

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ
ФЕДЕРАЦИИ**

**Федеральное государственное автономное образовательное
учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»**

**Кафедра инфокоммуникаций
«Работа с файлами в языке Python»**

**Отчет по лабораторной работе № 2.15
по дисциплине «Основы программной инженерии»**

Выполнил студент группы ПИЖ-б-о-21-1
Кучеренко С. Ю. « » 2023г.

Подпись студента _____

Работа защищена « » _____ 2023г.

Проверил Воронкин Р.А. _____

(подпись)

Ставрополь 2023

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.

Выполнение работы:

1. Изучить теоретический материал работы.
2. Создать общедоступный репозиторий на GitHub, в котором будет использована лицензия IT и язык программирования Python.

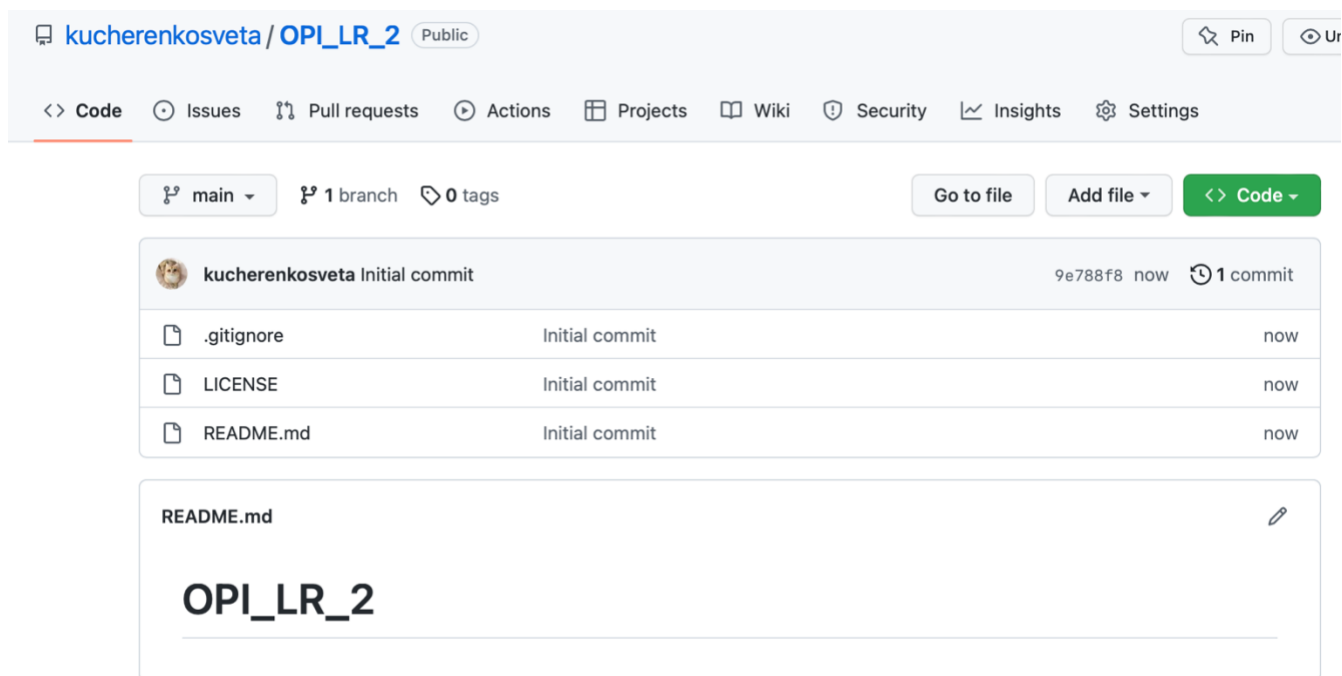


Рисунок 1 – Создание репозитория

3. Выполните клонирование созданного репозитория.

```
[(base) svetik@MacBook-Air-Svetik OPI % git clone https://github.com/kucherenkosveta/OPI_LR_2.git
Cloning into 'OPI_LR_2'...
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 5 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (5/5), done.
(base) svetik@MacBook-Air-Svetik OPI %
```

Рисунок 2 – Клонирование репозитория

4. Дополните файл .gitignore необходимыми правилами для работы с IDE PyCharm.



