Why we need file handling?

- ☐ RAM is volatile memory that loses its data when the computing device is turned off.
- ☐ If the data is planned to be used in future, it needs to be stored in non-volatile memory such as hard disk.
- ☐ On a hard disk, a File is a named location used to store the data.
- ☐ In order to read (or write) data from (or on) a file, the file needs to open and after finishing the read (or write), the file needs to be closed.

Python File Handling

- ☐ Following access modes are used to open a file.

r	**for reading an existing file.**
r+	**for both reading and writing.**
w	**for writing only. Creates a new file if it does not exist or truncates the file if it exists.**
'x'	**Open a file for exclusive creation. If the file already exists, the operation fails.**
w+	**for writing and reading.**
a	**for appending.**
a+	**for both appending and reading.**

- ☐ To read a file in the binary format, 'b' is added to the access modes.

Python File Objects

- ☐ Python has built-in file object that allows the user to access and manipulate a file.
- ☐ A file may be opened using the built-in function open() that allows to open a given file in a specified access mode and and returns a file object.

  Syntax

  open(address, mode)

  address: is the file's address

  mode: is the access mode of the file object.
- ☐ Te default format to read a file is the text format. In text mode, after reading the file, we get strings.

Closing a file Using Python

- ☐ After finishing the work related to a file opened using a Python file object, the file needs to be closed in order to free up the resources that were tied with the file.
- ☐ The Python close() method is used for closing a file.

```python
MyFile = open("Sample.txt", 'r')
# perform file operations
MyFile.close()
```

Reading a Python file

- To read the data from a file it needs to be opened in a rereading mode.

```
#File Sample.txt

Good Morning everyone!
Welcome to IC152.
```

The read(count) method

```python
MyFile1 = open("Sample.txt",'r')

print(MyFile1)
S = MyFile1.read()
print(type(S))
print(S)
MyFile1.close()
```

```
<_io.TextIOWrapper name='Sample.txt' mode='r'
    encoding='utf-8'>
<class 'str'>
Good Morning everyone!
Welcome to IC152.
```

The read(count) method

- A negative value of the parameter ‘count’ or calling read() method without the parameter ‘counts’ reads the complete file and returns the contents of the file as a string object.
- A non-negative value of the parameter ‘count’ would read at most ‘count’ bytes from the file.

The read(count) method

```python
MyFile2 = open("Sample.txt",'r')
S = MyFile2.read(12)     # read in the first 12 bytes
print(S)
S = MyFile2.read(20)     # read in the next 20 bytes
print(S)
S = MyFile2.read()     # read in the remaining of the file
print(S)
S = MyFile2.read()     # read in the remaining of the file
print(S)
MyFile2.close()
```

```
Good Morning
 everyone!
Welcome t
o IC152.
```

Reading line-by-line using for loop

- ☐ If the file size is large, one of the most efficient and fast way to read the file is to read the data/records line-by-line.
- ☐ A for loop can be used for reading file line-by-line

```python
MyFile3 = open("Sample.txt",'r')
for Record in MyFile3:
    print(Record, end= '')
```

```
Good Morning everyone!
Welcome to IC152.
```

readline(count) method

```python
MyFile4 = open("Sample.txt",'r')
print(MyFile4.readline(2))
print(MyFile4.readline(-1))
print(MyFile4.readline(0))
print(MyFile4.readline(40))
MyFile4.close()
```

```
Go
od Morning everyone!


Welcome to IC152.

```

readline(count) method

- ☐ A negative value of the parameter ‘count’ or calling readline() method without the parameter ‘counts’ method reads individual lines of a file (including the newline character) starting from the first line and returns a string.
- ☐ A non-negative value of the parameter ‘count’ would return a string of maximum byte ‘count’ (including the trailing newline).
- ☐ Any call after reaching EOF would lead to return of an empty string.

readlines(count) method

- ☐ readlines() method reads a file line by line (including the newline character) till the EOF and return a list containing the lines thus read.
- ☐ If the optional ‘count’ argument is present, instead of reading up to EOF, whole lines totalling approximately ‘count’ bytes (possibly after rounding up to an internal buffer size) are read.

readlines(count) method

```
MyFile5 = open("Sample.txt",'r')
S = MyFile5.readlines(5)
print(S)
S = MyFile5.readlines(3)
print(S)
MyFile5.close()
MyFile5 = open("Sample.txt",'r')
S = MyFile5.readlines(23)
print(S)
MyFile5.close()
```

```
['Good Morning everyone!\n']
['Welcome to IC152.']
['Good Morning everyone!\n', 'Welcome to IC152.']
```

flush method

- ☐ close() automatically flushes the internal buffer data before closing the file.
- ☐ flush() method is used when user desires to flush the data before closing the file. This method has no return value.

readable() Method

- ☐ The readable() method returns True if the file is readable, False if not.

```
MyFile9 = open("Sample.txt",'r'fds)
print(MyFile9.readable())
MyFile9.close()
```

```
True
```

write(string) method

- ☐ This method takes a string as an input and write to a file and does not have a return value.
- ☐ Sometimes due to buffering, it may require to call flush() method to show up the contents of the string passed as an argument.

