

**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 №2.15 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:

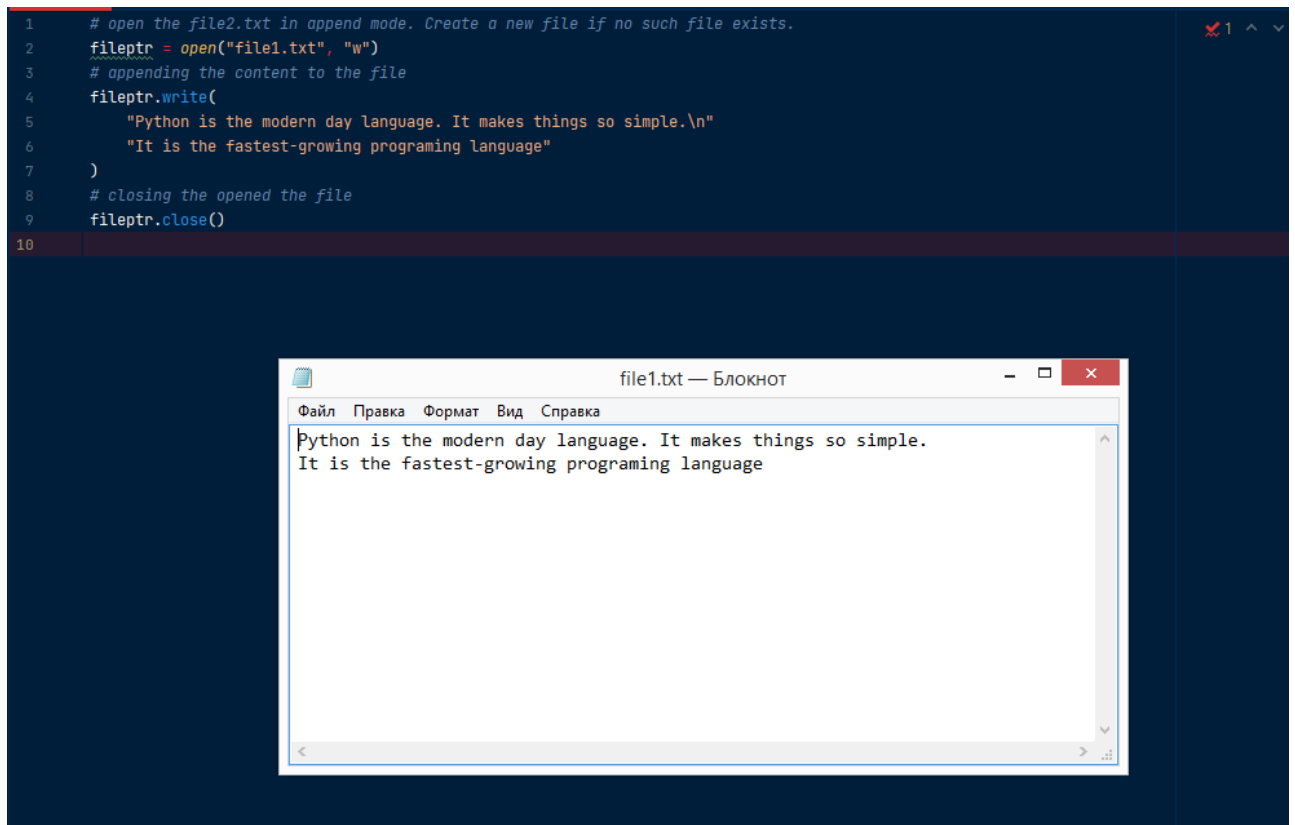


Рисунок 1 - Пример №1



Рисунок 2 - Пример №2

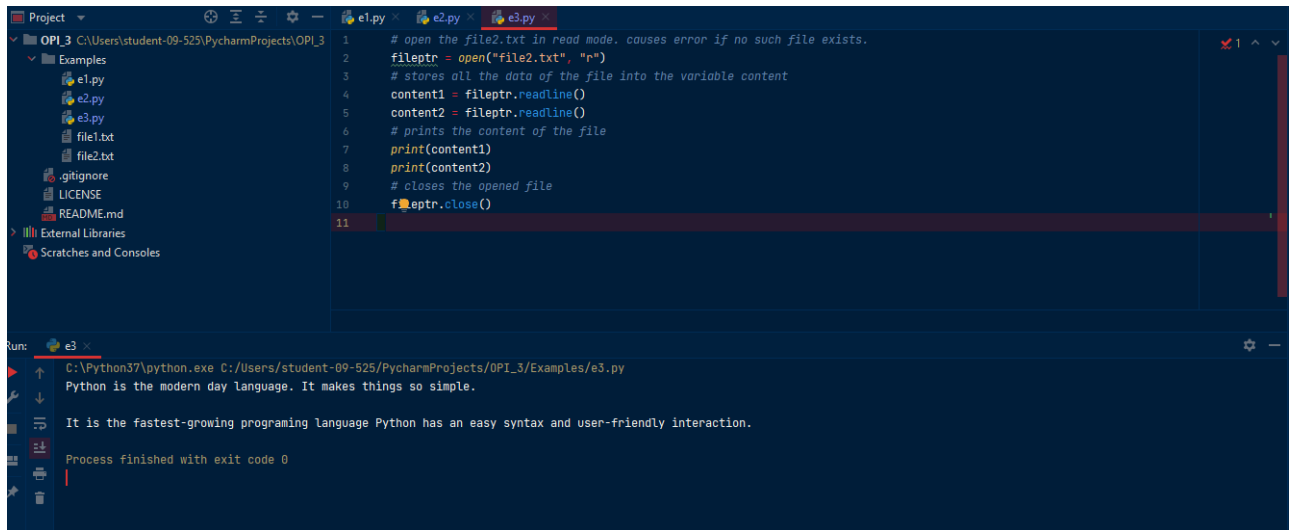


Рисунок 3 - Пример №3

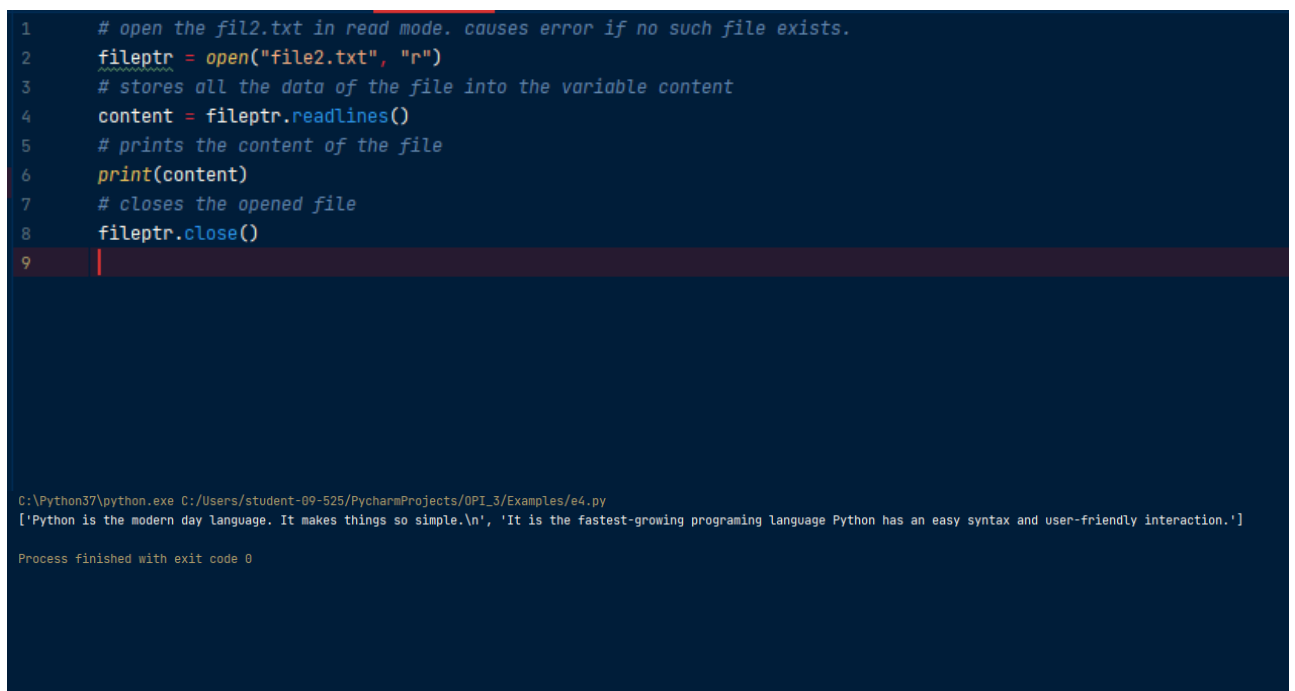


Рисунок 4 - Пример №4

```

1 # open the newfile.txt in read mode. causes error if no such file exists.
2 fileptr = open("newfile.txt", "x")
3 print(fileptr)
4 if fileptr: print("File created successfully")
5 # closes the opened file
6 fileptr.close()
7

```

```

C:\Python37\python.exe C:/Users/student-09-525/PycharmProjects/OPI_3/Examples/e5.py
<_io.TextIOWrapper name='newfile.txt' mode='x' encoding='cp1251'>
File created successfully

Process finished with exit code 0








