Files in Python

File

- Data is very important. To store data in computer we need files.
- File handling is an important part of any application.
- Python has several functions for creating, reading, updating, and deleting files.

Type of files in python

- Text files : Store data in the form of character
- Binary files :
  - Store data in the form of bytes(Highly suitable to store images)

File Handling

- The key function for working with files in Python is the open() function.
- The open() function takes two parameters; filename, and mode.
- There are four different methods (modes) for opening a file:
  - "r" - Read - Default value. Opens a file for reading, error if the file does not exist
  - "a" - Append - Opens a file for appending, creates the file if it does not exist
  - "w" - Write - Opens a file for writing, creates the file if it does not exist
  - "x" - Create - Creates the specified file, returns an error if the file exists
  - “ w+” - To write and read data of a file.The previous data in the file will be deleted
  - “r+” - To read and write data into a file.Previous data will be deleted.file pointer is placed at the beginning of the file.
  - “a+” – To append and read data of file.The file pointer will be at the end of the file if the file exits. If the file does not exits,it creates a new file for reading and writing.

File Handling (Cont…)

- In addition you can specify if the file should be handled as binary or text mode
  - "t" - Text - Default value. Text mode
  - "b" - Binary - Binary mode (e.g. images)

Open a File

- To open the file, use the built-in open() function.
- The open() function returns a file object.
- Open() has a read() method for reading the content of the file.

- Syntax:
  - f = open(“filename", "r")
  - Or
  - f = open("D:\\myfiles\ filename", "r")
  - print(f.read())

Read Only Parts of the File

- By default the read() method returns the whole text, but you can also specify how many characters you want to return.

- f = open("demofile.txt", "r")
- print(f.read(10))

- Return the 10 first characters of the file.

Read Lines

- You can return one line by using the readline() method.
  - f = open("demofile.txt", "r")
  - print(f.readline())
- By calling readline() two times, you can read the two first lines:
  - print(f.readline())
  - print(f.readline())
- By looping through the lines of the file, you can read the whole file, line by line:
  - f = open("demofile.txt", "r")
  - for fileline in f:
  -   print(fileline)

Close Files

- Close() function is used to close a file after you are finish with it.
    - f = open("demofile.txt", "r")
    - print(f.readline())
    - f.close()
    - You should always close your files, in some cases, due to buffering, changes made to a file may not show until you close the file.

Python File Write

- To write to an existing file, you must add a parameter to the open() function:
- "a" - Append - will append to the end of the file
- "w" - Write - will overwrite any existing content

```
f = open(“filename", "a")
f.write(“Data is appended to the file!")
f.close()

#open and read the file after the appending:
f = open("demofile2.txt", "r")
print(f.read())
```

```
f = open(“filename", "w")
f.write(“I have overwrite the content…")
f.close()

#open and read the file after the appending:

f = open("filename", "r")
print(f.read())
```

Create a New File

- open() method is used.
- "x" - Create - will create a file, returns an error if the file exist
- "a" - Append - will create a file if the specified file does not exist
- "w" - Write - will create a file if the specified file does not exist

- Create a file called “file1.txt":
- f = open("file1.txt ", "x")

- Create a new file if it does not exist:
- f = open("file1.txt ", "w")

Delete a File

- To delete a file, you must import the OS module, and run its os.remove() function.
  - import os
  - os.remove(“filename")
- Check if File exist:
- To avoid getting an error, you might want to check if the file exists before you try to delete it

-
  ```
  import os
  if os.path.exists(" filename"):
    os.remove(" filename")
  else:
    print("The file does not exist")
  ```

Delete Folder

- Remove the folder “foldername":
    - import os
    - os.rmdir(" foldername")

seek() and tell() method

- Tell() : to know the current position of file pointer
    - Pos=fileObj.tell()

- ❑ Seek() : To move file pointer to another position
    - ❑ f.fileObj.seek(offset,fromwhere)
        - ❑ Fromwhere : 0,1 or 2 (0 (default) : beginning, 1 : current position, 2 : ending of file)