Files

What is file?
A file is collection of data or information.
Files are used to store or save data permanently.

Files are two types.

1. Text File
2. Binary File

Text File: In text file data is stored in text format or text file allows only string type of data.

Binary File: In binary file data is stored in byte format or binary file allows only binary data (bytes).
Example of binary files are images, audio, videos, …

Basic steps to work with files

1. Open file
2. Write/Read
3. Close File

open(file, mode='r')
Open file and return a corresponding file object. If the file cannot be opened, an OSError is raised.

file is a path-like object giving the pathname (absolute or relative to the current working directory) of the file to be opened

mode is an optional string that specifies the mode in which the file is opened.

Character	Meaning
'r'	open for reading (default)
'w'	open for writing, truncating the file first
'x'	open for exclusive creation, failing if the file already exists
'a'	open for writing, appending to the end of file if it exists
'b'	binary mode
't'	text mode (default)
'+'	open for updating (reading and writing)

The default mode is 'r' (open for reading text, a synonym of 'rt').
Modes 'w+' and 'w+b' open and truncate the file.
Modes 'r+' and 'r+b' open the file with no truncation.

Text Files
Text files allow data of type string or text.
Types of text file

1. .txt
2. .csv (comma separated values)
3. .json (java script object notation)
4. .xml (extended markup language)

Creating .txt file

```
try:
   a=open("file1.txt","w")
   print("file created...")
except OSError:
   print("error in creating file")
```

Output
file created...

Functions/Methods used for writing text data

1. Write
2. Writelines
3. Print

write(*s*)
Write the string *s* to the stream and return the number of characters written.

Example:

```
try:
   a=open("file1.txt","w") # resource allocation
   print("file created...")
   c1=a.write("Python")
   c2=a.write("3.12")
except OSError:
   print("error in creating file")
except TypeError:
   print("allows only text data")
else:
   print(f'number of characters written {c1+c2}')
finally:
   a.close() # resource deallocation
```

Output
file created...
number of characters written 10

writelines(*lines, /*)
Write a list of lines to the stream. Line separators are not added, so it is usual for each of the lines provided to have a line separator at the end.

Example

```
import sys
try:
   f=open("file2.txt","w")
   list1=["java\n","python\n","C++\n","C\n"]
   f.writelines(list1)
   print("data saved")
except:
   t=sys.exc_info()
   print(t[1])
finally:
   f.close()
```

Output

data saved

print(values,sep=' ',end='\n',file=stdout)

Write a program to create student.txt file and store marks data

```
import sys
try:
   f=open("student.txt","a")
   while True:
      rollno=int(input("Enter Rollno "))
```

```
        name=input("Enter Name ")
        sub1=int(input("Enter Subject1 "))
        sub2=int(input("Etner Subject2 "))
        print(rollno,name,sub1,sub2,file=f)
        ans=input("Add another student?")
        if ans=="no":
            break
except:
    t=sys.exc_info()
    print(t)
finally:
    f.close()
```

Output

```
Enter Rollno 1
Enter Name naresh
Enter Subject1 60
Etner Subject2 70
Add another student?yes
Enter Rollno 2
Enter Name suresh
Enter Subject1 40
Etner Subject2 90
Add another student?yes
Enter Rollno 3
Enter Name ramesh
Enter Subject1 30
Etner Subject2 60
Add another student?no
```

Reading data from text file

1. read()

2. readlines()

reading data from file is done in sequentially and randomly.

read(*size=-1*, /)
Read and return at most *size* characters from the stream as a single str. If *size* is negative or None, reads until EOF.