```

Рисунок 5 - Пример №5

```

1 ▶ #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4 ▶ if __name__ == "__main__":
5     # open the text.txt in append mode. Create a new file if no such file exists.
6     with open("text.txt", "w", encoding="utf-8") as fileptr:
7         # appending the content to the file
8         print(
9             "UTF-8 is a variable-width character encoding used for electronic communication.",
10            file=fileptr
11        )
12        print(
13            "UTF-8 is capable of encoding all 1,112,064 valid character code points.",
14            file=fileptr
15        )
16        print(
17            "In Unicode using one to four one-byte (8-bit) code units.",
18            file=fileptr
19        )
20

```

 e1.py ×
  e2.py ×
  e3.py ×
  e4.py ×
  e5.py ×
  e6.py ×
  text.txt ×

```

1 UTF-8 is a variable-width character encoding used for electronic communication.
2 UTF-8 is capable of encoding all 1,112,064 valid character code points.
3 In Unicode using one to four one-byte (8-bit) code units.
4

```

Рисунок 6 - Пример №6

```

1 ▶ #!/usr/bin/env python3
2   # -*- coding: utf-8 -*-
3
4 ▶ if __name__ == "__main__":
5     with open("text.txt", "r", encoding="utf-8") as f:
6         sentences = f.readlines()
7
8     # Вывод предложений с запятыми.
9     for sentence in sentences:
10         if "," in sentence:
11             print(sentence)
12

```

C:\Python37\python.exe C:/Users/student-09-525/PycharmProjects/OPI_3/Examples/e7.py
UTF-8 is capable of encoding all 1,112,064 valid character code points.

