

Министерство науки и высшего образования Российской Федерации  
Федеральное государственное автономное образовательное учреждение  
высшего образования  
«СЕВЕРО - КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития  
Кафедра инфокоммуникаций

**ОТЧЕТ**  
**ПО ЛАБОРАТОРНОЙ РАБОТЕ №1**  
**дисциплины «Анализ данных»**

**Вариант №18**

Выполнил:  
Кулешов Олег Иванович  
2 курс, группа ИВТ-б-о-22-1,  
09.03.01 «Информатика и  
вычислительная техника»,  
направленность (профиль)  
«Программное обеспечение средств  
вычислительной техники и  
автоматизированных систем», очная  
форма обучения

---

(подпись)

Руководитель практики:  
Воронкин Р. А.

---

(подпись)

Отчет защищен с оценкой \_\_\_\_\_ Дата защиты \_\_\_\_\_

Ставрополь, 2024 г.

## Тема: Работа с файлами в языке Python

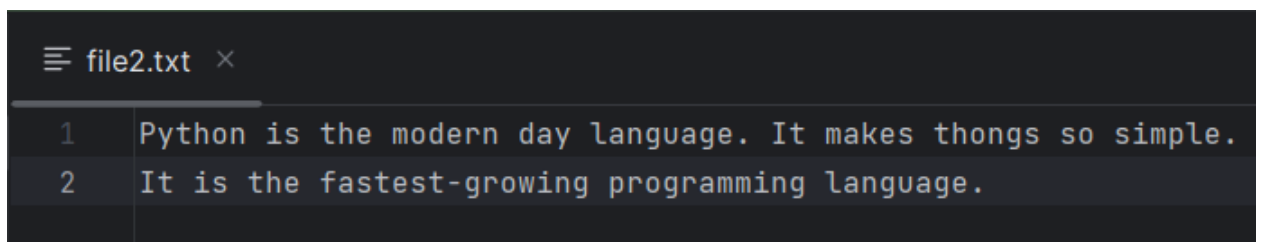
**Цель работы:** приобретение навыков по работе с текстовыми файлами при написании программ с помощью языка программирования Python версии 3.x, изучение основных методов модуля os для работы с файловой системой, получение аргументов командной строки.

### Листинг примера №1:

```
# open the file2.txt in append mode. Create a new file if no such file exists.
fileptr = open("file2.txt", "a")

# appending the content to the file
fileptr.write(
    "Python is the modern day language. It makes things so simple.\n"
    "It is the fastest-growing programming language"
)

# closing the opened the file
fileptr.close()
```



file2.txt x

1	Python is the modern day language. It makes thongs so simple.
2	It is the fastest-growing programming language.

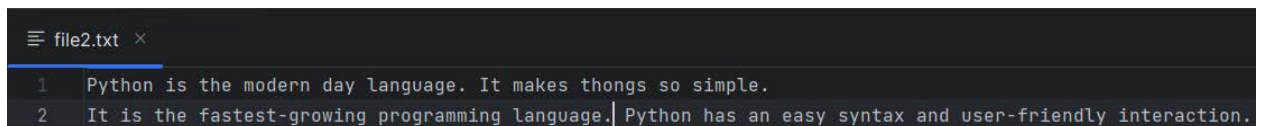
Рисунок 1. Результат

### Листинг примера №2:

```
# open the file.txt in write mode.
fileptr = open("file2.txt", "a")

# overwriting the content of the file
fileptr.write(" Python has an easy syntax and user-friendly interaction.")

#closing the opened file
fileptr.close()
```



file2.txt x

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

Рисунок 2. Результат

### Листинг примера №3:

```
# open file2.txt in read mode. causes error if no such file exists
fileptr = open("file2.txt", "r")

#stores all the data of the file into the variable content
content1 = fileptr.readline()
content2 = fileptr.readline()
```

```
# prints the content of the file
print(content1)
print(content2)

# closes the opened file
fileptr.close()
```

```
C:\Users\User\.conda\envs\Data_Analysis_1\python.exe C:\Users\User\PycharmProjects\Data_Analysis_1\example_3.py
Python is the modern day language. It makes things so simple.

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

Process finished with exit code 0
```

