COMP 125 Programming with Python
Files and Error Handling

Mehmet Sayar

Koç University

Working with Files in Python

- Python has a lot of built-in functions to work with files
- In order to work with a file we must first "**open**" it

```
file = open(file_name, access_mode)
```

- Then we can process it:
  - Read from a file
  - Write to a file

File Access Modes

Access Mode	Meaning
'r'	open for reading (default)
'w'	open for writing, create it if it doesn’t exist, truncate the file if it does (overwrites the file)
'x'	open for exclusive creation, fail if the file already exists
'a'	open for writing, appending to the end of the file if it exists
'b'	open the file in binary format, used with above access types, e.g. 'rb', (do not worry about it for now)
'r+'	open for both reading and writing, fail if the file does not exist
'w+', 'a+'	open for both reading and writing/appending
'x+'	open for both reading and writing, fail if the file already exists
'b+'	open in binary format, used with access types and have the same meaning

You do not need to memorize this table!

File Processing

- Virtually all programs that you've used **wrote and read files** from disk at some point:
  - Word processing (documents)
  - Web browser (cookies)
  - Games (saved progress)
  - Spyder (Python files)
  - Music player (songs)

- Similar to a variable, a file is a series of bits (ones and zeros).
  - in plain text, bits encode characters
  - in JPEGs, bits encode information about the color of the image pixelse
  - in MP3 files, bits encode frequency information

File Reading

- There are multiple ways to read a file in Python. Common ones:
  - Read the entire file (copy the entire file into memory)
  - Read the file line-by-line (makes more sense with text-based files)
  - Read the file certain number of bytes at a time (e.g. 1 character at a time)

- For this course, we are only going to cover one version of **line-by-line** method to keep things simple
  - Please use the office hours if you are interested in learning more! However, you will not be responsible for the other methods
  - Hint: `read()` and `readline()` methods

File Reading: Line-by-Line

```
file = open(file_name, 'r')
```

- We can use a for each loop to read the file line by line:

```
for line in file:
    do_something_with_the_line
    …
```

- This reads the files from the memory line by line as well (so the entire file is not stored at once – we can read large files and process them)
- The loop stops once the end of the file is reached

File Reading: Line-by-Line

```
file = open(file_name, 'r')
for line in file:
    do_something_with_the_line
    ...
```

- The `line` object includes every character from the last file position to the newline character.
- Use the `rstrip` string method to get rid of this if needed

File Writing

- There are multiple ways to write to a file in Python. Common ones:
  - Write a string directly
  - Write a collection/sequence of strings

- For this course, we are only going to cover writing a string directly to keep things simple
  - Please use the office hours if you are interested in learning more! However, you will not be responsible for the other methods
  - Hint: `writelines()`

File Writing: Strings

```
file = open(file_name, 'w')
```

- We use the `write` file method to write strings:

```
s = 'Hello World!'
file.write(s)
```

- This overwrites the file and does not put a newline character
  - You can concatenate your string with `'\n'` (`file.write(s + '\n')`) OR
  - Put `file.write('\n')` wherever you want to have a newline

File Writing: Append

```
file = open(file_name, 'a')
```

- We use the `write` file method to write strings:

```
s = 'Hello World!'
file.write(s)
```

- This appends the string at the end of the file

Working with Files in Python

- Python has a lot of built-in functions to work with files
- In order to work with a file we must first "**open**" it

```
file = open(file_name, access_mode)
```

- Then we can process it:
  - Read from a file: The foreach loop
  - Write to a file: The `write` file met
- We need to close it afterwards!

```
file.close()
```

- Spyder demo

Writing and Reading Numeric Data

- **Numbers must be converted to strings before they are written to a file**
- `str` function: converts value to string
- **Number are read from a text file as strings**
  - Must be converted to numeric type in order to perform mathematical operations
  - Use `int` and `float` functions to convert string to numeric value

More Details for the Interested

- You can do more complicated stuff like processing a file backwards (from its end to its start)
- When you open a word document, the cursor is set to the beginning
- When you open a file “the file pointer” points to the files beginning
- You can manually move this pointer around just like you would a cursor.
- This allows you to be more flexible in processing a file
- Hint for the interested: `tell` and `seek` methods of the file class