Process finished with exit code 0

Рисунок 7 - Пример №7

```

1   # open the file file2.txt in read mode
2   with open("file2.txt", "r") as fileptr:
3       # initially the filepointer is at 0
4       print("The filepointer is at byte :", fileptr.tell())
5
6       # changing the file pointer location to 10.
7       fileptr.seek(10);
8
9       # tell() returns the location of the fileptr.
10      print("After reading, the filepointer is at:", fileptr.tell())
11

```

C:\Python37\python.exe C:/Users/student-09-525/PycharmProjects/OPI_3/Examples/e8.py
The filepointer is at byte : 0
After reading, the filepointer is at: 10

Process finished with exit code 0

Рисунок 8 - Пример №8

```
1 import os
2 # rename file2.txt to file3.txt
3 os.rename("file2.txt", "file3.txt")
4
```

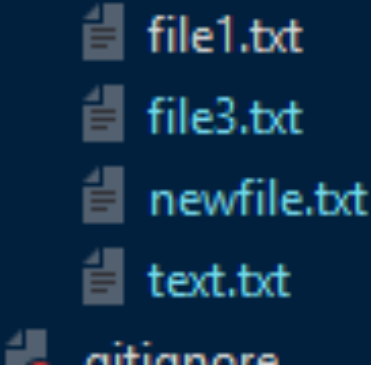


Рисунок 9 - Пример №9

```
1 import os
2 # deleting the file named file3.txt
3 os.remove("file3.txt")
4
```

