Date :

~~Regular Expressions~~

# FILE I/O

↳ working with files

```
# my_file = open("input.txt")
print(my_file)
```

name of file

open a file

$\boxed{\text{open()}}$ returns file object with name of file and mode default = r

# Read input from file

(file contents

↳ $\boxed{\text{read()}}$

```
input_txt = ~~myfile~~ my-file .read( )
print(input_txt)
```

* Note :-

① The contents of the file are read by a cursor in Python.

② cursor will be at end of file after read.

- print(my_file.seek(0)) → offset = sets current file position (cursor position)

$\boxed{\text{seek( )}}$

- print(my_file.read ine())

- $\boxed{\text{readline ( )}}$ → returns one line from file.

$\boxed{\text{readlines ( )}}$ → returns a list of lines from file eg ~~#~~ If you want to read first 2 lines of file

=) .readline( )

.read line ( )

~~r44~~ At the end, we have to close the file after we opened it

=) my_file.close ( )

~~#~~ Modes →m

① r → read to file (default mode)

② w → write to file (overwrites file content)

→ also creates a new file (if file does not exist)

③ a → append to file content

4 r+ → read and write to file

e.g. ← to open diff files, ~~open~~ create diff file objects.

```
my_file = open("input_file.txt", mode="r+")
input_text = my_file.read( )
print(input_text)

my_file.write(" hey its me")
my_file.close( )
```

## File paths

* Note :- When working with files, we must place in a try-except block → to handle File Exceptions

There are 2 file paths

1 relative path → relative to where I am

* ./ = from current folder

* ../ = go back a folder from current

② Absolute path → entire file path/directory from root folder

e.g. - C:/Users/HP/Desktop/input_file.txt

d **Example**

```
try:
    file_path = input("Enter the file path = ")
    my_file = open(file_path, mode = "r+")
#   ~~Print~~ input_txt = my_file.read( )        or  readline()
    print(input_txt)

    result = "/+ abcd"
#   prefix = "\n" + f"{result}"
    my_file.write(prefix)

    my_file.close( )
except FileNotFoundError as err:
        print("File not found")
        raise err
except IOError as err:
        print("Error in I/O by machine")
        raise err
```