**FILE HANDLING**

Files are where we organize / store our data in the form of Text Data / Binary Data.

**OPEN**: This method takes the path to the file or the name of the file as the first parameter, mode in which we want to open as the second parameter, and the 3rd parameter is an optional parameter, which is a buffer. If we don’t pass any number then Python will by default use 4096 or 8092 in the virtual machines.

Syntax : f = open(“filename”, “mode”, “buffer”)

**CLOSE**: Syntax: f.close() to close the file

**MODES**:

w : Write : All the contents will be deleted and the will start from the first.

r: Read : To Read the file from the beginning to the end.

a: Append: Current contents will not be deleted, but will add the new contents along.

w+: It’s a combination of Write and Read

r+: Its for read / write and append.

a+: Its for append and read

x: Exclusive Creation Mode: New file will be created with this mode. And if the file exists with the same name, then an error will be thrown.

All these modes will be only for Text Files.

If we are dealing in Binary Files, we have to append b in front of this.

wb / rb / ab/ w+b / r+b / a+b / xb

Code to Write:

# TO OPEN THE FILE FOR WRITTING

f=open(*"file1.txt"*,*"w"*)

dat = input(*"Enter the Text : "*)

f.write(dat)

f.close()

**Output in Console**:

Enter the Text : This is the first line in write mode.

**Output in Folder**:

A black text on a white background

Description automatically generated

**Output in File**:

A close up of a screen

Description automatically generated

Code to Read:

# TO OPEN THE FILE FOR READING

f=open(*"file1.txt"*,*"r"*)

s = f.read()

print(s)

f.close()

**Output in Console**:

This is the first line in write mode.

Code to Write Multiple String:

# TO OPEN THE FILE FOR WRITTING

f=open(*"file1.txt"*,*"w"*)

print(*"Enter the Text (Type # when you are done) "*)

dat =*""*

while dat != *'#'*:

dat= input()

f.write(dat + *"\n"*

f.close()

**Output in Console**:

Enter the Text (Type # when you are done)

Montieth Thursday

Good Friday

Holy Saturday

Easter Sunday

#

**Output in Folder**:

A black text on a white background

Description automatically generated

**Output in File**:

A screenshot of a computer

Description automatically generated

OS, SYS EXAMPLES:

Python has a library called OS module which has a sub module called path. That path has a method called isFile, to check if file exists.

The isFile method returns a Boolean true, if exists, if not false.

Sys module to exit out of our program, if the file does not exists.

To Check if the file exists or not:

Eg1:

# IMPORT os,sys TO CHECK IF THE FILE EXISTS

import os,sys

if os.path.isfile(*"file1.txt"*):

# TO OPEN THE FILE FOR READING

f=open(*"file1.txt"*,*"r"*)

s = f.read()

print(s)

f.close()

else:

print(*"File Not Found...."*)

Eg2:

# IMPORT os,sys TO CHECK IF THE FILE EXISTS

import os,sys

if os.path.isfile(*"file1.txt"*):

# TO OPEN THE FILE FOR READING

f=open(*"file1.txt"*,*"r"*)

else:

print(*"File Not Found...."*)

sys.exit()

s = f.read()

print(s)

f.close()

**Output in Console**:

Montieth Thursday

Good Friday

Holy Saturday

Easter Sunday

File Name wrongly provided Code:

Eg1:

# IMPORT os,sys TO CHECK IF THE FILE EXISTS

import os,sys

if os.path.isfile(*"file2.txt"*):

# TO OPEN THE FILE FOR READING

f=open(*"file1.txt"*,*"r"*)

s = f.read()

print(s)

f.close()

else:

print(*"File Not Found...."*)

Eg2:

# IMPORT os,sys TO CHECK IF THE FILE EXISTS

import os,sys

if os.path.isfile(*"file2.txt"*):

# TO OPEN THE FILE FOR READING

f=open(*"file1.txt"*,*"r"*)

else:

print(*"File Not Found...."*)

sys.exit()

s = f.read()

print(s)

f.close()

**Output in Console**:

File Not Found....

Pickle:

Using Pickle module to serialize an object into a file.

Create a Class:

class **Staff**:

def **\_\_init\_\_**(*self*, sid, sname, ssal):

*self*.staffId = sid

*self*.staffName = sname

*self*.staffSalary=ssal

def **displayStaff**(*self*):

print(*"Staff Id : "*,*self*.staffId)

print(*"Staff Name : "*,*self*.staffName)

print(*"Staff Salary : "*,*self*.staffSalary)

Create a program to insert data using pickle:

# IMPORT pickle and staff Class

import pickle, staff

fileName=open(*"Staff.dat"*,*"wb"*)

staff1=staff.Staff(10751,*"Kumar"*,75280)

pickle.dump(staff1,fileName)

fileName.close()

Output in Console:

No output in console as the file has been created.

