

**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 по дисциплине основы программной
инженерии**

Выполнил:
Выходцев Егор Дмитриевич,
2 курс, группа ПИЖ-б-о-20-1,

Проверил:
Доцент кафедры инфокоммуникаций,
Воронкин Р.А.

Ставрополь, 2022 г

1. Примеры из методических указаний

```
e1.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()
```

```
e1 (1) x
C:\Users\Evil\PycharmProjects\PyCharm\
file is opened successfully

Process finished with exit code 0
```

```
e2.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)
8
```

```
e2 x
C:\Users\Evil\PycharmProjects\LB5\ve
Hello

Process finished with exit code 0
```

```
e3.py x
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)
9
```

```
e3 x
C:\Users\Evil\PycharmProjects\LB5\ve
<class 'str'>
Python is
Process finished with exit code 0
```

```
e4.py x
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)
8
```

```
e4 x
C:\Users\Evil\PycharmProjects\LB5\venv\Scripts\python.exe "C:/Users/Evil/PycharmProjects/LR #3/e4.py"
Python is the modern day language. It makes things so simple.

It is the fastest-growing programing language Python has an easy syntax and user-friendly interaction.

Process finished with exit code 0
```

```
e5.py x
1 ▶ #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4
5 ▶ if __name__ == "__main__":
6     with open("file2.txt", "r") as fileptr:
7         print("The filepointer is at byte :", fileptr.tell())
8         content = fileptr.read()
9         print("After reading, the filepointer is at:", fileptr.tell())
10
```

```
e5 x
C:\Users\Evil\PycharmProjects\LB5\venv\Scripts\python.exe
The filepointer is at byte : 0
After reading, the filepointer is at: 165
Process finished with exit code 0
```

```
ex1.py x
1 ▶ #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4 ▶ if __name__ == '__main__':
5     with open("file2.txt", "w") 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         )
10
```

```
ex1.py x file2.txt x
1 Python is the modern day language. It makes things so simple.
2 It is the fastest-growing programming language
```

```
ex2.py x
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         )
9
```

```
ex2.py x file2.txt x
1 Python is the modern day language. It makes things so simple.
2 It is the fastest-growing programming languagePython has an easy syntax and user-friendly interaction.
```

```
ex3.py x
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)
10
```

```
ex3 x
C:\Users\Evil\PycharmProjects\LB5\venv\Scripts\python.exe "C:/Users/Evil/PycharmProjects/LR #3/ex3.py"
Python is the modern day language. It makes things so simple.

It is the fastest-growing programming languagePython has an easy syntax and user-friendly interaction.

Process finished with exit code 0
```

```
ex4.py x
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.readlines()
7         print(content)
8
```

```
ex4
C:\Users\Evil\PycharmProjects\LB5\venv\Scripts\python.exe "C:/Users/Evil/PycharmProjects/LR #3/ex4.py"
['Python is the modern day language. It makes things so simple.\n', 'It is the fastest-growing programming languagePython has an easy syntax and user-friendly interaction.']
Process finished with exit code 0
```

```
ex5.py x
1  ▶  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  ▶  if __name__ == '__main__':
5      with open("newfile.txt", "x") as fileptr:
6          print(fileptr)
7      if fileptr:
8          print("File created successfully")
9
```

```
ex5 x
C:\Users\Evil\PycharmProjects\LB5\venv\Scripts\python.exe "C:/Users/Evil/PycharmProjects/LB5/ex5.py"
<_io.TextIOWrapper name='newfile.txt' mode='x' encoding='cp1251'>
File created successfully
Process finished with exit code 0
```

```
ex6.py x
1 ▶ #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4
5 ▶ if __name__ == "__main__":
6     with open("text.txt", "w", encoding="utf-8") as fileptr:
7         print(
8             "UTF-8 is a variable-width character encoding used for electronic "
9
10            "communication.",
11            file=fileptr
12        )
13    print(
14        "UTF-8 is capable of encoding all 1,112,064 valid character code "
15
16        "points.",
17
18        file=fileptr
19    )
20    print(
21        "In Unicode using one to four one-byte (8-bit) code units.",
22        file=fileptr
23    )
24
```

```
ex6.py x ex7.py x
1 ▶ #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4
5 ▶ if __name__ == "__main__":
6     with open("text.txt", "r", encoding="utf-8") as f:
7         sentences = f.readlines()
8         for sentence in sentences:
9             if "," in sentence:
10                 print(sentence)
11
```

```
ex7 x
C:\Users\Evil\PycharmProjects\LB5\venv\Scripts\python.exe "C:/Users/Evil/Py
UTF-8 is capable of encoding all 1,112,064 valid character code points.

Process finished with exit code 0
```

```
ex8.py x
1 ▶ #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4
5 ▶ if __name__ == "__main__":
6     with open("file2.txt", "r") as fileptr:
7         print("The filepointer is at byte :", fileptr.tell())
8         fileptr.seek(10)
9         print("After reading, the filepointer is at:", fileptr.tell())
10
```

```
ex8 x
C:\Users\Evil\PycharmProjects\LB5\venv\Scr
The filepointer is at byte : 0
After reading, the filepointer is at: 10

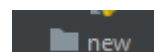
Process finished with exit code 0
```

```
ex9.py x
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")
10
```



```
ex10.py x
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")
10
```

```
ex11.py x
1 ▶ #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4
5 import os
6
7
8 ▶ if __name__ == "__main__":
9     os.mkdir("new")
10
```



```
ex12.py x
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)
11
```

```
ex12 x
C:\Users\Evil\PycharmProjects\LB5\venv\S
C:\Users\Evil\PycharmProjects\LR #3

Process finished with exit code 0
```

```
ex13.py x
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())
11
```

```
ex13 x
↑ C:\Users\Evil\PycharmProjects\LB5\ve
↓ C:\Windows

Process finished with exit code 0
```

```
ex14.py x
1 ▶ #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4
5 import os
6
7
8 ▶ if __name__ == "__main__":
9     os.rmdir("new")
10
```

```
ex15.py ×
1 ▶ #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4
5 import sys
6
7
8 ▶ if __name__ == "__main__":
9     print("Number of arguments:", len(sys.argv), "arguments")
10     print("Argument List:", str(sys.argv))
11
```

```
ex15 ×
C:\Users\Evil\PycharmProjects\LB5\venv\Scripts\python.exe "C:/Users/Evil/PycharmProjects/LR #3/ex15.py"
Number of arguments: 1 arguments
Argument List: ['C:/Users/Evil/PycharmProjects/LR #3/ex15.py']
Process finished with exit code 0
```

```
ex16.py ×
1 ▶ #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3 import sys
4
5 ▶ if __name__ == "__main__":
6     for idx, arg in enumerate(sys.argv):
7         print(f"Argument #{idx} is {arg}")
8         print("No. of arguments passed is ", len(sys.argv))
9
```

```
ex16 ×
C:\Users\Evil\PycharmProjects\LB5\venv\Scripts\python.exe "C:/Users/Evil/PycharmProjects/LR #3/ex16.py"
Argument #0 is C:/Users/Evil/PycharmProjects/LR #3/ex16.py
No. of arguments passed is 1
Process finished with exit code 0
```

```
ex17.py x
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(chars[idx])
18         print(f"Secret Password: {''.join(result)}")
19
```

