? h - home, c - CWD: ")
52     if option == "h":
53         directory = input("Type the directory name: ")
54         lvl = int(input("Enter the level of search if needed (-1) if don't: "))
55         limit_d = input("Do you want to print only the directories? d - yes: ")
56         if limit_d == "d":
57             if lvl > 0:
58                 tree(Path.home() / directory, lvl, True)
59             else:
60                 tree(Path.home() / directory, True)
61         else:
62             if lvl > 0:
63                 tree(Path.home() / directory, lvl)
64             else:
65                 tree(Path.home() / directory)
66     if option == "c":
67         lvl = int(input("Enter the level of search if needed (-1) if don't: "))
68         limit_d = input("Do you want to print only the directories? d - yes: ")
69         if limit_d == "d":
70             if lvl > 0:
71                 tree(Path.cwd(), lvl, True)
72             else:
73                 tree(Path.cwd(), True)
74         else:
75             if lvl > 0:
76                 tree(Path.cwd(), lvl)
77             else:
78                 tree(Path.cwd())
79

```

Рисунок 11 - Код ИДЗ №2

```

C:\Python37\python.exe C:/Users/student-09-525/PycharmProjects/OPI_7/Individual/ind2.py
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: PycharmProjects
Enter the level of search if needed (-1) if don't: -1
Do you want to print only the directories? d - yes: d
PycharmProjects
├── OPI_6
├── OPI_7

2 directories

C:\Python37\python.exe C:/Users/student-09-525/PycharmProjects/OPI_7/Individual/ind2.py
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: PycharmProjects
Enter the level of search if needed (-1) if don't: 2
Do you want to print only the directories? d - yes: no
PycharmProjects
├── OPI_6
│   ├── .git
│   ├── .gitignore
│   ├── .idea
│   ├── Docs
│   ├── Examples
│   ├── Individual
│   ├── LICENSE
│   ├── README.md
│   └── requirements.txt
├── OPI_7
│   ├── .git
│   ├── .gitignore
│   ├── .idea
│   ├── Examples
│   ├── Individual
│   ├── LICENSE
│   ├── README.md
│   └── test001.txt

11 directories, 8 files

C:\Python37\python.exe C:/Users/student-09-525/PycharmProjects/OPI_7/Individual/ind2.py
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: PycharmProjects
Enter the level of search if needed (-1) if don't: 3
Do you want to print only the directories? d - yes: d
PycharmProjects
├── OPI_6
│   ├── .git
│   │   ├── hooks
│   │   ├── info
│   │   ├── logs
│   │   ├── objects
│   │   └── refs
│   ├── .idea
│   │   └── inspectionProfiles
│   ├── Docs
│   ├── Examples
│   └── Individual
├── OPI_7
│   ├── .git
│   │   ├── hooks
│   │   ├── info
│   │   ├── logs
│   │   ├── objects
│   │   └── refs
│   ├── .idea
│   │   └── inspectionProfiles
│   ├── Examples
│   └── Individual

23 directories

C:\Python37\python.exe C:/Users/student-09-525/PycharmProjects/OPI_7/Individual/ind2.py
This program is showing files and directories on C: drive or CWD
Where do you want to work? h - home, c - CWD: c
Enter the level of search if needed (-1) if don't: -1
Do you want to print only the directories? d - yes: d
Individual
├── ind1.py
└── ind2.py

0 directories, 2 files

Process finished with exit code 0

```

Рисунок 12 - Примеры работы программы

ОТВЕТЫ НА ВОПРОСЫ

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')  
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 `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.

