

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 №3 по дисциплине
«Основы программной инженерии»

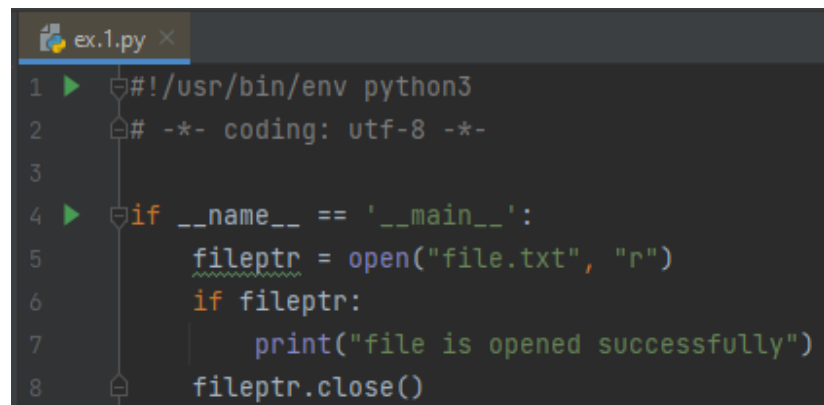
Выполнил студент
2 курса, группы ПИЖ-б-о-20-1
Тотубалина С.С.

Проверил:
Доцент кафедры инфокоммуникаций,
Воронкин Р.А.

Ставрополь, 2022 г

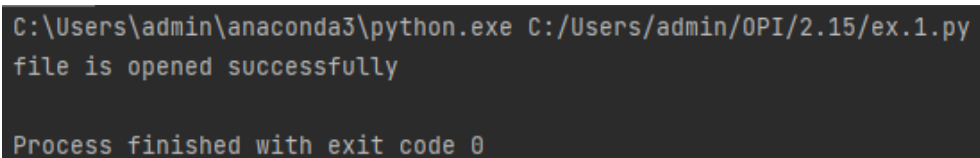
Ход работы

1. Проработка примеров



```
ex.1.py x
1 ▶ #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4 ▶ if __name__ == '__main__':
5     fileptr = open("file.txt", "r")
6     if fileptr:
7         print("file is opened successfully")
8     fileptr.close()
```

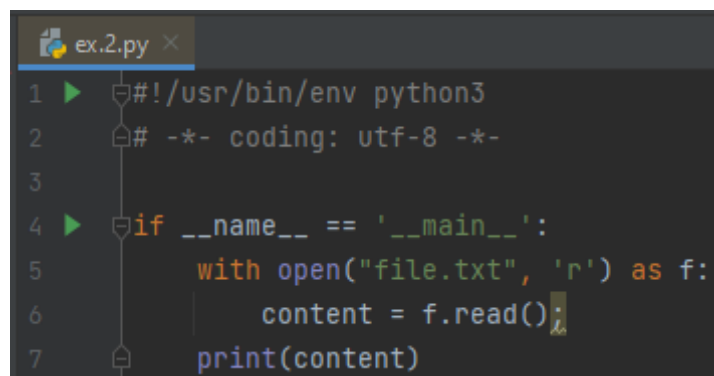
Рисунок 1 – код программы ex.1.py



```
C:\Users\admin\anaconda3\python.exe C:/Users/admin/OPI/2.15/ex.1.py
file is opened successfully

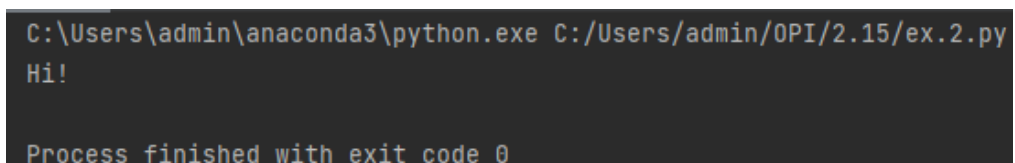
Process finished with exit code 0
```

Рисунок 2 – результат работы программы ex.1.py



```
ex.2.py x
1 ▶ #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4 ▶ if __name__ == '__main__':
5     with open("file.txt", 'r') as f:
6         content = f.read()
7     print(content)
```

Рисунок 3 – код программы ex.2.py



```
C:\Users\admin\anaconda3\python.exe C:/Users/admin/OPI/2.15/ex.2.py
Hi!

Process finished with exit code 0
```

