Южный федеральный университет
Институт математики, механики и компьютерных наук им. И. И. Воровича
направление подготовки «Прикладная математика и информатика»

Лабораторная работа по теме «Работа с файлами»[1]

Взаимодействие с файлами в `Python` делает возможным сохранить информацию, которая была обработана во время работы программы, чтобы позднее получить к ней доступ из вне. Базовые функции позволяют создавать, записывать и читать данные из файлов.
Чтобы начать работу с файлом, его необходимо открыть.

```
f = open("text.txt", "r")
```

У этой функции несколько параметров. Основные три это: путь файла, режим открытия и кодировка. Путь файла может быть абсолютным или относительным.

Как только файл был открыт, был создан файловый объект. Об этом файловом объекте можно просмотреть слующие сведения, называемые *атрибутами файла*:

f.closed	проверяет закрыт ли файл; возвращает **True** или **False**
f.mode	возвращает режим доступа, в котором был открыт файл
f.name	возвращает имя файла

ВАЖНО! После выполнения всех действий над файлом, его обязательно нужно закрыть с помощью метода close()!

```
f.close()
```

Основные методы работы с файлами:

open(name, regime)	открытие файла
close()	закрытие файла
read()	чтение всего файла целиком
read([N])	чтение первых N-символов (N - байт) файла
readline()	чтение из файла одной строки
readlines()	чтение из файла всех строк в список строк
readable()	проверяет доступен ли файл для чтения; возвращает **True** или **False**
write(line) writable()	запись в файл указанной строки проверяет доступен ли файл для записи; возвращает **True** или **False**
seek(p)	перемещение указателя на p байт относительно начала файла
seekable()	проверяет можно ли сдвинуть указатель в другую позицию; возвращает **True** или **False**
tell()	позиция указателя относительно начала файла в виде целого числа
truncate(N)	отсекает N символов с текущей позиции; если N больше, чем длина файла, оставшиеся символы будут заполнены нулевым байтом (символ `"\x00"`)
writelines(lines)	запись в файл указанную последовательность строк; разделители строк не добавляются
close()	закрытие файла

[1]Разработано А.М. Филимоновой (кафедра ВМиМФ мехмата ЮФУ)

Режимы открытия файла:

Режим	Обозначение
"r"	открытие на чтение (значение по умолчанию)
"w"	открытие на запись; содержимое файла удаляется; если файл не существует, то создастся новый
"x"	открытие на запись; если файл существует – вызывается исключение
"a"	открытие на дозапись; информация запишется в конец файла
"b"	открытие в двоичном режиме
"t"	открытие в текстовом режиме (значение по умолчанию)
"+"	открытие на чтение и запись

Режимы могут быть объединены: `"rb"` – чтение в двоичном режиме. Режим открытия файла по умолчанию – `"rt"`. При использовании `"r+"` и `"w+"` файл будет открыт и на чтение и на запись. Отличие `"r+"` и `"w+"` заключается в том, что во втором случае создастся новый файл, если такого нет. В первом же случае возникнет исключение.

Пример 1. Основные методы работы с файлами.

```
filename = "test.txt"
f = open(filename, "w")

check = f.writable()
print("Файл доступен для записи? -- %5s" % check)
if check:
    N = int(input("Введите количество строк в файле: "))
    for i in range(N):
        f.write("This is line № %d\n" %i)
else:
    print("Файл не доступен для записи! ")
f.close()

f = open(filename, "r")
P = int(input("На сколько сдвинуть указатель?"))
check = f.seekable()
if check:
    f.seek(P)
    print("Указатель на позиции %d" %f.tell())
    print("Указатель на символе "%s"" %f.read(1))

f.seek(0)
text = f.readlines()
for line in text:
    print(line, end = "")
print()
f.close()

f = open(filename,"a+")
TR = int(input("Введите сколько символов оставить в файле: "))
f.truncate(TR)

f.seek(0)
text = f.readlines()
print(text)
```

```
f.close()
print()

f = open(filename, "a")
list_names = [str(i) for i in range(10)]
list_names.append("\n")
f.write("\n")
f.writelines(list_names)
f.close()

f = open(filename)
text = f.readlines()
for line in text:
    print(line, end = "")
f.close()
```

Исходный код программы из **Примера 1** с подробными комментариями (конспект семинарского занятия) находится в файле `lab_file_ex1.py`.

Пример 2. Посчитать длину файла (количество символов).

```
from pathlib import Path
file = Path() / "test.txt"  # или Path("/test.txt")
size = file.stat().st_size
print(size)
```

Задачи для самостоятельного решения.

---

1. Дан файл, содержащий несколько строк текста. Подсчитать сколько в нём строк, слов и непробельных символов. Указание: Функции `len()` и `split()` не использовать.

---

2. Создать файл случайных целых чисел. Поменять в нем все элементы на их квадраты.
3. Создать файл случайных вещественных чисел. Поменять в нем местами максимальный и минимальный элементы.

---

4. Создать файл случайных вещественных чисел. Заменить в нем каждый элемент с чётным **номером** на три нуля.

---

5. Создать файл случайных целых чисел, количество которых также случайно, но строго меньше 50. Увеличить его размер до 50 элементов, записав в начало файла необходимое количество нулей.
6. Дан файл, содержащий несколько строк текста и слово, введённое с клавиатуры. Проверить, есть ли это слово в файле. Если есть – вывести позицию его начала.

---

7. Дан файл целых чисел, разделённых пробелами, состоящий из нескольких строк. Найти в каждой строке максимальный по модулю элемент. В новый файл записать максимальные значения каждой из строк. Вывести на экран содержимое нового файла.
8. Дан файл целых чисел, разделённых пробелами. Создать два новых файла. В один из файлов поместить все чётные числа из исходного файла, в другой – все нечётные. Вывести на экран содержимое обоих файлов.

---

9. Дан текстовый файл. Определить строку, являющуюся его серединой и вывести на экран. Оставить только вторую половину файла. В начале каждой строки дописать её номер. Вывести на экран содержимое изменённого файла.

10. Дан текстовый файл и число N. Определить и вывести на экран строку с номером N. Оставить в файле только строки от начала до N включительно. В начале каждой строки дописать её номер. Вывести на экран содержимое изменённого файла.

---

11. Дано два файла, состоящие из целых чисел разделённых пробелами (количество строк в файлах может быть любым). Проверить, что набор чисел в обоих файлах одинаковый.

12*. Дан текстовый файл, содержащий несколько алгебраических выражений. Проверить, верно ли в каждом из них расставлены скобки. Сформировать новый файл с таким же количеством строк, что и исходный файл, в каждой строке которого записано "верно"или "неверно". Будем считать, что скобки расставлены *верно*, если выражение имеет смысл.

Пример верных выражений: $a + (b - c) * a$, $a * (2 - b)$.

Пример неверных выражений: $a+)b * c$, $(a + b * c + a$, $b-)(a + 4$.