

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

**Отчет по лабораторной работе № 3
«Работа с файлами в языке Python»**

по дисциплине «Основы программной инженерии»

Выполнила:
Первых Дарья Александровна,
2 курс, группа ПИЖ-б-о-20-1,
Проверил:
Доцент кафедры инфокоммуникаций,
Воронкин Р.А.

Ставрополь, 2022 г.

ВЫПОЛНЕНИЕ

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

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

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

fileptr.close()
```

Рисунок 1 – Пример записи файла

```
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4
5  with open("file2.txt", "w") as fileptr:
6
7      fileptr.write(
8          "Python is the modern day language. It makes things so simple.\n"
9          "It is the fastest-growing programming language"
10     )
11
```

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 -*-

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

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

print(content1)
print(content2)

fileptr.close()

```

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

```

1  # !/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4
5  with open("file2.txt", "r") as fileptr:
6
7      content1 = fileptr.readline()
8      content2 = fileptr.readline()
9
10     print(content1)
11     print(content2)

```

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

main ×

C:\Users\podar\study\anaconda\envs\LR3\python.exe "C:/Users/podar/study
Python is the modern day language. It makes things so simple.

It is the fastest-growing programing languagePython has an easy syntax

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()`

```

1  # !/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4
5  with open("file2.txt", "r") as fileptr:
6      content = fileptr.readlines()
7      print(content)
8

```

main x

C:\Users\podar\study\anaconda\envs\LR3\python.exe "C:/Us

['Python is the modern day language. It makes things so

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

```
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4
5 fileptr = open("newfile.txt", "x")
6 print(fileptr)
7
8 if fileptr:
9     print("File created successfully")
10
11 fileptr.close()
12
```

main x

C:\Users\podar\study\anaconda\envs\LR3\python.exe "C:/Users/podar/stu
<_io.TextIOWrapper name='newfile.txt' mode='x' encoding='cp1251'>
File created successfully

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

```

1  #!/usr/bin/env python3
2  #- coding: utf-8 -*-
3
4
5  if __name__ == "__main__":
6      with open("test.txt", "w", encoding="utf-8") as fileptr:
7          print(
8              "UTF-8 is a variable-width character encoding used for electronic c
9              file=fileptr
10         )
11     print(
12         "UTF-8 is capable of encoding all 1, 112, 064 valid character code
13         file=fileptr
14     )
15     print(
16         "In unicode using one ti four one-byte (8-bit) code units.",
17         file=fileptr
18     )

```

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", "r", encoding="utf-8") as fileptr:
        sentences = f.readlines()

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

```

Рисунок 12 – Пример изменения кодировки

```
1 #!/usr/bin/env python3
2 #- coding: utf-8 -*-
3
4
5 with open("file2.txt", "r") as fileptr:
6     print("The filepointer is at byte: ", fileptr.tell())
7
8     content = fileptr.read()
9
10    print("After reading, the filepointer is af:", fileptr.tell())
11
```

main x

C:\Users\podar\study\anaconda\envs\LR3\python.exe "C:/Users/podar/study
The filepointer is at byte: 0
After reading, the filepointer is af: 220
Process finished with exit code 0

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

```
1 #!/usr/bin/env python3
2 #- coding: utf-8 -*-
3
4
5 import os
6 ...
7 os.remove("file3.txt")
```

Рисунок 15 – Пример удаления файла

```

1 #!/usr/bin/env python3
2 #- coding: utf-8 -*-
3
4
5 import os
6 ...
7 os.mkdir("new")

```

Рисунок 16 – Пример создания нового каталога

```

1 #!/usr/bin/env python3
2 #- coding: utf-8 -*-
3
4
5 import os
6 ...
7 path = os.getcwd()
8 print(path)

```

main x

C:\Users\podar\study\anaconda\envs\LR3\python.exe "C:/Users/podar/
C:\Users\podar\study\PyCharm Community Edition 2021.2.3\LR3

Рисунок 17 – Пример получения текущего каталога

```

1 #!/usr/bin/env python3
2 #- coding: utf-8 -*-
3
4
5 import os
6 ...
7 os.chdir("C:\\Windows")
8 print(os.getcwd())

```

main x

C:\Users\podar\study\anaconda\envs\LR3\python.e
C:\Windows

Process finished with exit code 0

Рисунок 18 – Пример изменения текущего рабочего каталога


```

1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4
5  import os
6  ...
7  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

```

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

```

Рисунок 21 – Пример подсчёта количества аргументов

```

6 import secrets
7 import string
8 import sys
9
10
11 if __name__ == "__main__":
12     if len(sys.argv) != 2:
13         print("The password length is not given!", file=sys.stderr)
14         sys.exit(1)
15
16     chars = string.ascii_letters + string.punctuation + string.digits
17     length_pwd = int(sys.argv[1])
18
19     result = []
20     for _ in range(length_pwd):
21         idx = secrets.SystemRandom().randrange(len(chars))
22         result.append(chars[idx])
23
24     print(f"Secret Password: {''.join(result)}")

```

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

Индивидуальное задание.