Рисунок 4 – результат работы программы ex.2.py

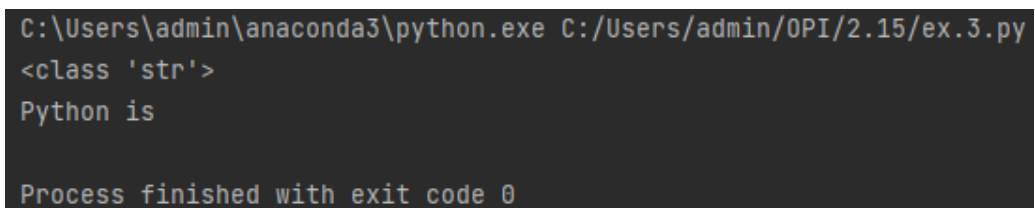


```

1  ▶  #!/usr/bin/env python3
2      # -*- coding: utf-8 -*-
3
4  ▶  if __name__ == '__main__':
5      with open("file2.txt", "r") as fileptr:
6          content = fileptr.read(10)
7          print(type(content))
8          print(content)

```

Рисунок 5 – код программы ex.3.py



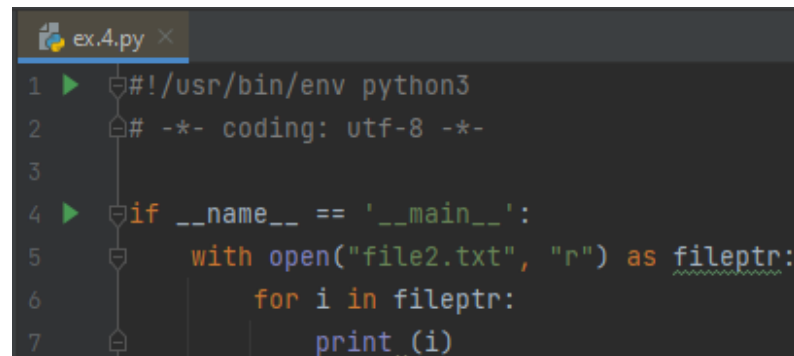
```

C:\Users\admin\anaconda3\python.exe C:/Users/admin/OPI/2.15/ex.3.py
<class 'str'>
Python is

Process finished with exit code 0

```

Рисунок 6 – результат работы программы ex.3.py

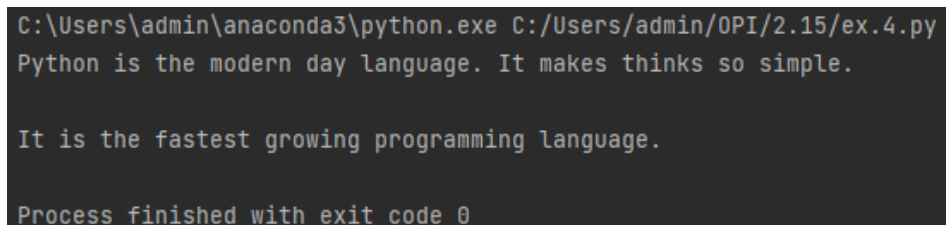


```

1  ▶  #!/usr/bin/env python3
2      # -*- coding: utf-8 -*-
3
4  ▶  if __name__ == '__main__':
5      with open("file2.txt", "r") as fileptr:
6          for i in fileptr:
7              print(i)

```

Рисунок 7 – код программы ex.4.py



```

C:\Users\admin\anaconda3\python.exe C:/Users/admin/OPI/2.15/ex.4.py
Python is the modern day language. It makes thinks so simple.

It is the fastest growing programming language.

Process finished with exit code 0

```

Рисунок 8 – результат работы программы ex.4.py

```

1 ▶ #!/usr/bin/env python3
2   #- coding: utf-8 -*-
3
4 ▶ if __name__ == '__main__':
5     with open("file2.txt", "r") as fileptr:
6         print("The filepointer is at byte :", fileptr.tell())
7         content = fileptr.read()
8         print("After reading, the filepointer is at:", fileptr.tell())

```

Рисунок 9 – код программы ex.5.py

```

C:\Users\admin\anaconda3\python.exe C:/Users/admin/OPI/2.15/ex.5.py
The filepointer is at byte : 0
After reading, the filepointer is at: 110

Process finished with exit code 0

