WORKING WITH PYTHON FILES:
File is collection of related information. In Python, a file operation takes place in the following order.

1 Open a file 2 Read or write a file
3 Append a file 4 Close the file

Opening a file
Python has a built-in function open() to open a file. This function returns a file object. It returns a "file handle" - a variable used to perform operations on the file. file handle has.!
1.Open 2.Write 3.Read 4. Close

Syntax
FileObject=open(file_name [, access_mode][, buffering])

Parameter details:
file_name: It contains the name of the file.
access_mode: Default file access mode is read (r).
buffering: The buffering value 0, no buffering takes place. If 1, line buffering is performed.

Example:
```
FileObj=open("AnyName.txt")#relative path
FileObj=open("C:\\Python33\\AnyName.txt") #(AbsolutePath)
```

File Different Modes:

Modes	Description
r	Opens a file for reading only. (default)
b	Opens in binary mode.
r+	Opens a file for both reading & writing.
rb+	Opens a file for both reading & writing in binary format
w	Opens a file for writing only.
a	Opens a file for appending.
a+	Opens a file for both appending and reading.
't'	Opens in text mode. (default)
x	Open a file for exclusive creation. If file already exists Operation fails.

The file Object Attributes

Attribute	Description
file.name	Returns name of the file
file.mode	Returns access mode
writable()	Returns boolean value
readable()	Retruns boolean value
file.closed	Retruns boolean value

The best way to do this is using the with statement. This ensures that the file is closed when the block inside with is existed. We don't need to explicitly call the close() method. It done Implicitly.

Syntax:
```
with open("MyFile.txt",mode='r',encoding = 'utf-8') as MyFObj:
   # Perform Required File Operations
```

Example:
```
try:
```

```
    with open("MyFile.txt",mode='r',encoding='utf-8') as MyFile:
        print(MyFile.name)
        print(MyFile.mode)
        print(MyFile.closed)
        print(MyFile.readable())
        print(MyFile.writable())
except IOError:
    print("SorryFileNotExisted")
finally:
    print("FinallyBlockSuccess")
```