

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ
АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО
ОБРАЗОВАНИЯ «СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»
ИНСТИТУТ ЦИФРОВОГО РАЗВИТИЯ**

**Отчёт о лабораторной работе №3 по дисциплине основы программной
инженерии**

Выполнила:

Нестеренко Тамара Антоновна,
2 курс, группа ПИЖ-б-о-20-1,

Проверил:

Доцент кафедры инфокоммуникаций,
Воронкин Р.А.

Ставрополь, 2022 г.

ВЫПОЛНЕНИЕ

1. Практическая часть

```
fileptr = open("file2.txt", "w")

fileptr.write(
    "Python is the modern day language. It makes things so simple.\n"
    "It is the fastest-growing programing language"
)

fileptr.close()
```

Рисунок 1 – Пример записи файла

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

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"
    )
```

Рисунок 2 – Пример записи файла с помощью with

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

fileptr = open("file2.txt", "a")

fileptr.write("Python has an easy syntax and user-friendly interaction.")

fileptr.close()
```

Рисунок 3 – Пример открытия и записи файла

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

with open("file2.txt", "a") as fileptr:
    fileptr.write("Python has an easy syntax and user-friendly interaction.")
```

Рисунок 4 – Пример открытия и записи файла с помощью with

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

fileprt = open("file2.txt", "r")

content1 = fileptr.readline()
content2 = fileptr.readline()

print(content1)
print(content2)

fileprt.close()
```

Рисунок 5 – Пример чтения строк с помощью метода readline()

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

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

    content1 = fileptr.readline()
    content2 = fileptr.readline()

    print(content1)
    print(content2)
```

Рисунок 6 – Пример чтения строк с помощью метода readline() с with

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

fileptr = open("file2.txt", "r")

content = fileptr.readline()

print(content)

fileptr.close()
```

Рисунок 7 – Пример чтения строк с помощью функции `readlines()`

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

with open("file2.txt", "r") as fileptr:
    content = fileptr.readlines()
    print(content)
```

Рисунок 8 – Пример чтения строк с помощью функции `readlines()` с `with`

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

fileptr = open("newfile.txt", "x")
print(fileptr)

if fileptr:
    print("File created successfully")

fileptr.close()
```

Рисунок 9 – Пример создания файла

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

with open("newfile.txt", "x") as fileptr:
    print(fileptr)

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

Рисунок 10 – Пример создания файла с with

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

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

        for sentence in sentences:
            if "," in sentence:
                print(sentence)
```

Рисунок 11 – Пример программы, которая считывает текст из файла и выводит на экран только предложения, содержащие запятые. Каждое предложение в файле записано на отдельной строке.

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

if __name__ == "__main__":
    with open("test.txt", "w", encoding="utf-8") as fileptr:
        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, 864 valid character code points.",
            file=fileptr
        )
        print(
            "In unicode using one to four one-byte (8-bit) code units.",
            file=fileptr
        )
```

Рисунок 12 – Пример изменения кодировки

```

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

with open("file2.txt", "r") as fileptr:
    print("The filepointer is at byte: ", fileptr.tell())

    content = fileptr.read()

    print("After reading, the filepointer is af:", fileptr.tell())

```

Рисунок 13 – Пример использования метода tell()

```

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

import os
...
os.rename("file2.txt", "file3.txt")

```

Рисунок 14 – Пример переименования файла

```

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

import os
...
os.remove("file3.txt")

```

Рисунок 15 – Пример удаления файла

```

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

import os
...
os.mkdir("new")

```

Рисунок 16 – Пример создания нового каталога

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

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

Рисунок 17 – Пример получения текущего каталога

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

import os
...
os.chdir("C:\\Windows")
print(os.getcwd())
```

Рисунок 18 – Пример изменения текущего рабочего каталога

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

import os
...
os.rmdir("new")
```

Рисунок 19 – Пример удаления каталога

```
# !/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))
```

Рисунок 20 – Пример подсчёта количества аргументов

```

#!/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))

```

Рисунок 21 – Пример подсчёта количества аргументов

```

#!/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)}")

```

Рисунок 22 – Пример программы для генерации пароля заданной длины. Длина пароля должна передаваться как аргумент командной строки сценария


```
main.py
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     counter = 0
9     key = input("Введите ключ: ")
10    for sentence in sentences:
11        words = sentence.split()
12        for word in words:
13            if key == word:
14                counter += 1
15    print(sentence, counter)
16
17 if __name__ == "__main__":
18     No sentence in sentences
19
20 main
21 Введите ключ:
22 History of the Internet:
23 1
24 From its very beginnings the Internet became a crucial part of each and any infrastructure.
25 2
26 Similar to the discoveries of electricity, microorganisms or elementary particles, the creation of the Internet has turned a new page in the history of humanity.
27 6
28 The history of the Internet has begun in the middle of the 19th century as a result of rapid development of computer science.
29 9
30 Computers of that age were still relatively underperforming and needed constant maintenance.
31 9
32 Some kind of an effective and automated method of time-sharing between users needed to be devised and implemented for them to work reliably.
```

Рисунок 23 – Пример решения индивидуального задания №1 (Вариант 16)

```
15 # !/usr/bin/env python3
16 #-*- coding: utf-8 -*-
17
18
19 if __name__ == "__main__":
20     with open("text.txt", "r", encoding="utf-8") as f:
21         text = f.read().lower()
22         text = text.split()
23
24     with open("service.txt", "r", encoding="utf-8") as t:
25         service = t.read().lower()
26         service = service.split()
27     k = open("new.txt", "w", encoding="utf-8")
28
29     for word_ser in service:
30         word_ser = word_ser.split()
31         for word_text in text:
32             word_text = word_text.split()
33             if word_text == word_ser:
34                 String = ''.join(word_text).lower()
35                 long = len(String)
36                 word_text = String.split()
37                 while long != 0:
```

```

        while long != 0:
            str2 = ''
            for element in String:
                element = '*'
                str2 += element
                long -= 1
            k.write(' ')
            k.write(str2)
            k.write(' ')
        else:
            word_text = ''.join(word_text)
            k.write(word_text)

k.close()

```

Рисунок 24 – Пример решения индивидуального задания №2 (Вариант 16 (14))

```

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)
12
13
14
15
16
17
18
19
20
21
22

```

main x

C:\Users\тома нестеренко\PycharmProjects\pythonProject1

Process finished with exit code 0

Рисунок 25 – Пример решения индивидуального задания №3

2. Вопросы для защиты

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
```

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.

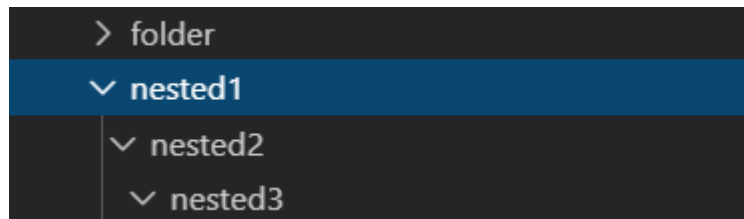
7. Какие существуют, помимо рассмотренных, функции модуля `os` для работы с файловой системой?

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(os.path.join(dirpath, filename))
```

```
            # 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

Ссылка на репозиторий: https://github.com/tamaranesterenko/Python_LR_3-2