```

Рисунок 10 – результат работы программы ex.5.py

```

1 ▶ #!/usr/bin/env python3
2   #- coding: utf-8 -*-
3
4 ▶ if __name__ == '__main__':
5     with open("file2.txt", "r") as fileptr:
6         fileptr.write(
7             "Python is the modern day language. It makes things so simple.\n"
8             "It is the fastest-growing programming language"
9         )

```

Рисунок 11 – код программы ex.6.py

```

ex.6.py × file2.txt ×
1 Python is the modern day language. It makes things so simple.
2 It is the fastest-growing programming language

```

Рисунок 12 – результат работы программы ex.6.py

```

1 ▶ #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4 ▶ if __name__ == '__main__':
5     with open("file2.txt", "a") as fileptr:
6         fileptr.write(
7             "Python has an easy syntax and user-friendly interaction."
8         )

```

Рисунок 13 – код программы ex.7.py

```

1 Python is the modern day language. It makes things so simple.
2 It is the fastest-growing programming language
3 Python has an easy syntax and user-friendly interaction.

```

Рисунок 14 – результат работы программы ex.7.py

```

1 ▶ #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4 ▶ if __name__ == '__main__':
5     with open("file2.txt", "r") as fileptr:
6         content1 = fileptr.readline()
7         content2 = fileptr.readline()
8         print(content1)
9         print(content2)

```

Рисунок 15 – код программы ex.8.py

```

C:\Users\admin\anaconda3\python.exe C:/Users/admin/OPI/2.15/ex.8.py
Python is the modern day language. It makes things so simple.

It is the fastest-growing programming language

Process finished with exit code 0

```

Рисунок 16 – результат работы программы ex.8.py

```

1 ▶ 1 #!/usr/bin/env python3
2   2 # -*- coding: utf-8 -*-
3
4 ▶ 4 if __name__ == '__main__':
5   5     with open("file2.txt", "r") as fileptr:
6   6         content = fileptr.readlines()
7   7         print(content)

```

Рисунок 17 – код программы ex.9.py

```

C:\Users\admin\anaconda3\python.exe C:/Users/admin/OPI/2.15/ex.9.py
['Python is the modern day language. It makes things so simple.\n', 'It is the fastest-growing programing language\n', 'Python has an easy syntax and user-friendly interaction.']
Process finished with exit code 0

```

Рисунок 18 – результат работы программы ex.9.py

```

1 ▶ 1 #!/usr/bin/env python3
2   2 # -*- coding: utf-8 -*-
3
4 ▶ 4 if __name__ == '__main__':
5   5     with open("newfile.txt", "x") as fileptr:
6   6         print(fileptr)
7   7     if fileptr:
8   8         print("File created successfully")

```

Рисунок 19 – код программы ex.10.py

```

C:\Users\admin\anaconda3\python.exe C:/Users/admin/OPI/2.15/ex.10.py
<_io.TextIOWrapper name='newfile.txt' mode='x' encoding='cp1251'>
File created successfully
Process finished with exit code 0

```

Рисунок 20 – результат работы программы ex.10.py

```

1 ▶ 1 #!/usr/bin/env python3
2   2 # -*- coding: utf-8 -*-
3
4 ▶ 4 if __name__ == '__main__':
5     5 with open("text.txt", "w", encoding="utf-8") as fileptr:
6         6 print(
7             7 "UTF-8 is a variable-width character encoding used for electronic "
8             8
9             9 "communication.",
10            10 file=fileptr
11        )
12    12 print(
13        13 "UTF-8 is capable of encoding all 1, 112, 064 valid character code "
14        14
15        15 "points.",
16        16 file=fileptr
17    )
18    18 print(
19        19 "In Unicode using one to four one-byte (8-bit) code units.",
20        20 file=fileptr
21    )

```

Рисунок 21 – код программы ex.11.py

```

1 ▶ 1 #!/usr/bin/env python3
2   2 # -*- coding: utf-8 -*-
3
4 ▶ 4 if __name__ == '__main__':
5     5 with open("text.txt", "r", encoding="utf-8") as f:
6         6 sentences = f.readlines()
7         7 for sentence in sentences:
8             8 if "," in sentence:
9                 9 print(sentence)

```

Рисунок 22 – код программы ex.12.py

```

C:\Users\admin\anaconda3\python.exe C:/Users/admin/OPI/2.15/ex.12.py
UTF-8 is capable of encoding all 1, 112, 064 valid character code points.

Process finished with exit code 0