Рисунок 3 – Дополнение файла .gitignore

5. Организуйте свой репозиторий в соответствии с моделью ветвления git-flow.

```
[(base) svetik@MacBook-Air-Svetik OPI_LR_2 % git flow init

Which branch should be used for bringing forth production releases?
- main
Branch name for production releases: [main]
Branch name for "next release" development: [develop]

How to name your supporting branch prefixes?
Feature branches? [feature/]
Bugfix branches? [bugfix/]
Release branches? [release/]
Hotfix branches? [hotfix/]
Support branches? [support/]
Version tag prefix? []
Hooks and filters directory? [/Users/svetik/Desktop/OPI/OPI_LR_2/.git/hooks]
(base) svetik@MacBook-Air-Svetik OPI_LR_2 % █
```

Рисунок 4 – Организация репозитория в соответствии с моделью git-flow

6. Создайте проект PyCharm в папке репозитория.
7. Проработать примеры лабораторной работы.

Пример 1. Запись файла.

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

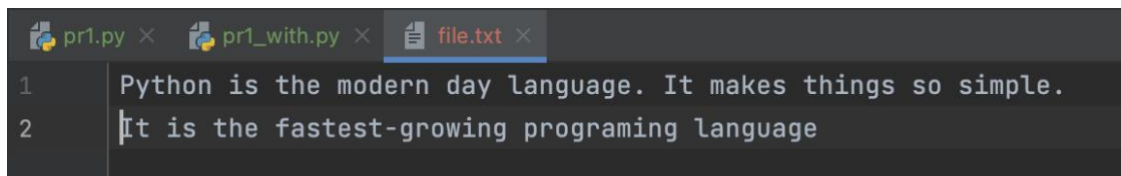
if __name__ == '__main__':
    # open the file2.txt in append mode. Create a new file if no such file exists.
    fileptr = open("file.txt", "w")

    # 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()
```

Код с with:

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

if __name__ == '__main__':
    # open the file2.txt in append mode. Create a new file if no such file exists.
    with open("file.txt", "w") as fileptr:
        # 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 programing language"
        )
```



```
pr1.py x pr1_with.py x file.txt x
1 Python is the modern day language. It makes things so simple.
2 It is the fastest-growing programing language
```

Рисунок 5 – Результат работы программы

Пример 2. Запись файла.

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

if __name__ == '__main__':
    # open the file.txt in write mode.
    fileptr = open("file.txt", "a")

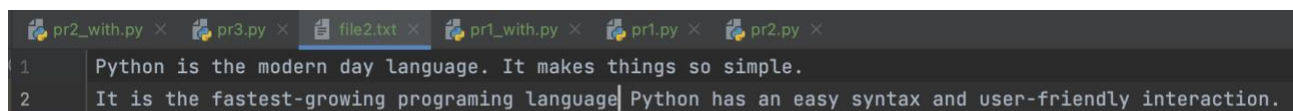
    # overwriting the content of the file
    fileptr.write(" Python has an easy syntax and user-friendly interaction.")

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

Код с with:

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

if __name__ == '__main__':
    # open the file2.txt in write mode.
    with open("file.txt", "a") as fileptr:
        # overwriting the content of the file
        fileptr.write(" Python has an easy syntax and user-friendly interaction.")
```



```
pr2_with.py x pr3.py x file2.txt x pr1_with.py x pr1.py x pr2.py x
1 Python is the modern day language. It makes things so simple.
2 It is the fastest-growing programing language Python has an easy syntax and user-friendly interaction.
```

Рисунок 6 – Результат работы программы

Пример 3. Чтение строк с помощью метода readline().

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

if __name__ == '__main__':
    # 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
    content1 = fileptr.readline()
    content2 = fileptr.readline()

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

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

Код с with:

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

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

```
/Users/svetik/Desktop/OPI/OPI_LR_2/PyCharm/venv/bin/python /Users/svetik/Desktop/OPI/OP
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-friend

Process finished with exit code 0
```

Рисунок 7 – Результат работы программы

Пример 4. Чтение строк с помощью функции readlines().

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

if __name__ == '__main__':
    # open the fil2.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()
```

Код с with:

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

if __name__ == '__main__':
    # open the file2.txt in read mode. causes error if no such file exists.
    with open("file2.txt", "r") as fileptr:
        # stores all the data of the file into the variable content
        content = fileptr.readlines()
        # prints the content of the file
        print(content)
```

```
/Users/svetik/Desktop/OPI/OPI_LR_2/PyCharm/venv/bin/python /Users/svetik/Desktop/OPI/OP
['Python is the modern day language. It makes things so simple.\n', 'It is the fastest-

Process finished with exit code 0
```

Рисунок 8 – Результат работы программы

Пример 5. Создание нового файла.

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

if __name__ == '__main__':
    # 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()
```

Код с with:

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

if __name__ == '__main__':
    # open the newfile.txt in read mode. causes error if no such file exists.
    with open("newfile.txt", "x") as fileptr:
        print(fileptr)

    if fileptr:
        print("File created successfully")
```

```
/Users/svetik/Desktop/OPI/OPI_LR_2/PyCharm/venv/bin/python /Users/svetik/Desktop/OPI/OPI_LR_2/PyCharm/venv/bin/python
<_io.TextIOWrapper name='newfile.txt' mode='x' encoding='UTF-8'>
File created successfully

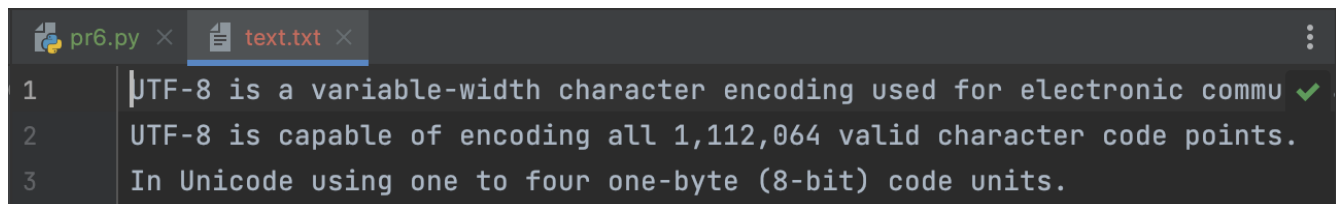
Process finished with exit code 0
```

Рисунок 9 – Результат работы программы

Пример 6. Изменение кодировки файла.

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

if __name__ == '__main__':
    # open the text.txt in append mode. Create a new file if 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)
```



```
pr6.py × text.txt ×
1 UTF-8 is a variable-width character encoding used for electronic commu ✓
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.
```

Рисунок 10 – Результат работы программы

Пример 7. Написать программу, которая считывает текст из файла и выводит на экран только предложения, содержащие запятые. Каждое предложение в файле записано на отдельной строке.

Код:

```
#!/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)
```

```
/Users/svetik/Desktop/OPI/OPI_LR_2/PyCharm/venv/bin/python /Users/svetik/Desktop/OPI/OP
UTF-8 is capable of encoding all 1,112,064 valid character code points.
```

```
Process finished with exit code 0
```

Рисунок 11 – Результат работы программы

Пример 8. Позиция указателя файла.

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

if __name__ == "__main__":
    # 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())
```

```
The filepointer is at byte : 0
After reading, the filepointer is at: 10
```

Рисунок 12 – Результат работы программы

Пример 9. Переименование файла.

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

if __name__ == "__main__":
    # rename file2.txt to file3.txt
    os.rename("file2.txt", "file3.txt")
```

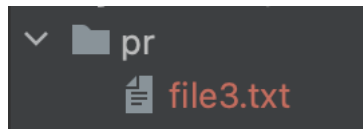


Рисунок 13 – Результат работы программы

Пример 10. Удаление файла Код:

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

if __name__ == "__main__":
    # deleting the file named file3.txt
    os.remove("file3.txt")
```

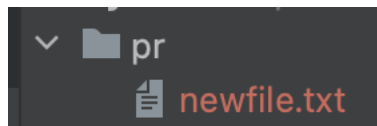


Рисунок 14 – Результат работы программы

Пример 11. Создание нового каталога.

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

if __name__ == "__main__":
    # creating a new directory with the name new
    os.mkdir("new")
```

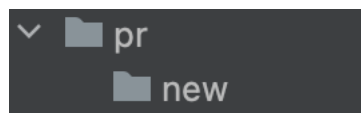


Рисунок 15 – Результат работы программы

Пример 12. Получение текущего рабочего каталога.

