Working With Files:

- **Non persistency programs**
  - Non Persistency programs one which enable you to store the data temporarily during the program execution
  - The data which we are passing during executing the program as input and the values which we getting after executing program as an out those values will be stored in the system temporarily

- **Persistency Programs**
  - Persistency program one which enable you to store the data in a system memory permanently.
    1. Persistency programs can be done in 2 ways
       - Using Files
       - Using Databases

- **What is File ?**
  - Files are used to store the data permanently in the system memory.
  - Files are good to store limited data
  - Files doesn't provide any sec for confidential data through username and password
  - To overcome this problem then databases are used
    1. Oracle | sqlServer | MySql | DB2……

- Open()
  - It will create an File object
  - This function two arguments 1.Filename and 2.file mode
  - Example : f=open("Sample","r")

File Modes :

"w: " it will open the file for the purpose of writing the data . if the specified file is already not existed then it will create a new file and allow for writing the data. If the specified file is existed then it will open the file and allow for writing the data into it "here old data is replaced with new data"

"a: " it will open the file for the purpose of appending the data . if the specified file is already not existed then it will create a new file and allow for writing the data. If the specified file is existed then it will open the file and allow adding new data to the existed

"r": it will open the file for reading the Data . it is the default mode . if the specified file not existed then it raise "FileNotFoundError":

"x" : exclusive Mode it will open the file iff when the file is not Existed and allow for writing the data. If the file is already existed then it raise an Error "FileExistsError"

“w+” : write and read

“r+” : read and write

Example

```
f=open("data1","w")
print("Type is : ",type(f))  #<class '_io.TextIOWrapper'>

print("Filename is : ",f.name) #name attribute of TextIOWrapper
print("file mode is : ",f.mode)
print("Is file Writable : ",f.writable() )
print("is file readable : ",f.readable())
print("is File closed ? : ",f.closed)
f.close()
print("is File Closed ? : ",f.closed)
```

- name attribute will return the name of the file
- mode will return the file mode [“w”,”r”,”x” ….]
- writable() will return True if the file mode is writable
- readable() will return True if the file mode is readable
- closed attribute will return True If the if is closed else it will return “False”
- close() method to close the current File Object.

For Writing The Data into the File

write(data) : it will allow to write the data into file

Example:

```
f=open("data2","w")   #file will be created in the same directory
f.write("shashi")
f.write("madhu")
f.write("manasa")
f.write("praveen")
f.write("naveen")
f.close()
print("Data is Saved ")   //Here data will be written in the same line in file
```

Example:

```
f=open("data3","w")   #file will be created in the same directory
f.write("shashi \n ")
f.write("madhu \n")
f.write("manasa \n")
f.write("praveen \n ")
f.write("naveen")
f.close()
print("Data is Saved ") //Here data will be written in the different lines in file
```

writeLines(iterable) : it will allows to write group of lines to the file at a time, but the lines of the data should be in the form of iterable (list | tuple | set)

Example:

```
#writelines(iterable)
lst=["1.welcome to sssit \n","2.have a nice day \n",
        "3.have a good day"]
f=open("data6","w")
f.writelines(lst)
f.close()
print("Data is Saved ")
```

Reading the Data From The User and Store them into the File : Dynamic Data:

```
f=open("data4","w")
print("Enter data press '*' for Exit ")
data=input()
while data!='*':
    f.write(data+"\n")
    data=input()
print("Data is Saved ")
f.close()
```

For Reading the Data From The File:

read(): It will read entire data From the file in a single short and it will return them in the form of "str" type.

Example:

```
f=open("data2","r")
s=f.read()
print("Type is : ",type(s))
print("Data is : ",s)
f.close()
```

read(bytes):

It will read and return the specified no.of. bytes from the specified file

```
#read(bytes) -> str
f=open("data2","r")
s=f.read()
print("Data is : ",s)
print("- "*30)
pos=f.tell()
print("position of File pointer : ",pos)

f.seek(0)
pos=f.tell()
```

```
print("Position of File pointer : ",pos)
nb=f.read(6)
print("Data is : ",nb)
f.close()
```

tell() : it will return current position of the file pointer

seek(position): used to make the file pointer to move to the particular location

readline() : it will read line from the File and it will move next line

Example:

```
f=open("data6","r")
l1=f.readline()
print("type is : ",type(l1))
print(l1)

l2=f.readline()
print(l2)

l3=f.readline()
print(l3)
f.close()
```

readlines() -> list:

It will read all the lines from the file an it will return then in the form of list Collection

```
import time
f=open("data6","r")
lst=f.readlines()
print("type is : ",type(lst))
for line in lst :  #lst is list of lines from the file
    time.sleep(1)
    print(line,end='')
```

Reading the Data From Given File:

```
import time
fname=input("enter filename for reading data ") #shashi
f=None  #global variable
try:
    f=open(fname,"r")
except FileNotFoundError:
    print("Sorry Given File Not Existed ")
else:
    data=f.read()
    for i in data:
        time.sleep(.1)
```

```
        print(i,end='')
finally:
  try:
        f.close()
  except AttributeError:
        pass
```

Example 2:

```
import time
fname=input("enter filename for reading data ") #shashi
f=None  #global variable

try:
   f=open(fname,"r")
except FileNotFoundError:
   print("Sorry Given File Not Existed ")
else:
   data=f.read()
   for i in data:
        time.sleep(.1)
        print(i,end='')
finally:
   if f is not None:
      f.close()
```