```

Рисунок 23 – результат работы программы ex.12.py

```

1 ▶ #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4 ▶ if __name__ == '__main__':
5     with open("file2.txt", "r") as filptr:
6         print("The filepointer is at byte:", filptr.tell())
7         filptr.seek(10)
8         print("After reading, the filepointer is at:", filptr.tell())

```

Рисунок 24 – код программы ex.13.py

```

C:\Users\admin\anaconda3\python.exe C:/Users/admin/OPI/2.15/ex.13.py
The filepointer is at byte: 0
After reading, the filepointer is at: 10

Process finished with exit code 0

```

Рисунок 25 – результат работы программы ex.13.py

```

1 ▶ #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4
5     import os
6
7
8 ▶ if __name__ == "__main__":
9     os.rename("file2.txt", "file3.txt")

```

Рисунок 26 – код программы ex.14.py

```

1 ▶ #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4
5     import os
6
7
8 ▶ if __name__ == "__main__":
9     os.remove("newfile.txt")

```

Рисунок 27 – код программы ex.15.py


```

1  ▶  #!/usr/bin/env python3
2      # -*- coding: utf-8 -*-
3
4
5      import os
6
7
8  ▶  if __name__ == "__main__":
9      os.mkdir("new")

```

Рисунок 28 – код программы ex.16.py

```

1  ▶  #!/usr/bin/env python3
2      # -*- coding: utf-8 -*-
3
4
5      import os
6
7
8  ▶  if __name__ == "__main__":
9      path = os.getcwd()
10     print(path)

```

Рисунок 29 – код программы ex.17.py

```

C:\Users\admin\anaconda3\python.exe C:/Users/admin/OPI/2.15/ex.17.py
C:\Users\admin\OPI\2.15

Process finished with exit code 0

```

Рисунок 30 – результат работы программы ex.17.py

```

1  ▶  #!/usr/bin/env python3
2      # -*- coding: utf-8 -*-
3
4
5      import os
6
7
8  ▶  if __name__ == "__main__":
9      os.chdir("C:\\Windows")
10     print(os.getcwd())

```

Рисунок 31 – код программы ex.18.py

```
C:\Users\admin\anaconda3\python.exe C:/Users/admin/OPI/2.15/ex.18.py
C:\Windows

Process finished with exit code 0
```

Рисунок 32 – результат работы программы ex.18.py

```
1 ▶ ④#!/usr/bin/env python3
2    ④# -*- coding: utf-8 -*-
3
4
5    import os
6
7
8 ▶ ④if __name__ == "__main__":
9    os.rmdir("new")
```

Рисунок 33 – код программы ex.19.py

```
1 ▶ ④#!/usr/bin/env python3
2    ④# -*- coding: utf-8 -*-
3
4
5    import sys
6
7
8 ▶ ④if __name__ == "__main__":
9    print("Number of arguments:", len(sys.argv), "arguments")
10   print("Argument List:", str(sys.argv))
```

Рисунок 34 – код программы ex.20.py

```
C:\Users\admin\anaconda3\python.exe C:/Users/admin/OPI/2.15/ex.20.py
Number of arguments: 1 arguments
Argument List: ['C:/Users/admin/OPI/2.15/ex.20.py']

Process finished with exit code 0
```

Рисунок 35 – результат работы программы ex.20.py

```

1 ▶ #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4
5 import sys
6
7
8 ▶ if __name__ == "__main__":
9     for idx, arg in enumerate(sys.argv):
10         print(f"Argument #{idx} is {arg}")
11         print("No. of arguments passed is ", len(sys.argv))

```

Рисунок 36 – код программы ex.21.py

```

C:\Users\admin\anaconda3\python.exe C:/Users/admin/0PI/2.15/ex.21.py
Argument #0 is C:/Users/admin/0PI/2.15/ex.21.py
No. of arguments passed is  1

Process finished with exit code 0

```

Рисунок 37 – результат работы программы ex.21.py

```

1 ▶ #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3 import os
4 import secrets
5 import string
6 import sys
7
8 ▶ if __name__ == "__main__":
9     if len(sys.argv) != 2:
10         print("The password length is not given!", file=sys.stderr)
11         sys.exit(1)
12     chars = string.ascii_letters + string.punctuation + string.digits
13     length_pwd = int(sys.argv[1])
14     result = []
15     for _ in range(length_pwd):
16         idx = secrets.SystemRandom().randrange(len(chars))
17         result.append(f"Secret Password: {''.join(result)}")