Generated File:

A screenshot of a computer

Description automatically generated

UnPickle:

Create a program to retrieve data using pickle.load:

# IMPORT pickle and staff Class

import pickle, staff

fileName=open(*"Staff.dat"*,*"wb"*)

staff1=staff.Staff(10751,*"Kumar"*,75280)

pickle.dump(staff1,fileName)

fileName.close()

Output in Console:

Staff Id : 10751

Staff Name : Kumar

Staff Salary : 75280

Read:

**Code to read from the file:**

fName = open("courses.txt","r")  
print(fName.read())

**Output:**

C:\Users\kamal\pythonlab\Scripts\python.exe C:\Users\kamal\PycharmProjects\pythonProjectdemo\readDemo.py

C# .Net Core

ASP .Net Core

Java

Java Spring Boot

Python

Django

Docker

Kubernates

Process finished with exit code 0

**Code using read(), readline(), readlines()**

fName = open("courses.txt","r")  
print("OUTPUT USING READ()")  
print(fName.read())  
print()  
*# SEEK WILL TAKE THE FILE CURSOR BACK TO THE FIRST LOCATION*fName.seek(0)  
*#NOW PRINT THE READLINE WILL PRINT THE FIRST THE LINE*print("OUTPUT USING READLINE()")  
print(fName.readline())  
print()  
fName.seek(0)  
print("OUTPUT USING READLINES()")  
print(fName.readlines())  
print()  
fName.close()

**Output:**

C:\Users\kamal\pythonlab\Scripts\python.exe C:\Users\kamal\PycharmProjects\pythonProjectdemo\readDemo.py

OUTPUT USING READ()

C# .Net Core

ASP .Net Core

Java

Java Spring Boot

Python

Django

Docker

Kubernates

OUTPUT USING READLINE()

C# .Net Core

OUTPUT USING READLINES()

['C# .Net Core\n', 'ASP .Net Core\n', 'Java\n', 'Java Spring Boot\n', 'Python\n', 'Django\n', 'Docker\n', 'Kubernates']

Process finished with exit code 0

Write:

**Courses.txt file**

A screenshot of a computer

Description automatically generated

**Code using write(), writelines()**

fName = open("courses.txt","w+")  
*#IF WE EXECUTE THIS THE FULL DATE IN THE COURSES.TXT FILE WILL BE ERASED  
# THE BELOW LINE WILL BE ADDED.*fName.write("Softwares which have worked with....")  
fName.write("\n")  
fName.write("====================================")  
fName.write("\n")  
*# WRITELINES WILL TAKE THE DATA AS LISTS*fName.writelines(['C# .Net Core\n', 'ASP .Net Core\n', 'Java\n', 'Java Spring Boot\n', 'Python\n', 'Django\n', 'Docker\n', 'Kubernates'])  
  
*#TO FIND THE CUSOR LOCATION  
#['C# .Net Core\n', 'ASP .Net Core\n', 'Java\n', 'Java Spring Boot\n', 'Python\n', 'Django\n', 'Docker\n', 'Kubernates']*print("Cursor location after entering the above list @ ",fName.tell())  
  
*# TO TAKE THE CURSOR BACK TO FIRST LINE*fName.seek(0)  
print("Cursor location after the seek is 0 - ",fName.tell())  
*# PRINT FROM THE FIRST POSITION*print()  
print(fName.read())  
fName.close()

**Output:**

C:\Users\kamal\pythonlab\Scripts\python.exe C:\Users\kamal\PycharmProjects\pythonProjectdemo\writeDemo.py

Cursor location after entering the above list @ 163

Cursor location after the seek is 0 - 0

Softwares which have worked with....

====================================

C# .Net Core

ASP .Net Core

Java

Java Spring Boot

Python

Django

Docker

Kubernates

Process finished with exit code 0

Append:

**Code**

fName = open("courses.txt","a+")  
print("Cursor location when we start Append operation - ",fName.tell())  
fName.write("\n")  
fName.write("Database which have worked with....")  
fName.write("\n")  
fName.write("====================================")  
fName.write("\n")  
*# WRITELINES WILL TAKE THE DATA AS LISTS*fName.writelines(['SQL Server\n', 'MySQL\n', 'Oracle\n', 'PostgresSQL\n', 'MongoDB\n', 'VBA\n', 'Access\n'])  
fName.seek(0)  
print()  
print(fName.read())  
*#ALSO WE CAN PRINT LIKE THIS TOO*fName.seek(0)  
print()  
print("Another way of printing the data using array...\n")  
arrayData=[]  
for line in fName:  
 arrayData.append(line)  
print(arrayData)  
  
fName.close()

**Output:**

C:\Users\kamal\pythonlab\Scripts\python.exe C:\Users\kamal\PycharmProjects\pythonProjectdemo\appendDemo.py

Cursor location when we start Append operation - 163

Softwares which have worked with....

====================================

C# .Net Core

ASP .Net Core

Java

Java Spring Boot

Python

Django

Docker

Kubernates

Database which have worked with....

====================================

SQL Server

MySQL

Oracle

PostgresSQL

MongoDB

VBA

Access

Another way of printing the data using array...

['Softwares which have worked with....\n', '====================================\n', 'C# .Net Core\n', 'ASP .Net Core\n', 'Java\n', 'Java Spring Boot\n', 'Python\n', 'Django\n', 'Docker\n', 'Kubernates\n', 'Database which have worked with....\n', '====================================\n', 'SQL Server\n', 'MySQL\n', 'Oracle\n', 'PostgresSQL\n', 'MongoDB\n', 'VBA\n', 'Access\n']

Process finished with exit code 0

Count Lines:

**Code:**

fName = open("courses.txt","r")  
print(len(fName.readlines()))  
fName.seek(0)  
*# IS ADDING THE LAST LINE WHICH IS TO BE APPENDED*print(len(fName.read().split("\n")))  
fName.seek(0)  
fName.close()

**Output:**

C:\Users\kamal\pythonlab\Scripts\python.exe C:\Users\kamal\PycharmProjects\pythonProjectdemo\countline.py

19

20

Process finished with exit code 0

**QUIZ**

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated**