Ques. What is File Handling in Python?

- Suppose you are working on a file saved on your personal computer. If you want to perform any operation on that file like opening it, updating it or any other operation on that, all that comes under File handling.
- Types Of File in Python
- Binary file
    - **Document files:** .pdf, .doc, .xls etc.
    - **Image files:** .png, .jpg, .gif, .bmp etc.
    - **Video files:** .mp4, .3gp, .mkv, .avi etc.
    - **Audio files:** .mp3, .wav, .mka, .aac etc.
    - **Database files:** .mdb, .accde, .frm, .sqlite etc.
    - **Archive files:** .zip, .rar, .iso, .7z etc.
    - **Executable files:** .exe, .dll, .class etc.
- Text file
    - **Web standards:** html, XML, CSS, JSON etc.
    - Source code: c, app, js, py, java etc.
    - Documents: txt, tex, RTF etc.
    - Tabular data: csv, tsv etc
    - Configuration: ini, cfg, reg etc

Ques. Opening and Closing a File in Python?

open() Function:- This function takes two arguments. First is the filename along with its complete path, and the other is access mode. This function returns a file object.

```
open(filename, mode)
```

Note:- By default, the access mode is read mode if you don't specify any mode.

Ques. How to create a new file?

- create a new file:- "x" - Create - will create a file, returns an error if the file exist.

```
f = open("myfile.txt", "x")
```

How to read file in python

- Open a File on the Server:- The open() function returns a file object, which has a **read() method** for reading the content of the file.

```
file = open("demofile.txt", "r")
print(f.read())
file.close()
```

- 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(5))

Output:- Hello
```

- The **readline() function** helps you read a **single** line from the file.

```
file_obj = open("example.txt", "r", encoding="utf-8")
print(file_obj.readline())
file_obj.close()

Output:-
Hello World
```

- As the name suggests, the **readlines() method** reads **all the lines** in the file and returns them properly separated in a list.

```
file_obj = open("example.txt", "r", encoding="utf-8")
print(file_obj.readlines())
file_obj.close()

Output:-
This is a new file.
We have now learnt all read functions.
We are trying a new method
to loop over files.
```

- **Close Files:-** It is a good practice to always close the file when you are done with it.

```
file = open("demofile.txt", "r")
print(f.readline())
file.close()
```

How to Write a File in Python

- We're creating it using the **w** mode. Once we open the new file, it's obviously empty. We're then going to write content into it.
- Write - will overwrite any existing content and Create a new file if it does not exist.

```
file_obj = open("writing.txt", "w")
```

- **a** Append - will append to the end of the file

```
file = open("writing.txt", "a")
file.write("This way, I will preserve the existing contents in the file")
print(file.read())
file.close()
```

Delete File and folder?

- **Delete File** To delete a file, you must **import the OS module**, and run its os.remove() function

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

- **Delete Folder** To delete an entire folder, use the os.rmdir() method
- It can only remove empty folders.

```
import os
os.rmdir("myfolder")
```

Modes	Description
r	Opens a file in read-only mode. The pointer of the file is at the beginning of the file. This is also the default mode.
rb	Same as r mode, except this opens the file in binary mode.
r+	Opens the file for reading and writing. The pointer is at the beginning of the file.
rb+	Same as r+ mode, except this, opens the file in binary mode.
w	Opens the file for writing. Overwrites the existing file and if the file is not present, then creates a new one.
wb	Same as w mode, except this opens the file in binary format.
w+	Opens the file for both reading and writing, rest is the same as w mode.

Modes	Description
wb+	Same as w+ except this opens the file in binary format.
a	Opens the file for appending. If the file is present, then the pointer is at the end of the file, else it creates a new file for writing.
ab	Same as a mode, except this opens the file in binary format.
a+	Opens the file for appending and reading. The file pointer is at the end of the file if the file exists, else it creates a new file for reading and writing.
ab+	Same as a+ mode, except this, opens the file in binary format.

using with statement?

- The method shown in the above section is not entirely safe. If some exception occurs while opening the file, then the code will exit without closing the file.

```
# Opening file in read mode and printing the contents of the file.
with open("test.txt", mode='r') as f:
    data = f.readlines() #This reads all the lines from the file in a list.
    print(data) #This will print the content of the Hello World file!

# Opening a file in write mode.
with open("test.txt", mode='w') as f:
    f.write("Data after write operation")
# Opening file in read mode to check the contents.
with open("test.txt", mode='r') as f:
    data = f.readlines() # this reads all the lines from the file in a list.
    print(data) #this will print the overwritten content of the file that is "Data after write operation"

# Opening a file in append mode and appending data to the file.
with open("test.txt", "a") as f:
    f.write(" Appending new data to the file")
# Opening file in read mode to check the contents.
with open("test.txt", mode='r') as f:
    data = f.readlines() #This reads all the lines from the file in a list.
    print(data) #this will print the existing content of file plus the appended content
```

Ques. How do you remove a file from a folder in python?

????p

Ques. Program to Delete all files with a specific extension?

```
import os
from os import listdir
```

```
my_path = 'C:\Python Pool\Test\'
for file_name in listdir(my_path):
    if file_name.endswith('.txt'):
        os.remove(my_path + file_name)
```