```

Рисунок 38 – код программы ex.22.py

2. Индивидуальное задание №1.

```
1 ▶ #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4
5 import os
6
7
8 ▶ if __name__ == "__main__":
9     with open("file3.txt", "r") as file:
10         for line in file.readlines():
11             split_line = line.split()
12             split_line = iter(split_line)
13             print(''.join(
14                 [f'{two} {one}' for one, two in zip(split_line, split_line)]
15             ))
16
17 )
```

Рисунок 39 – код программы ind.1.py

```
C:\Users\admin\anaconda3\python.exe C:/Users/admin/OPI/2.15/ind.1.py
is Pythonmodern thelanguage. daymakes Itso things
is Itfastest-growing thelanguage programinghas Pythoneasy anand syntaxinteraction. user-friendly

Process finished with exit code 0
```

Рисунок 40 – результат работы программы ind.1.py

3. Индивидуальное задание №2.

```
1 ▶ #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4
5 import os
6
7
8 ▶ if __name__ == "__main__":
9     with open("text.txt", "r") as file:
10         string = ''
11         string = "".join(file.readlines())
12     words = string.split()
13     words.sort(key=len, reverse=True)
14     for word in words:
15         if len(word) == len(words[0]):
16             print(word)
```

Рисунок 41 – код программы ind.2.py

```
C:\Users\admin\anaconda3\python.exe C:/Users/admin/OPI/2.15/ind.2.py
wrjwfrfjwjfwjfff
qwhrwwfjwuwujwu
fghetlpmagetdnsp
Process finished with exit code 0
```

Рисунок 42 – результат работы программы ind.2.py

4. Задание с использованием модуля os.

```
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  import os
5
6
7  if __name__ == "__main__":
8      os.rename("dog.png", "cat.png")
9      os.remove("cat.png")
10     path = os.getcwd()
11     print(path)
```

Рисунок 43 – код программы os_task.py

```
C:\Users\admin\anaconda3\python.exe C:/Users/admin/OPI/2.15/os_task.py
C:\Users\admin\OPI\2.15
Process finished with exit code 0
```

Рисунок 44 – результат работы программы os_task.py

Ответы на вопросы:

1. Как открыть файл в языке Python только для чтения?

С помощью команды: `fileobj = open("file.txt", "r")`

2. Как открыть файл в языке Python только для записи?

С помощью команды: `fileobj = open("file.txt", "w")`

3. Как прочитать данные из файла в языке Python?

К примеру, с помощью данного набора команд:

```
with open("file.txt", 'r') as f:
```

```
content = f.read();
```

```
print(content)
```

Построчное чтение содержимого файла в цикле:

```
with open("file2.txt", "r") as fileptr:
```

```
for i in fileptr:
```

```
print(i)
```

Где `i` – одна строка файла.

Построчное чтение содержимого файла с помощью методов файлового объекта:

```
with open("file2.txt", "r") as fileptr:
```

```
content1 = fileptr.readline()
```

```
content2 = fileptr.readline()
```

```
print(content1)
```

```
print(content2)
```

Мы вызывали функцию `readline()` два раза, поэтому она считывает две строки из файла.

Чтение строк с помощью функции `readlines()`:

```
with open("file2.txt", "r") as fileptr:
```

```
content = fileptr.readlines()
```

```
print(content)
```

`readlines()` считывает строки в файле до его конца (EOF)

4. Как записать данные в файл в языке Python?

Чтобы записать текст в файл, нам нужно открыть файл с помощью метода `open` с одним из следующих режимов доступа:

'w': он перезапишет файл, если какой-либо файл существует. Указатель файла находится в начале файла.

'a': добавит существующий файл. Указатель файла находится в конце файла. Он создает новый файл, если файл не существует.

5. Как закрыть файл в языке Python?

После того, как все операции будут выполнены с файлом, мы должны закрыть его с помощью нашего скрипта Python, используя метод `close()`. Любая незаписанная информация уничтожается после вызова метода `close()` для файлового объекта.

```
fileobject.close()
```

Преимущество использования оператора `with` заключается в том, что он обеспечивает гарантию закрытия файла независимо от того, как закрывается вложенный блок. Всегда рекомендуется использовать оператор `with` для файлов. Если во вложенном блоке кода возникает прерывание, возврат или исключение, тогда он автоматически закрывает файл, и нам не нужно писать функцию `close()`. Это не позволяет файлу исказиться.

