Read & write text files

- Suggested video Text files in Python by Socratica

Read entire file as a string

```
with open("countries.txt", "r") as f:   # Open file for reading
    content = f.read()                  # Load entire file into 1 large string
    print(content)                      # Do something with the string
```

Read entire file as a list, one string per line

```
with open("countries.txt", "r") as f:   # Open file for reading
    content = f.read()                  # Load entire file into 1 large string
    lines = content.splitlines()        # Split into lines
    for line in lines:
        print(line)
```

Writing a text file - Using just a string

```
content = "My exciting material"
with open("stuff.txt", "w") as f:   # Open file stuff.txt for writing
    f.write(content)                # Write this string to the file
```

Writing a text file - Using a list of strings

```
content = ['Leah', 'Obi-wan', 'Yoda', 'Rey', 'Finn', 'bb-8']
save = "\n".join(content)           # Convert list to a string, adding a new-line character
after each string
with open("people.txt", "w") as f:  # Open file people.txt for writing
    f.write(save)                   # Write this string to the file
```

Add to a file without replacing the original content

```
content = "More exciting material"
with open("stuff.txt", "a") as f:   # Open file stuff.txt for appending
    f.write(content)                # Add this string to the file
```

About files...

- The `with` statement will close the file when you unindent
- `.splitlines()` behaves like `.split("\n")`
- Opening a file using `w` mode will overwrite an existing file
- For greater predictability, cast everything to strings before writing to files

By the way, just before we finish the section on reading/writing files, you may have wondered what the "r" or "w" in the `open()` function meant. This instructs Python how we want to access the file we request. The different modes for opening a file that will be relevant for now are as follows:

- `r` for reading
- `w` for writing (erasing it if it exist)
- `a` opens for appending

OS tools

There is a range of operating system functionality that you will commonly use with working with files.

Some functions that will be useful

```
# Import the Operating System module
import os

# Check if a file or folder exists of a given name
if os.path.exists("filename.txt"):
    print("The item exists")

# Check if a file exists
if os.path.isfile("filename.txt"):
    print("The item is a file")

# Check if a folder exists
if os.path.isdir("documents"):
    print("The item is a folder/directory")

# Delete a file
# - USE WITH CAUTION
os.remove("file.txt")

# Rename a file
os.rename("oldfile.txt", "newfile.txt")

# Create a folder
# - Will error if it already exists
os.mkdir("folder")

# Remove a folder
# - Will error if it is not empty
os.rmdir("folder")

# Not relevant to files, but fun to know about....
# Get current logged in user
username = os.getlogin()
```

Problem set

1. Read a text file into a string, print it to the screen.
2. Read a text file into a list of strings, and print out the number of lines in the file.
3. Ask the user for the name of a file they'd like to create. Ask the user to type an input, and then save that as the content of the file.
4. Ask the user for the name of a file they'd like to create. Using a while loop, keep ask the user to type an input and only stop when they enter an empty input. Save all the lines entered as the content of the file.
5. Read a list of strings from a text file. Tell the user how many lines there are and ask them to enter a line number indicating one they would like to read. Print just the content of that line to the user.
6. Read a list of strings from a text file. Tell the user how many lines there are and ask them to enter a line number indicating one they would like to change. Prompt the user for the new content of

the relevant line. Write to the file the new list of strings.

Challenge question

Write a simple to-do app. Each line represents an task. The first character should be – if the task still needs doing, and x once it has been marked as complete. When the program starts, it should present the user with 3 options as follows.

```
Welcome to simple-to-dos!

* Enter the task number to mark it as complete,
* Hit enter on an empty line to quit, or
* write some text to add a new task.

The tasks currently pending are:

1 Cook dinner
2 Clean bedroom

Your input >>
```

An example of the text file follows.

```
x Write blog post
- Cook dinner
- Clean bedroom
x Make weekly tiktok
x Subscribe to Mr B TV
```