2. Решение индивидуальных заданий

2.1 Индивидуальное задание №1 (рис. 1-3).

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

Рисунок 1 – Код программы

```
main.py x file3.txt x
1 Python is the modern day language. It makes things so simple.
2 It is the fastest-growing programming language Python has an easy syntax and user-friendly interaction.
```

Рисунок 2 – Содержимое файла file3.txt

```
main x
C:\Users\Evil\PycharmProjects\LB5\venv\Scripts\python.exe "C:/Users/Evil/PycharmProjects/LR #3/idz1/main
is Python modern the language. day makes It so things
is It fastest-growing the language programming has Python easy an and syntax interaction. user-friendly
Process finished with exit code 0
```

Рисунок 3 – Результат работы программы

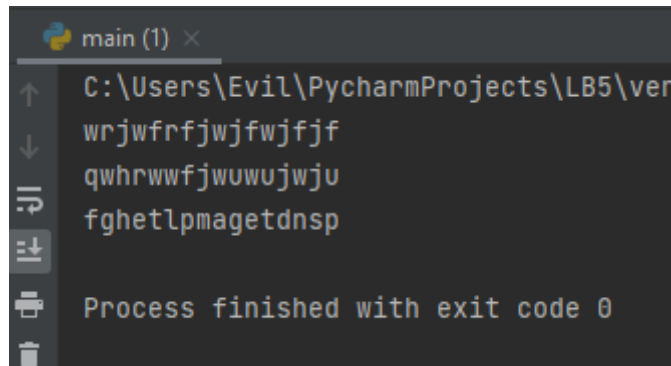
2.2 Индивидуальное задание №2 (рис 4-6).