write(string) method

```
MyFile5 = open("Sample.txt",'a')
MyFile5.write('\nHow are you doing in the course.')
MyFile5.close()
MyFile5 = open("Sample.txt",'r')
S = MyFile5.read()
print(S)
MyFile5.close()
```

```
Good Morning everyone!
Welcome to IC152.
How are you doing in the course.]
```

writelines(List) method

- ☐ writelines(List) method writes the items of a list object to file.
- ☐ Where the texts will be inserted depends on the file mode and stream position.
- ☐ If the file is open in the append mode "a", the text will be inserted at the current file stream position (default at the end of the file).
- ☐ If the file is open in the write mode "w", the file will be emptied before the text will be inserted (default at 0).

writable() Method

- ☐ The writable() method returns True if the file is writable and False if not.
- ☐ A file is writable if it is opened using "a" (for appending) or "w" (for writing).

```
MyFile10 = open("Sample.txt",'a')
print(MyFile10.writable())
MyFile10.close()
```

```
True
```

writelines(List) method

```
MyFile6 = open("Sample.txt",'a')
MyFile6.writelines(['\nQuiz I solutions,', '\nQuiz I answer
    sheets'])
MyFile6.close()
MyFile = open("Sample.txt",'r')
S = MyFile5.read()
print(S)
MyFile5.close()
```

```
Good Morning everyone!
Welcome to IC152.
How are you doing in the course.
Quiz I solutions.
Quiz I answer sheets.
```

tell()

- This method returns an integer that represents the file object’s position from the beginning of the file in the form of bytes.

Sample1.txt

```
B15001,Ramesh,CS,12
B15002,Rajesh,CE,14
B15004,Pawan ,ME,10
B15005,Ram   ,EE,16
B15003,Shivam,DS,15
```

```
>>>my_file = open("Sample1.txt")
>>>S = my_file.read(13)
>>>print(S)
B15001,Ramesh
>>>print(my_file.tell())
13
>>>my_file.close()
```

seekable() Method

- The seekable() method returns True if the file is seekable and False if not.
- A file is seekable if it allows access to the file stream, like the seek() method.

```
MyFile10 = open("Sample.txt",'r')
print(MyFile10.seekable())
```

```
True
```

seek() Method

- seek() method is used to change the file object’s position in a file stream.

Syntax:

seek(Nbytes, Current)

- Nbytes: indicates the number of bytes to be moved.
- Current(optional): the position from where the bytes are to be moved. By default the current takes value 0, which means absolute file positioning (i.e. file beginning position), other allowed values are 1, which means seek relative to the current position and 2 means seek relative to the file’s end.

Sample1.txt

```
B15001,Ramesh,CS,12
B15002,Rajesh,CE,14
B15004,Pawan ,ME,10
B15005,Ram   ,EE,16
B15003,Shivam,DS,15
```

```
>>>my_file = open("Sample1.txt",'r')
>>>print("Current position:",my_file.tell())
Current position: 0
>>>my_file.seek(7)
>>>print("Current position:",my_file.tell())
Current position: 7
>>>print(my_file.read(6))
Ramesh
>>>print("Current position:",my_file.tell())
Current position: 13
>>>my_file.seek(4,1)
UnsupportedOperation: can't do nonzero cur-relative seeks
```

Sample1.txt

```
B15001,Ramesh,CS,12
B15002,Rajesh,CE,14
B15004,Pawan ,ME,10
B15005,Ram   ,EE,16
B15003,Shivam,DS,15
```

```
>>>my_file = open("Sample1.txt",'rb')
>>>print("Current position:",my_file.tell())
Current position: 0
>>>my_file.seek(7)
>>>print("Current position:",my_file.tell())
Current position: 7
>>>print(my_file.read(6))
b'Ramesh'
>>>print("Current position:",my_file.tell())
Current position: 13
>>>my_file.seek(4,1)
>>>print(my_file.read(2))
b'12'
```

Deleting a File

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

```
import os
os.remove("Sample.txt")
```

- ☐ Deleting file that it does not exists will give an error message:

```
>>>import os
>>>os.remove("Sample.txt")
FileNotFoundError: [WinError 2] The system cannot find the
    file specified: 'Sample.txt'
```

Sample1.txt

```
B15001,Ramesh,CS,12
B15002,Rajesh,CE,14
B15004,Pawan ,ME,10
B15005,Ram   ,EE,16
B15003,Shivam,DS,15
```

```
my_file = open("Sample1.txt",'br')
my_file.seek(-12,2)
print(my_file.read(6))
b'Shivam'
my_file.seek(4,1)
print(my_file.read(5))
b'15'
my_file.seek(0,0)
print(my_file.readline(6))
b'B15001'
```

Deleting a File

- ☐ To avoid getting an error, one must check if the file exists before trying to delete it:

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

```
The file does not exist
```

Exercise

- ☐ Find the output of the code given below.

#File Sample.txt

```
Good Morning everyone!
Welcome to IC152.
```

```python
my_file = open("Sample.txt",'r+')
my_file.write("How are you doing in the course?")
my_file.seek(0)
print(my_file.read())
my_file.close()
```

Exercise

- ☐ Find the output of the code given below.

#File Sample.txt

```
Good Morning everyone!
Welcome to IC152.
```

```python
my_file = open("Sample.txt",'w+')
my_file.write("How are you doing in the course?")
my_file.seek(0)
print(my_file.read())
my_file.close()
```