Рисунок 3. Результат

#### Листинг примера №4:

```
# open the file2.txt in read mode. causes error if no such file exists.
fileptr = open("file2.txt", "r")

# stores all the data of the file into the variable content
content = fileptr.readlines()
# prints the content of the file
print(content)

# closes the opened file
fileptr.close()
```

```
C:\Users\User\.conda\envs\Data_Analysis_1\python.exe C:\Users\User\PycharmProjects\Data_Analysis_1\example_4.py
['Python is the modern day language. It makes things so simple.\n', 'It is the fastest-growing programming language Python has an easy syntax and user-friendly interaction.\n', 'Python has an easy syntax and user-friendly interaction.']

Process finished with exit code 0
```

Рисунок 4. Результат

#### Листинг примера №5:

```
# open the newfile.txt in read mode. causes error if no such file exists.
fileptr = open("newfile.txt", "x")
print(fileptr)

if fileptr:
    print("File created successfully")

# closes the opened file
fileptr.close()
```

```
C:\Users\User\.conda\envs\Data_Analysis_1\python.exe C:\Users\User\PycharmProjects\Data_Analysis_1\example_5.py
<_io.TextIOWrapper name='newfile.txt' mode='x' encoding='cp1251'>
File created successfully

Process finished with exit code 0
```

Рисунок 5. Результат

### Листинг примера №6:

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

if __name__ == "__main__":
    # open the text.txt in append mode. Create a new file no such file
    exists.
    with open("text.txt", "w", encoding="utf-8") as fileptr:
        # appending the content to the file
        print(
            "UTF-8 is a variable-width character encoding used for electronic
communication",
            file=fileptr
        )
        print(
            "UTF-8 is capable of encoding all 1,112,064 valid character code
points.",
            file=fileptr
        )
        print(
            "In Unicode using one to four one-byte (8-bit) code units.",
            file=fileptr
        )
```

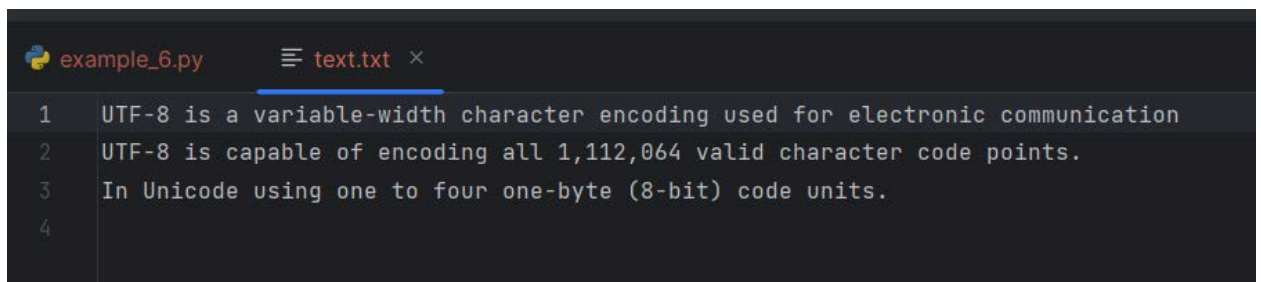


Рисунок 6. Результат

### Листинг примера №7:

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

if __name__ == "__main__":
    with open("text.txt", "r", encoding="utf-8") as fileptr:
        sentences = fileptr.readlines()

    # Вывод предложений с запятыми.
    for sentence in sentences:
        if "," in sentence:
            print(sentence)
```

```
C:\Users\User\.conda\envs\Data_Analysis_1\python.exe C:\Users\User\PycharmProjects\Data_Analysis_1\example_7.py
UTF-8 is capable of encoding all 1,112,064 valid character code points.

Process finished with exit code 0
```

Рисунок 7. Результат

### Листинг примера №8:

```
# open the file file2.txt in read mode
with open("file2.txt", "r") as fileptr:
    #initially the filepointer is at 0
    print("The filepointer is at byte :", fileptr.tell())

    #changing the file pointer location to 10.
    fileptr.seek(10);

    #tell() returns the location of the fileptr.
    print("After reading, the filepointer is at:", fileptr.tell())
```

```
C:\Users\User\.conda\envs\Data_Analysis_1\python.exe C:\Users\User\PycharmProjects\Data_Analysis_1\example_8.py
The filepointer is at byte : 0
After reading, the filepointer is at: 10

Process finished with exit code 0
```

### Рисунок 8. Результат

### Листинг примера №9:

```
import os

# rename file2.txt to file3.txt
os.rename("file2.txt", "file3.txt")
```

В результате файл в текущей рабочей папке с именем file2.txt переименовался в file3.txt.

