Методические указания

Урок 10.1. Работа с файлами

Задачи урока:

- Научиться работать с локальными файлами

0. Подготовка к уроку

До начала урока преподавателю необходимо:

1) Просмотреть, как ученики справились с домашним заданием
2) Прочитать методичку

1. Сохранение данных в файл

Учитель: Прошлые занятия, мы с вами познакомились с основными конструкциями языка. Теперь мы можем уже написать какую либо простую программу, но хотелось бы, чтобы мы имели возможность сохранять и загружать данные извне.
Существуют несколько вариантов

1. Сохранение данных в базу данных
2. Сохранение данных в документ(txt, xml и т.д)

Мы с вами рассмотрим 2 вариант.
Для работы с файлами(чтение, запись, перезапись) существует команда open, открывающая файл. Давайте создадим текстовый документ рядом с файлом нашей программы и назовем его example.txt. Запишите в файл пару любых строк и попробуем считать их

```
file = open('example.txt')
print(file.read())
```

Результатом выполнения программы будет вывод каждой нашей строки на новой строке. Функция read(), указывает, что мы хотим прочесть данный файл. По умолчанию документ открывается только для чтения. После любой операции с файлом его желательно закрыть

```
file = open('example.txt')
```

```
print(file.read())
file.close()
```

Для изменения режима открытия файла указываются определенные параметры

Режим	**Описание**
r	Только для чтения.
w	Только для записи. Создаст новый файл, если не найдет с указанным именем.
r+	Для чтения и записи.
w+	Для чтения и записи. Создаст новый файл для записи, если не найдет с указанным именем.
a	Откроет для добавления нового содержимого. Создаст новый файл для записи, если не найдет с указанным именем.
a+	Откроет для добавления нового содержимого. Создаст новый файл для чтения записи, если не найдет с указанным именем.

Параметр ‘r’ стоит по умолчанию. Т.е данный код равен коду

```
file = open('example.txt', ‘r’)
print(file.read())
file.close()
```

Допустим мы хотим добавить в файл новую строчку. Согласно таблице, в данном случае нам подойдет параметр ‘a’, так как ‘w’ полностью перезапишет файл

```
file = open('example.txt', 'a')
file.write('\npython')
file.close()
file = open('example.txt', 'r')
print(file.read())
file.close()
```

В коде выше, используется символ переноса строки \n для записи python на новую строку. Если же вдруг файла с таким именем не найдено, то программа автоматически создаст файл с таким именем и запишет в него строку.
В случае когда мы хотим полностью перезаписать файл мы можем использовать параметр ‘w’

```
file = open('example.txt', 'w')
file.write('\npython')
file.close()
file = open('example.txt', 'r')
print(file.read())
file.close()
```

В данном случае после выполнения кода все строки в текстовом файле будут стерты, а после добавлена новая строка.

При использовании функции open() не стоит забывать закрывать файл с помощью close(), для того, чтобы все ресурсы и мусор были удалены из памяти.

Учитель: Еще один подход — использовать инструкцию with, которая упрощает обработку ошибок, а также задач по закрытию и очистке.
В таком случае инструкция close не нужна, потому что with автоматически закроет файл.

Вот как это реализовать в коде.

```
with open('example.txt') as file:
    print(file.read())
```

Как мы видим в данном случае мы не использовали close(). В целом остальная работа с файлом проходит также.

Давайте теперь познакомимся ближе с функциями чтения и записи данных в файл

Функция read() используется для чтения содержимого файла после открытия его в режиме чтения (r).

```
file.read(size)
```

- file = объект файла
- size = количество символов, которые нужно прочитать. Если не указать, то файл прочитается целиком.

```
f = open('example.txt','r')
f.read(7)  # чтение 7 символов из example.txt
```

```
f = open('example.txt','r')
f.read(7)  # чтение 7 символов из example.txt
f.read(7)  # чтение следующих 7 символов
```

Функция readlines() используется для построчного чтения содержимого файла. Она используется для крупных файлов. С ее помощью можно получать доступ к любой строке в любой момент.

```
x = open('example.txt','r')
x.readline()  # прочитать первую строку
x.readlines(2)  # прочитать вторую строку
x.readlines()  # прочитать все строки
```

Функция write() используется для записи в файлы Python, открытые в режиме записи.
Если пытаться открыть файл, которого не существует, в этом режиме, тогда будет создан новый.
Предположим, файла example2.txt не существует. Он будет создан при попытке открыть его в режиме записи.

```
f = open('example2.txt','w')  # открытие в режиме записи
f.write('Hello \n World')  # запись Hello World в файл
f.close()  # закрытие файла
```

Методы файла:

file.close()	закрывает открытый файл
file.flush()	очищает внутренний буфер

file.__next__()	возвращает следующую строку файла
file.read(n)	чтение первых n символов файла
file.readline()	читает одну строчку строки или файла
file.readlines()	читает и возвращает список всех строк в файле
file.tell()	возвращает текущую позицию в файле
file.truncate(n)	уменьшает размер файл. Если n указала, то файл обрезается до n байт, если нет — до текущей позиции
file.write(str)	добавляет строку str в файл
file.writelines(sequence)	добавляет последовательность строк в файл

Рассмотрим пример с чтением файла целиком в одну строку, целиком в список и построчно

```
filename = "test.txt"

# 1. Чтение из файла (в одну строку)
with open(filename, encoding="utf-8") as file:
  data = file.read()
  print(data)

# 2. Чтение из файла (в список)
with open(filename, encoding="utf-8") as file
  data = file.readlines()
  print(data)

# 3. Чтение из файла (построчно)
with open(filename, encoding="utf-8") as file:
  for line in file:
     print(line.strip())
```

Также существует огромное количество модулей, которые как поставляются вместе с python, так и устанавливаются отдельно и позволяют расширить возможности языка.

Например встроенный в язык модуль os позволяет работать с файлами и директориями в операционной системе.

Функция rename() используется для переименования файлов в Python. Для ее использования сперва нужно импортировать модуль os.

```
import os
os.rename(src,dest)
```

- src = файл, который нужно переименовать
- dest = новое имя файла

```
import os
os.rename("example2.txt","example3.txt")
```

2. Решение задач

Задача 1

Написать функцию, которая принимает у пользователя имя файла example3.txt. Заранее стоит создать данный файл и записать туда несколько строк. Функция должна вернуть первую строку записанную в данный файл, а после переименовать его в example4.txt

Решение

```
import os

file_name = input()
def read_file(file_name):
    with open(file_name, 'r') as lines:
        line = lines.readline()
    os.rename(file_name, 'example4.txt')
    return line

print(read_file(file_name))
```

Дополнительно

Если на уроке остается время, то ученикам можно предложить начать прорешивать домашнее задание.

Домашняя работа

Задача 1