Coping Data From One File to Another :

```
import time
src_fname=input("enter src file for reading data ") #data9
fsrc=None   #file for src
ftrg=None    #file for target
try:
   fsrc=open(src_fname,"r")
except FileNotFoundError:
   print("Sorry SRC_File Not Found ")
else:
   print("Src file found ")
   trg_fname=input("enter trg filename ")
   try:
      ftrg=open(trg_fname,"x")
   except FileExistsError:
      print("Target File Existed ...")
   else:
      data=fsrc.read()
      ftrg.write(data)
      print("Data is copied ")
finally:
   if fsrc!=None:
      fsrc.close()
```

```
    if ftrg!=None:
        ftrg.close()
```

Reading Data Using with open()

Syn: with open(filename,filemode) as ref:

==========

==========

- if we any file using with open() or context mode . then it is not required to close the file explicitly reason, the file object will be closed by the PVM whenever the control is comes out from the with suite

Example:

```
import time
fname=input("Enter filename ")
try:
    with open(fname,"r") as f:
        data=f.read()
        for i in data:
            time.sleep(.1)
            print(i,end='')
except:
    print("Sorry File not Found ")
```

Working with CSV Files :

- CSV file is an ordinary text file where the data will be stored in the form row and columns but here column values should be separated by using (,) thus these files are called CSV (Comma separated values) File and file should be saved with an extension is ".csv"

Example:

Student.csv

sno,sname,scity
101,ramesh,hyderabad
102,madhu,vizag
103,manasa,khammam

To Read the Data From CSV

- We have to reader() From CSV module
- Reader function required src _filename " ie.from where to read we need to specify "
- reader() will return Collection of list Objects

Example:

```
import csv
import time
with open("student1.csv","r") as f:
    data=csv.reader(f)
    for row in data:
        time.sleep(1)
        print(row[0],"\t",row[1],"\t",row[2])
```

Writing the Data into CSV Files:

In order to write the data into csv file then we have to writer() method from CSV module . which required specify where to write Target _Filename

writer(File object) -> writer object

- If you want write new row into the CSV File then we have to use writerow( ) from writer class object

writerow(iterable) is from csv.writer object

```
import csv
with open("mystudent2.csv","w",newline='') as f:
    writer=csv.writer(f)
    lst=['sno','sname','course','fee']
    writer.writerow(lst)
```

```
n=int(input("enter no.of.records u want store :"))
for i in range(1,n+1):
    sno=input("enter sno ")
    sname=input("ename sname")
    course=input("enter course ")
    fee=input("enter Fee ")
    writer.writerow([sno,sname,course,fee])
```

Pickling and Unpicking

- process of writing the Objects into file is called picking
- If you want write the objects into the file then we have open the file in "wb" mode [write binary mode]
- To write the objects into the file then we have to use "dump()" from pickle module which required what write and where to write.

Example:

#MyModule.py

```
class Employee:
    def setEmployee(self):
        self.eno=input("Enter eno : ")
        self.ename=input("Enter ename : ")
        self.esalary=input("Enter salary : ")
    def getEmployee(self):
        print("Eno is : ",self.eno)
```

```
    print("Ename is : ",self.ename)
    print("Esalary is : ",self.esalary)
```

Writing the Objects into the File:

```
import pickle
from MyModule import Employee
                        # Emploee class is existed MyModule
f=open("einfo","wb")
n=int(input("Enter no.of.objects do u want store ? : "))
for i in range(1,n+1):
    e=Employee()
    e.setEmployee()
    pickle.dump(e,f)
    print("Object is Saved ")
    print("- "*30)
f.close()
```

Unpacking:

- It is the process of Reading the Objects from the file
- To read the objects from the file then we have to open the file in "rb" read binary mode
- To read the objects from the file then we have to use "load()" from pickle module

```
import pickle
import time
from MyModule import *

with open("einfo","rb") as f:
    try:
        while True:
            time.sleep(1)
            e=pickle.load(f)
            if isinstance(e,Employee):
                e.getEmployee()
            elif isinstance(e,'Student'):
                pass
            print("- "*30)
    except EOFError:
        print("------ End-Of-File -------")
```

Working With Image Files :

- While working any binary files like audio, video files and image file then we have to use “rb” read binary or “wb” write binary modes

#Creating a duplicate image

```
fs=open("D:\SHASHI_CLASSES\SHASHI_CLASSES\PERSONAL\clp.jpg","rb")
ft=open("new_image.jpg","wb")
bytes=fs.read()
ft.write(bytes)
fs.close()
ft.close()
print("image is cloned ...!!! ")
```

Ensuring the specified Path is Existed Or Not :

```
'''exists() from os.path module it will return True if the specified path is exists  '''
import os.path
#b=os.path.exists("demo1.py")
#if b==True:
#if os.path.exists("demo1.py")==True:
if os.path.exists("demo1.py"):
   print("Yes it is Existed ")
else:
   print("Sorry Not Existed ")
```

Checking the Given Path is File Or Directory :

''' exists(path)

- Will return True if the given path is Existed |

isfile(path)

- It will return True if the given Path is File |

isdir(path)

- It will return True if the given Path is Directory
    - these are the function from os.path module '''

```
import os.path
name=input("Enter Name :")
if os.path.exists(name):
    print("Yes it is existed ")
    if os.path.isdir(name):
        print("Yes it is Directory ")
    elif os.path.isfile(name):
        print("Yes it is a File ")
else:
    print("sorry not existed ")
```