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

```
import os

if __name__ == "__main__":
    path = os.getcwd()
    print(path)
```

```
/Users/svetik/Desktop/OPI/OPI_LR_2/PyCharm/venv/bin/python /Users/svetik/Desktop/OPI/OP
/Users/svetik/Desktop/OPI/OPI_LR_2/PyCharm/pr
```

```
Process finished with exit code 0
```

Рисунок 16 – Результат работы программы

Пример 13. Изменение текущего рабочего каталога.

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

```
import os

if __name__ == "__main__":
    # Changing current directory with the new directory
    os.chdir("/Users/svetik/Desktop")
    # It will display the current working directory
    print(os.getcwd())
```

```
/Users/svetik/Desktop/OPI/OPI_LR_2/PyCharm/venv/bin/python /Users/svetik/Desktop/OPI/OP
/Users/svetik/Desktop
```

```
Process finished with exit code 0
```

Рисунок 17 – Результат работы программы

Пример 14. Удаление каталога.

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

import os

if __name__ == "__main__":
    # removing the new directory
    os.rmdir("new")
```

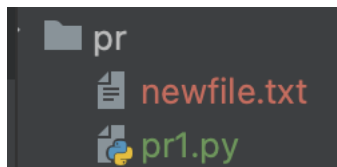


Рисунок 18 – Результат работы программы

Пример 15. Доступ к элементам командной строки в языке программирования Python.

```
#!/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))
```

```
/Users/svetik/Desktop/OPI/OPI_LR_2/PyCharm/venv/bin/python /Users/svetik/De
Number of arguments: 1 arguments
Argument List: ['/Users/svetik/Desktop/OPI/OPI_LR_2/PyCharm/pr/pr15.py']

Process finished with exit code 0
```

Рисунок 19 – Результат работы программы

Пример 16. Изменение кодировки файла.

```
#!/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))
```

```
/Users/svetik/Desktop/OPI/OPI_LR_2/PyCharm/venv/bin/python /Users/svetik/
Argument #0 is /Users/svetik/Desktop/OPI/OPI_LR_2/PyCharm/pr/pr16.py
No. of arguments passed is 1

Process finished with exit code 0
```

Рисунок 20 – Результат работы программы

Пример 17. Изменение кодировки файла.

```
#!/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)
        sys.exit(1)

    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)}")
```

8. Выполнить индивидуальные задания.

Задание 1. Составить программу с использованием списков и словарей для решения задачи. Номер варианта определяется по согласованию с преподавателем. Исходный файл, из которого выполняется чтение, необходимо также добавить в репозиторий, каждое предложение в файле должно находиться на отдельной строке.

Вариант 9. Написать программу, которая считывает английский текст из файла и выводит на экран слова текста, начинающиеся и оканчивающиеся на гласные буквы.

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

def s(text):
    lts = 'ueioay'
    for word in text:
        words = []
        for prword in lts:
            if word[0] == prword:
                words = str(lts)
        for prword in words:
            if word.endswith(prword):
                print(word)

if __name__ == "__main__":
```

```
with open("text.txt", "r", encoding="utf-8") as fileptr:
    sentences = fileptr.readlines()
text = str(sentences).lower().split()
s(text)
```

```
also
are
a
are
are
one
are
a
one
any
above
improve
accuracy
once
are
are
a
image
one

Process finished with exit code 0
```

Рисунок 21 – Результат работы программы

Задание 2. Составить программу с использованием текстовых файлов.