Рисунок 10 - Пример №10

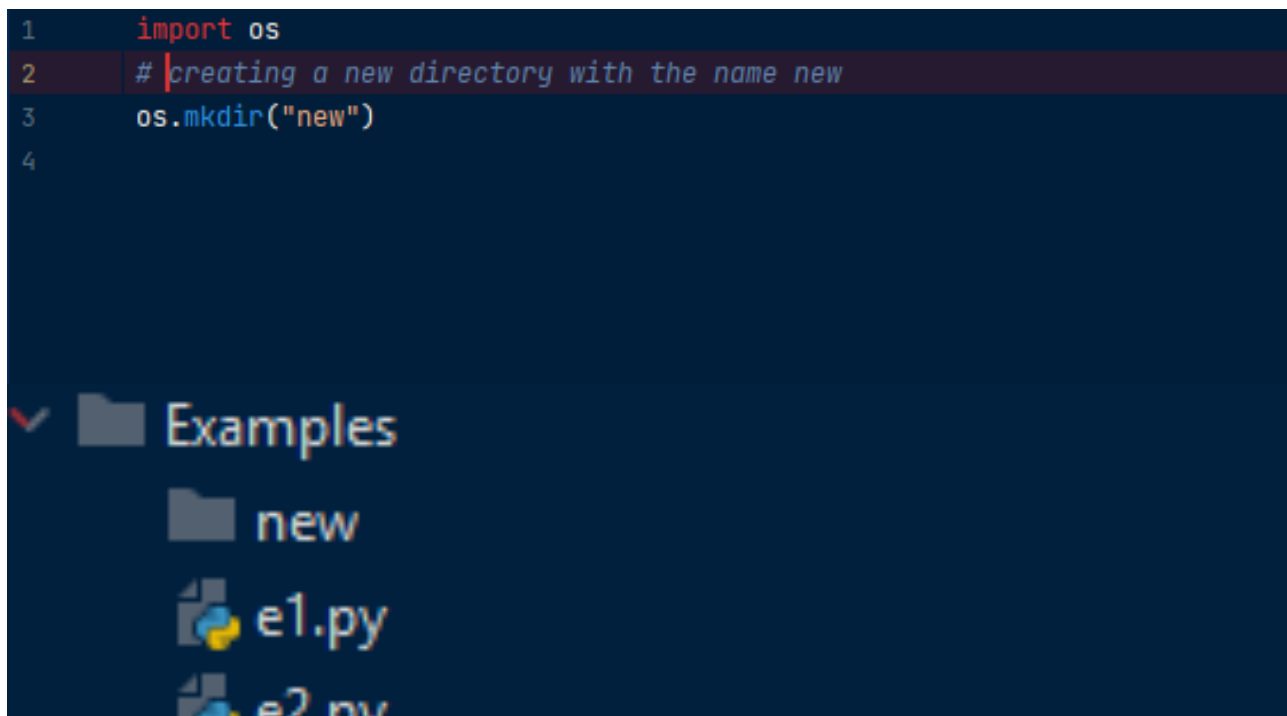


Рисунок 11 - Пример №11

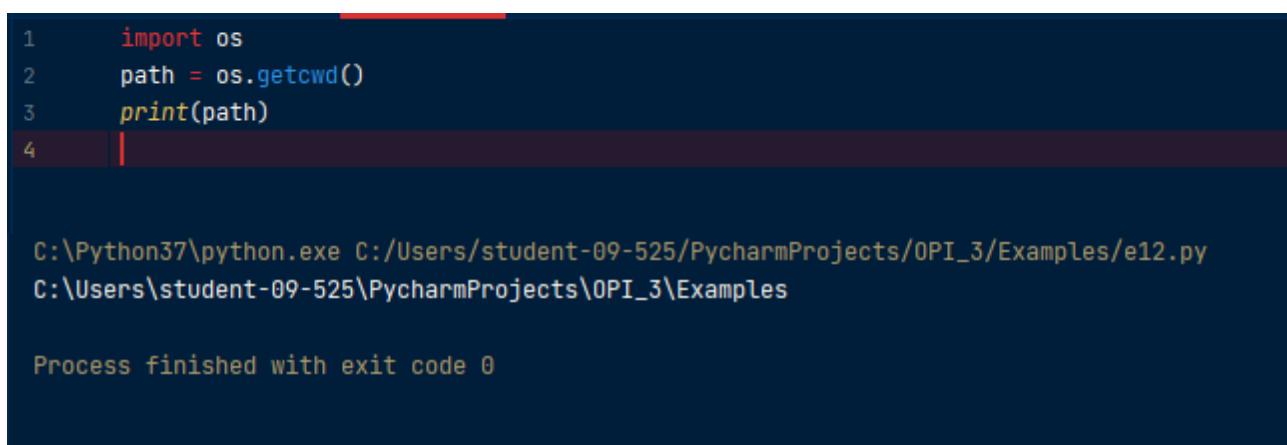


Рисунок 12 - Пример №12

```
1 import os
2 # Changing current directory with the new directory
3 os.chdir("C:\\Windows")
4
5 # It will display the current working directory
6 print(os.getcwd())
7
```

C:\Python37\python.exe C:/Users/student-09-525/PycharmProjects/OPI_3/Examples/e13.py
C:\Windows

Process finished with exit code 0

Рисунок 13 - Пример №13

```
1 import os
2
3 # removing the new directory
4 os.rmdir("new")
5
```

Рисунок 14 - Пример №4

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

Рисунок 15 - Пример № 15


```

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

```

Рисунок 16 - Пример №16

```

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

```

Рисунок 17 - Пример №17

ИНДИВИДУАЛЬНОЕ ЗАДАНИЕ:

```
1 import re
2 f = open('text.txt', encoding='UTF')
3
4 while True:
5     line = f.readline()
6     if len(line) == 0: # Нулевая длина обозначает конец файла
7         break
8     if not re.findall(r'\d{2}', line):
9         print(line)
10
11 f.close() # закрываем файл
12
```

C:\Python37\python.exe C:/Users/student-09-525/PycharmProjects/OPI_3/Examples/idz1.py
UTF-8 is a variable-width character encoding used for electronic communication.

In Unicode using one to four one-byte (8-bit) code units.

Process finished with exit code 0

Рисунок 18 - Индивидуальное задание №1

```
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 = " ".join(file.readlines())
8     words = string.split()
9     words.sort(key=len, reverse=True)
10    for word in words:
11        if len(word) == len(words[0]):
12            print(word)
13
```

C:\Python37\python.exe C:/Users/student-09-525/PycharmProjects/OPI_3/Examples/idz2.py
variable-width
communication.

Process finished with exit code 0

Рисунок 19 - Индивидуальное задание №2

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.

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 programming 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")
```

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