Chapter 10: Using files for Reading and Writing
What are files?

- Files are named large collections of bytes on permanent storage.
- Files typically have a **base name** and a **suffix**
- Example: `arch.jpg`
  - Base name: `arch`
  - Suffix: `.jpg`
- Files exist in **directories** (sometimes called **folders**)

```python
>>> file = pickAFile()
>>> print(file)
C:\Documents and Settings\aalban\Desktop\CS131\mediasources\arch.jpg
```
Directories

- Directories can contain files or other directories.
- There is a base directory on your computer, sometimes called the *root directory*
  - Example: C:\
- A complete description of what directories to visit to get to a file is called a *path*
We call this structure a “tree”

- C:\ is the *root* of the tree.
- The tree has *branches*, each of which is a directory.
- Any directory (branch) can contain more directories (branches) and files (*leaves*)
How to open a file

- We open files to either read them or write on them
- `open(filename, how)`
  - `filename`: the name (and path) of the file to be opened
    - If you don’t provide a full path, the file is assumed to be in the same directory as JES.
  - `how` is a two character string that says what you want to do with the file.
    - “rt” means “read the file as text”
    - “wt” means “write the file as text”
    - “rb” and “wb” means read or write binary
      - To manipulate binary files (e.g. JPEG, WAV, Word, Excel, etc)
File methods

- `open()` opens a file and returns a file object that you use to manipulate the file.

- `read()` reads the whole file as a single string. Returns a string.

- `readlines()` reads the whole file into a list where each element is a single line (a string). Returns a list of strings.
  - Each line ends with the new line character (‘\n’)

- `read()` and `readlines()` can only be used once without closing and reopening the file.

- `readline()` reads one line of the file at the time. It returns a string.
File methods and functions

- **write(aString)** writes a string to the file.

- **close()** closes the file.
  - The file cannot be manipulated by the program until it’s opened again.

- Remember that if you are working with numbers:
  - If you read “numbers” from the file you need to convert them from string format to numeric format using the **int** and **float** functions
  - If you are writing numbers to the file, then you need to convert them from the numeric format to the string format using the **str** function.
Example

```python
>>> file = open("data.txt", "rt")
>>> content = file.read();
>>> file.close()
>>> file2 = open("results.txt", "wt")
>>> file2.write("Here are the results \n")
>>> file2.close()
```

- Note: when you open a file for writing, if the file already exits, the contents of the file will be overwritten.
Examples

- Opening, reading, and closing files - from the command area
- Writing on files – from the command area
  - formLetter()
  - changeLittle()
  - replaceWord()
The OS Module

- The OS module offers a number of powerful capabilities for dealing with files, e.g., renaming files, finding out when a file was last modified, and so on.
- We start accessing the OS module by typing:
  ```python
  import os
  ```
- The function that knows about directories is `listdir()`, used as `os.listdir(somePath)`
  - takes a path to a directory as input.
  - returns a list with the base filename and suffix of the files in the input directory
- Example:
  ```python
  >>> print os.listdir(r"C:\MediaSources")
  ```
Creating paths

- If you need the complete path a for a filename returned by `listdir()` then you need to combine the input directory given to `listdir` and the base filename
- But `directory + filename` is not enough
- You still need a path delimiter, like “/”
  - It’s different for each platform (e.g. Windows, MacOS)
  - Python gives us a notation that works: “//” is as a path delimiter for any platform.
- So use this to get the complete path to a file:
  `directory + "//" + filename`
Creating paths: Example

```python
>>> files = os.listdir(r"C:\mediasources")
>>> fileName = files[5]
>>> print fileName
a440.wav

>>> fullPath = "C:\mediasources" + "/" + fileName
>>> print fullPath
C:\mediasources//a440.wav

>>> sound = makeSound(fullPath)
```