9. Создание пароля посредством генерирования случайных символов может обернуться сложностью в запоминании полученной относительно надежной последовательности. Некоторые системы создания паролей рекомендуют сцеплять вместе два слова на английском языке, тем самым упрощая запоминание заветного ряда символов – правда, в ущерб его надежности. Напишите программу, которая будет открывать файл со списком слов, случайным образом выбирать два из них и сцеплять вместе для получения итогового пароля. При создании пароля исходите из следующего требования: он должен состоять минимум из восьми символов и максимум из десяти, а каждое из используемых слов должно быть длиной хотя бы в три буквы. Кроме того, сделайте заглавными первые буквы обоих слов, чтобы легко можно было понять, где заканчивается одно и начинается другое. По завершении процесса полученный пароль должен быть отображен на экране.

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

""" Напишите программу, которая будет открывать файл со списком
слов, случайным образом выбирать два из них и сцеплять вместе для получения
итогового
пароля. При создании пароля исходите из следующего требования: он должен
состоять
минимум из восьми символов и максимум из десяти, а каждое из используемых слов
```

```

должно быть длиной хотя бы в три буквы. Кроме того, сделайте заглавными первые
буквы
обоих слов, чтобы легко можно было понять, где заканчивается одно и начинается
другое.
По завершении процесса полученный пароль должен быть отображен на экране."""

import random

if __name__ == "__main__":
    with open('words.txt') as file:
        lines = file.readlines()
        while True:
            word1 = random.choice(lines).title()[:-2]
            word2 = random.choice(lines).title()[:-2]
            if 8 <= len(word1) + len(word2) <= 10 and len(word1) >= 3 and
len(word2) >= 3:
                password = word1 + word2
                break
        print(password)

```

```

/Users/svetik/Desktop/OPI/OPI_LR_2/PyChar
WritinWais

Process finished with exit code 0

```

Рисунок 22 – Результат работы программы

9. Зафиксируйте изменения в репозитории.

10. Самостоятельно подберите или придумайте задачу для работы с изученными функциями модуля os. Приведите решение этой задачи.

```

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

"""
Скачали календарь на февраль, март и апрель, но файлы были названы
February.png, April.png и April(1).png.
Нужно переименовать April в March, а April(1) в April, чтобы не путаться в
месяцах.
Так как февраль уже прошел, решили что хранение изображения календаря на
февраль - не нужно.
Следует удалить February.png.
В процессе работы запутались в каталогах, нужно определить текущий.
"""

import os

if __name__ == "__main__":
    os.rename("April.png", "March.png")
    os.rename("April(1).png", "April.png")
    os.remove("February.png")
    path = os.getcwd()
    print(path)

```

10. Зафиксируйте сделанные изменения в репозитории.

Вопросы для защиты работы:

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

С помощью команды: `fileobj = open("file.txt", "r")` или с помощью `with open("file2.txt", "w") as fileptr:`

2. Как открыть файл в языке Python только для записи

Для этого нужно указать код доступа «r» или «rb» (открывает файл в двоичном формате)

3. Как прочитать данные из файла в языке Python?

После открытия можно воспользоваться командами `f.read()` или `f.readlines()` (возвращает массив строк), `f.readline()` (возвращает 1 строку)

4. Как записать данные в файл в языке Python?

Во-первых, файл должен быть открыт с соответствующим режимом доступа, например, «w» (переписывает файл) или «a» (добавляет в конец).

Во-вторых, воспользоваться специальной командой, такой как `f.write("///")`.

5. Как закрыть файл в языке Python?

Необходимо воспользоваться методом `close()`, однако, если файл был открыт с «with», то закрытие выполнится автоматически.

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 определённое количество байтов.

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

Вернуться в предыдущую директорию: `os.chdir("..")`

Сделать несколько вложенных папок:

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

Заменить (переместить) этот файл в другой каталог: `os.replace("renamed-text.txt", "folder/renamed-text.txt")`

Распечатать все файлы и папки в текущем каталоге:

```
print("Все папки и файлы:", os.listdir())
```

Генератор дерева каталогов: `os.walk()`

Распечатать все файлы и папки рекурсивно: `for dirpath, dirnames, filenames in os.walk(".):`

Получение информации о файле: `os.stat()`

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.