Написать программу, которая считывает текст из файла и выводит на экран все его предложения в обратном порядке.

```
1  #!/usr/bin/env python3
2  #- coding: utf-8 -*-
3
4
5  if __name__ == '__main__':
6      with open('text.txt', 'r') as f:
7          text = f.read()
8          print(text)
9          text = text.split(" ")
10         text.reverse()
11         print(text)
```

if __name__ == '__main__'

main x

I easily and successfully passed it the first time
May luck be near
Behind my shoulder to stand
So that the control point on the opi
Turned out to be "Excellent" for me to pass
On this day, luck will only be on my side
['side', 'my', 'on', 'be', 'only', 'will', 'luck', 'day,', 'this']

Рисунок 23 – Решение идз

Продолжая тему предыдущего упражнения, в тех же операционных системах на базе Unix обычно есть и утилита с названием `tail`, которая отображает последние десять строк содержимого файла, имя которого передается в качестве аргумента командной строки. Реализуйте программу, которая будет делать то же самое. В случае отсутствия файла, указанного пользователем, или аргумента командной строки вам нужно вывести соответствующее сообщение.

```
12         for line in (f.readlines()[-10:]):
13             print(line, end="")
14     else:
15         print("Файл не получен для обработки!", file=sys.stderr)
16
17
18 ► if __name__ == "__main__":
19     filename = input("Введите имя файла для обработки: ")
20     if os.path.isfile(filename):
21         linux_tail(filename)
22     else:
23         print(f"Файла {filename} не существует", file=sys.stderr)
24
```

indiv x

C:\Users\podar\study\anaconda\envs\LR3\python.exe "C:/Users/podar/study/PyCharm C

Введите имя файла для обработки: **prim.txt**

I easily and successfully passed 3 lr the first time

May luck be near

Behind my shoulder to stand

So that the control point on the opi

Turned out to be "Excellent" for me to pass

On this day, luck will only be on my side

Рисунок 24 – Решение идз

Скачали календарь на февраль, март и апрель, но из-за недосыпа, 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.




 April(1).png	09.03.2022 23:42	Файл "PNG"	54 КБ
 April.png	09.03.2022 23:42	Файл "PNG"	54 КБ
 February.png	09.03.2022 23:47	Файл "PNG"	53 КБ

Рисунок 25 – Скаченные файлы

```
1  #!/usr/bin/env python3
2  #- coding: utf-8 -*-
3
4  import os
5
6
7  if __name__ == "__main__":
8      os.rename("April.png", "March.png")
9      os.rename("April(1).png", "April.png")
10     os.remove("February.png")
11     path = os.getcwd()
12     print(path)
```

main x

C:\Users\podar\study\anaconda\envs\LR3\python.exe "C:/Users/
C:\Users\podar\study\PyCharm Community Edition 2021.2.3\LR3

Рисунок 26 – Код программы




 April.png	09.03.2022 23:42	Файл "PNG"	54 КБ
 March.png	09.03.2022 23:42	Файл "PNG"	54 КБ
 zd3.py	09.03.2022 23:53	Python File	1 КБ

Рисунок 27 – Результат выполнения программы

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
```

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.

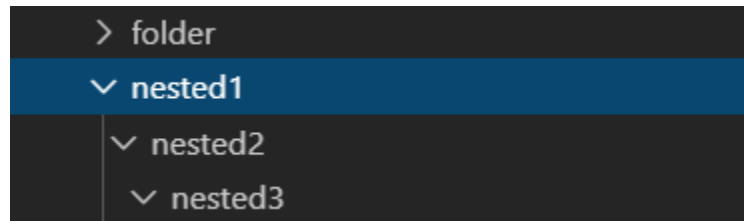
7. Какие существуют, помимо рассмотренных, функции модуля `os` для работы с файловой системой?

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("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