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

Рисунок 4 – Код программы

```
main.py x text.txt x
1 wfajrwej fwjfwj ww
2 wrjwfrfjwjfwjfff
3 qwhrwwfjwuwjwju
4 fwifwi fwjw jwkoqwkqj
5 w
6 wfrwgf w fghetlpmagetdnsp q f
```

Рисунок 5 – Содержимое файла text.txt

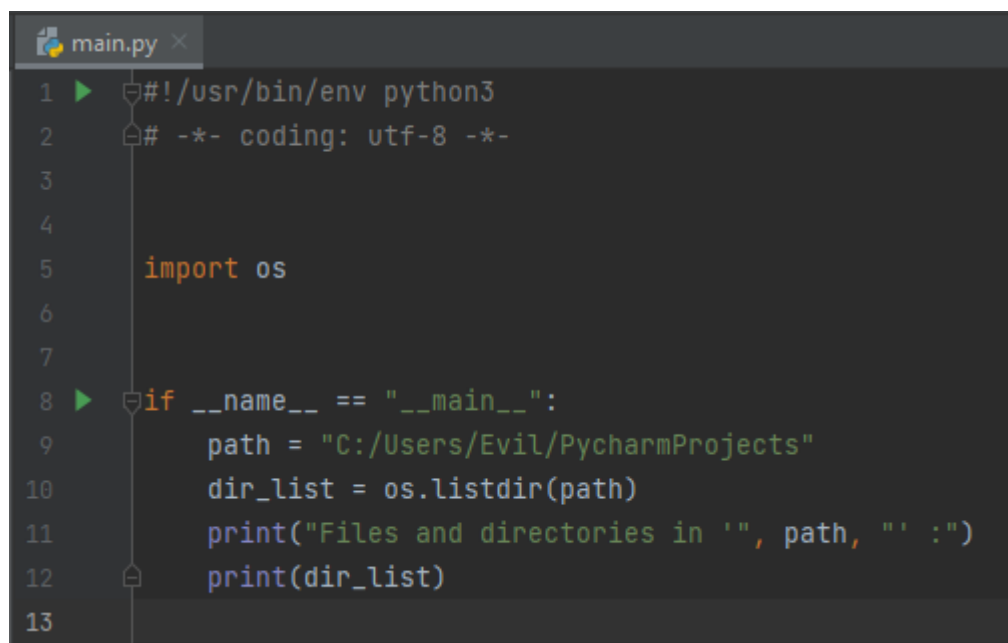


```
main (1) x
C:\Users\Evil\PycharmProjects\LB5\venv\Scripts\python.exe
wrjwfrfjwjfwjfjf
qwhrwwfjwuwjwju
fghetlpmagetdnsp
Process finished with exit code 0
```

Рисунок 6 – Результат работы программы

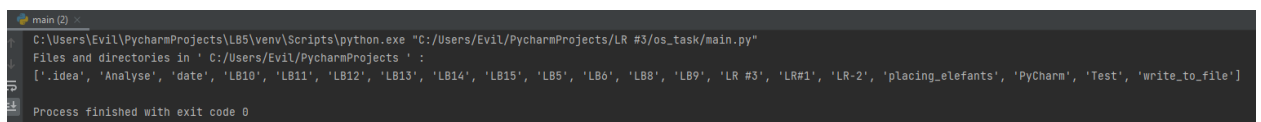
3. Задача с использованием модуля os (рис 7,8).

Условие: Вывод всех директорий в папке PyCharmProjects



```
main.py x
1 > #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4
5 import os
6
7
8 > if __name__ == "__main__":
9     path = "C:/Users/Evil/PycharmProjects"
10     dir_list = os.listdir(path)
11     print("Files and directories in '", path, "':")
12     print(dir_list)
13
```

Рисунок 7 – Код программы



```
main (2) x
C:\Users\Evil\PycharmProjects\LB5\venv\Scripts\python.exe "C:/Users/Evil/PycharmProjects/LR #3/os_task/main.py"
Files and directories in ' C:/Users/Evil/PycharmProjects ':
['.idea', 'Analyse', 'date', 'LB10', 'LB11', 'LB12', 'LB13', 'LB14', 'LB15', 'LB6', 'LB8', 'LB9', 'LR #3', 'LR#1', 'LR-2', 'placing_elephants', 'PyCharm', 'Test', 'write_to_file']
Process finished with exit code 0
```

Рисунок 8 – Результат выполнения программы

4.. Ответы на контрольные 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.

Пример:

```
with open("file2.txt", "w") as fileptr:
    fileptr.write(
        "Python is the modern day language. It makes things so simple.\n"
        "It is the fastest-growing programing language"
    )
```

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)
```

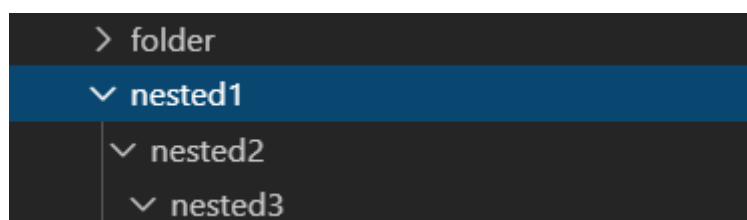
С помощью `file.seek()` можно перемещать указатель в файле на определенное количество байтов.

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")
```

Это создаст три папки рекурсивно, как показано на следующем изображении:



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