### Листинг примера №10:

```
import os

# deleting the file named file3.txt
os.remove("file3.txt")
```

В результате в текущей рабочей папке файл с именем file3.txt удалился

### Листинг примера №11:

```
import os

#creating a new directory with the name new
os.mkdir("new")
```

В результате в текущей рабочей папке будет создан каталог с именем “new”.

### Листинг примера №12:

```
import os

path = os.getcwd()
print(path)
```

```
C:\Users\User\.conda\envs\Data_Analysis_1\python.exe C:\Users\User\PycharmProjects\Data_Analysis_1\example_12.py
C:\Users\User\PycharmProjects\Data_Analysis_1

Process finished with exit code 0
```

Рисунок 9. Результат (вывод полного пути проекта)

### Листинг примера №13:

```
import os

# Changing current directory with the new directory
os.chdir("C:\\Windows")
#It will display the current working directory
print(os.getcwd())
```

```
C:\Users\User\.conda\envs\Data_Analysis_1\python.exe
C:\Windows

Process finished with exit code 0
```

Рисунок 10. Результат (сменим текущий каталог на C:\Windows и выведем новое имя текущего каталога)

### Листинг примера №14:

```
import os

# removing the new directory
os.rmdir("new")
```

В результате данного кода удаляется директория с именем “new” в текущем рабочем каталоге.

### Листинг примера №15:

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

import sys

if __name__ == "__main__":
    print("Number of arguments:", len(sys.argv), "arguments")
    print("Argument List:", str(sys.argv))
```

```
(Data_Analysis_1) PS C:\Users\User\PycharmProjects\Data_Analysis_1> python example_15.py arg1 arg2 arg3
Number of arguments: 4 arguments
Argument List: ['example_15.py', 'arg1', 'arg2', 'arg3']
(Data_Analysis_1) PS C:\Users\User\PycharmProjects\Data_Analysis_1>
```

Рисунок 11. Результат

### Листинг примера №16:

```
#!/usr/bin/env python3
# __ coding: utf-8 __

import sys

if __name__ == "__main__":
    for idx, arg in enumerate(sys.argv):
        print(f"Argument #{idx} is {arg}")
    print("No. of arguments passed is ", len(sys.argv))
```

```
(Data_Analysis_1) PS C:\Users\User\PycharmProjects\Data_Analysis_1> python example_16.py Knowledge Hut 21
Argument #0 is example_16.py
Argument #1 is Knowledge
Argument #2 is Hut
Argument #3 is 21
No. of arguments passed is 4
(Data_Analysis_1) PS C:\Users\User\PycharmProjects\Data_Analysis_1>
```

Рисунок 12. Результат

### Листинг примера №17:

```
#!/usr/bin/env python3
# __ coding: utf-8 __

import os
import secrets
import string
import sys

if __name__ == "__main__":
    if len(sys.argv) != 2:
        print("The password length is not given!", file=sys.stderr)

    chars = string.ascii_letters + string.punctuation + string.digits
    length_pwd = int(sys.argv[1])

    result = []
    for _ in range(length_pwd):
        idx = secrets.SystemRandom().randrange(len(chars))
        result.append(chars[idx])

    print(f"Secret Password: {''.join(result)}")
```

```
(Data_Analysis_1) PS C:\Users\User\PycharmProjects\Data_Analysis_1> python example_17.py 12
Secret Password: e3:084xeLsG~
(Data_Analysis_1) PS C:\Users\User\PycharmProjects\Data_Analysis_1>
```

Рисунок 13. Результат

**Задача №1.** Написать программу, которая считывает текст из файла и выводит на экран сначала предложения, начинающиеся с однобуквенных слов, а затем все остальные.

### Листинг программы:

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

if __name__ == "__main__":
    with open('for_task_1.txt', 'r', encoding='utf-8') as file:
        sentences_one_letter = []
        other_sentences = []

        for line in file:
            sentence = line.strip()
            words = sentence.split()

            if len(words) > 0:
                if len(words[0]) == 1:
                    sentences_one_letter.append(sentence)
                else:
                    other_sentences.append(sentence)

        print("Предложения, начинающиеся с одной буквы:")
        for sentence in sentences_one_letter:
            print(sentence)

        print("\nОстальные предложения:")
        for sentence in other_sentences:
            print(sentence)
```

```
C:\Users\User\.conda\envs\Data_Analysis_1\python.exe C:\Users\User
Предложения, начинающиеся с одной буквы:
В те дни в таинственных долинах,
И славу нашей старины,
И сердца трепетные сны.

Остальные предложения:
Весной, при кликах лебединых,
Близ вод, сиявших в тишине,
Являться муза стала мне.
Моя студенческая келья
Вдруг озарилась: муза в ней
Открыла пир молодых затей,
Воспела детские веселья,

Process finished with exit code 0
```