6. Изучите самостоятельно работу конструкции `with ... as`. Каково ее назначение в языке Python? Где она может быть использована еще, помимо работы с файлами?

Данная конструкция является менеджером контекста. Помимо файлов может использоваться в работе с базами данных:

```
def get_all_songs():  
    with sqlite3.connect('db/songs.db') as connection:  
        cursor = connection.cursor()  
        cursor.execute("SELECT * FROM songs ORDER BY id desc")  
        all_songs = cursor.fetchall()  
        return all_songs
```

7. Изучите самостоятельно документацию Python по работе с файлами.

Какие помимо рассмотренных существуют методы записи/чтения

информации из файла?

Есть возможность записать в файл большой объем данных, если он может быть представлен в виде списка строк.

```
with open("examp.le", "w") as f:
```

```
    f.writelines(list_of_strings)
```

Существует еще один, менее известный, способ, но, возможно, самый удобный из представленных. И как бы не было странно, он заключается в использовании функции `print()`. Сначала это утверждение может показаться странным, потому что общеизвестно, что с помощью нее происходит вывод в консоль. И это правда. Но если передать в необязательный аргумент `file` объект типа `io.TextIOWrapper`, каким и является объект файла, с которым мы работаем, то поток вывода функции `print()` перенаправляется из консоли в файл.

```
with open("examp.le", "w") as f:
```

```
    print(some_data, file=f)
```

С помощью `file.seek()` можно перемещать указатель в файле на определенное количество байтов.

8. Какие существуют, помимо рассмотренных, функции модуля `os` для работы с файловой системой?

Предположим, вы хотите создать не только одну папку, но и несколько вложенных:

```
# вернуться в предыдущую директорию
```

```
os.chdir("..")
```

```
# сделать несколько вложенных папок
```

```
os.makedirs("nested1/nested2/nested3")
```

Перемещение файлов

Функцию `os.replace()` можно использовать для перемещения файлов или каталогов:

```
# заменить (переместить) этот файл в другой каталог
```

```
os.replace("renamed-text.txt", "folder/renamed-text.txt")
```

Стоит обратить внимание, что это перезапишет путь, поэтому если в папке `folder` уже есть файл с таким же именем (`renamed-text.txt`), он будет перезаписан.

Список файлов и директорий

```
# распечатать все файлы и папки в текущем каталоге  
print("Все папки и файлы:", os.listdir())
```

Функция `os.listdir()` возвращает список, который содержит имена файлов в папке. Если в качестве аргумента не указывать ничего, вернется список файлов и папок текущего рабочего каталога:

```
Все папки и файлы: ['folder', 'handling-files', 'nested1', 'text.txt']
```

А что если нужно узнать состав и этих папок тоже? Для этого нужно использовать функцию `os.walk()`:

```
# распечатать все файлы и папки рекурсивно  
for dirpath, dirnames, filenames in os.walk("."):   
# перебрать каталоги  
for dirname in dirnames:  
    print("Каталог:", os.path.join(dirpath, dirname))  
# перебрать файлы  
for filename in filenames:  
    print("Файл:", os.path.join(dirpath, filename))
```

`os.walk()` — это генератор дерева каталогов. Он будет перебирать все переданные составляющие. Здесь в качестве аргумента передано значение «.», которое обозначает верхушку дерева:

```
Каталог: .\folder  
Каталог: .\handling-files  
Каталог: .\nested1  
Файл: .\text.txt  
Файл: .\handling-files\listing_files.py  
Файл: .\handling-files\README.md  
Каталог: .\nested1\nested2  
Каталог: .\nested1\nested2\nested3
```

Метод `os.path.join()` был использован для объединения текущего пути с именем файла/папки.

Получение информации о файлах

Для получения информации о файле в ОС используется функция `os.stat()`, которая выполняет системный вызов `stat()` по выбранному пути:

```
open("text.txt", "w").write("Это текстовый файл")
```

```
# вывести некоторые данные о файле
```

```
print(os.stat("text.txt"))
```

Это вернет кортеж с отдельными метриками. В их числе есть следующие:

`st_size` — размер файла в байтах

`st_atime` — время последнего доступа в секундах (временная метка)

`st_mtime` — время последнего изменения

`st_ctime` — в Windows это время создания файла, а в Linux — последнего изменения метаданных

Для получения конкретного атрибута нужно писать следующим образом:

```
# например, получить размер файла
```

```
print("Размер файла:", os.stat("text.txt").st_size)
```

Вывод:

Размер файла: 19