Рисунок 14. Результат



**Задача №2.** Продолжаем тему операционных систем на базе Unix, в которых обычно также есть утилита с названием `cat`, что является сокращением от `concatenate` (сцепить). Эта утилита выводит на экран объединенное содержимое нескольких файлов, имена которых передаются ей в качестве аргументов командной строки. При этом файлы сцепляются в том порядке, в котором указаны в аргументах. Напишите программу на Python, имитирующую работу этой утилиты. В процессе работы программа должна выдавать сообщения о том, какие файлы открыть не удастся, и переходить к следующим файлам. Если программа была запущена без аргументов командной строки, на экран должно быть выведено соответствующее сообщение об ошибке.

### Листинг программы:

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

import sys

def cat_files(file_names):
    for file_name in file_names:
        try:
            with open(file_name, 'r', encoding='utf-8') as file:
                print(file.read())
        except FileNotFoundError:
            print(f"Не удалось открыть файл: {file_name}")

if __name__ == '__main__':
    if len(sys.argv) < 2:
        print("Ошибка: не указаны файлы для объединения.")
    else:
        files_to_cat = sys.argv[1:]
        cat_files(files_to_cat)
```

```
(Data_Analysis_1) PS C:\Users\User\PycharmProjects\Data_Analysis_1> python cat_simulator.py for_cat_1.txt for_cat_2.txt
Hello. World!
Demkar or Klepir
(Data_Analysis_1) PS C:\Users\User\PycharmProjects\Data_Analysis_1> |
```

Рисунок 15. Работа в терминале

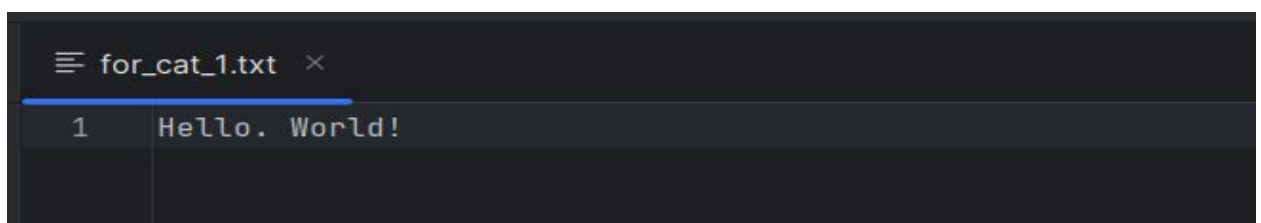


Рисунок 16. Содержимое 1-го файла

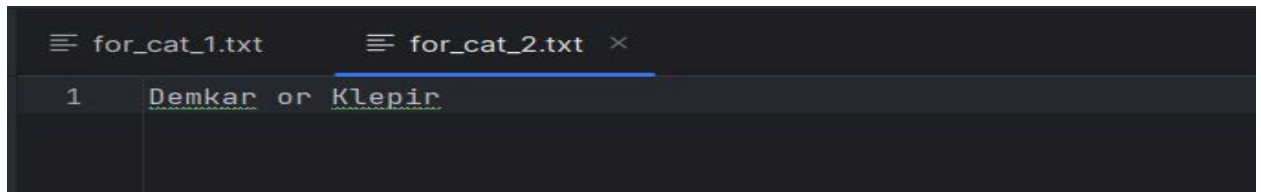


Рисунок 17. Содержимое 2-го файла

**Задача №3.** Самостоятельно подберите или придумайте задачу для работы с изученными функциями модуля `os`. Приведите решение этой задачи.

Написал программу, которая создает директорию "Oleg Kuleshov" и в этой директории создает 10 текстовых файлов с названием "текст1.txt", "текст2.txt", ..., "текст10.txt", причём текст в файлах генерируется случайно.

#### Листинг программы:

```
import os
import random
import string

# Создал директорию "Oleg Kuleshov"
directory_name = "Oleg Kuleshov"
os.makedirs(directory_name, exist_ok=True)
print(f"Создана директория: {directory_name}")

# Переход в созданную директорию
os.chdir(directory_name)

# Создание 10 текстовых файлов с различным содержимым
for i in range(1, 11):
    file_name = f"текст{i}.txt"
    with open(file_name, 'w') as file:
        random_text = ''.join(random.choices(string.ascii_letters +
string.digits, k=50)) # Сгенерируем случайный текст
        file.write(random_text)
    print(f"Создан файл: {file_name}")

print("Все файлы созданы в директории 'Oleg Kuleshov'")
```

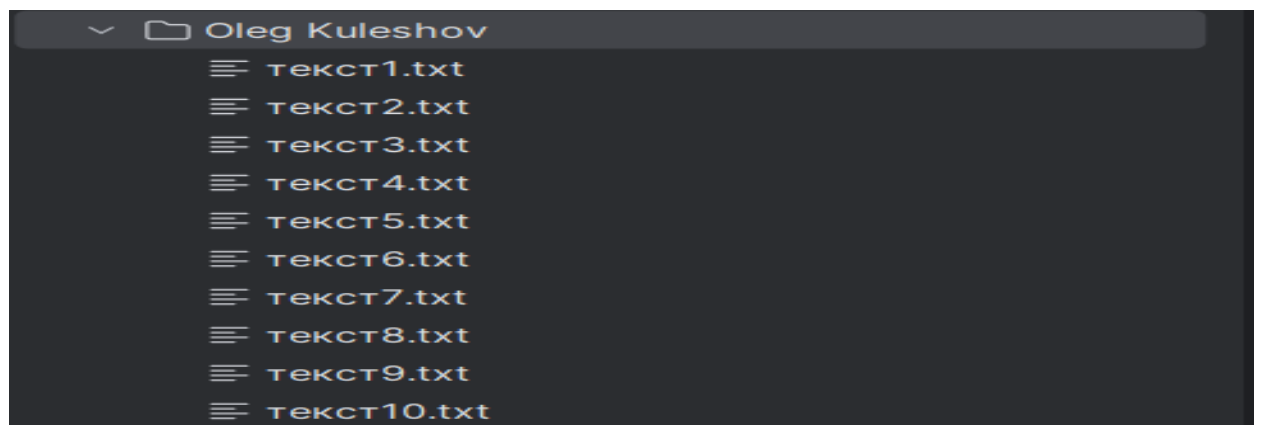


Рисунок 18. Результат (создание директории и 10 текстовых файлов в ней)



Рисунок 19. Случайное содержимое файла

**Вывод:** в ходе выполнения данной лабораторной работы были приобретены навыки взаимодействия с текстовыми файлами при написании программ с помощью языка программирования Python версии 3.x, изучены основные методы модуля `os` для работы с файловой системой, получены аргументы командной строки.

### Ответы на контрольные вопросы

1. Для открытия файла только для чтения в Python используется функция `open()` с параметром `'r'`.
2. Для открытия файла только для записи в Python используется функция `open()` с параметром `'w'`.
3. Для чтения данных из файла в Python можно использовать метод `read()`.
4. Для записи данных в файл в Python можно использовать метод `write()`.
5. Чтобы закрыть файл в Python, нужно использовать метод `close()`.
6. Конструкция `with ... as` используется в Python для автоматического управления ресурсами. Когда блок кода завершается, все ресурсы, открытые внутри этого блока, будут автоматически закрыты. Это может быть использовано не только для работы с файлами, но и с базами данных, сокетами и другими ресурсами.
7. Помимо методов `read()` и `write()` существуют такие методы как `readline()` (чтение построчно), `writelines()` (запись списка строк в файл), `seek()` (перемещение указателя файла) и другие.
8. Некоторые другие функции модуля `os` для работы с файловой системой в Python включают `os.path.exists()` (проверка существования файла), `os.path.isdir()` (проверка, является ли объект директорией) и другие.