Files

What is file?

A file is named memory location in hard disk. Files are used to save data permanently.
The objects or data stored within RAM/Main memory is temporary.

There are two types’ files

1. Text file
2. Binary file

Text file contains data in text or string format.
Binary file contain data in bytes format.
Example of binary file: images, audio, video, …

Basic steps to work with files

1. Open file (Read/Writing)
2. Read/Write
3. Close file

open()

It is a predefined function in python. This function opens the file in given mode and return file object.

Syntax: open(“filename”,mode)

Modes

1. “w” : Write Mode, if file is opened in write mode, programmer can able to write data inside file. If given filename is exists it truncates the file and open again. If given file not exists PVM creates new file.
2. “r” : Read Mode, this mode is used to read the content of the file. This opens the file in read mode, if file exists. If file not exists open function raises FileNotFoundError.
3. “a”: Append Mode, this mode allows to add more content to existing file. If file not exists, it will create new file.
4. “x” : exclusive creation mode, this mode is used writing data inside file. If given filename exists open function raises error. if given filename not exists it opens file in writing mode.
5. “w+” : open and truncate the file
6. “r+” : open the file with no truncation.

7. t : text file
8. b : binary file

Note: default type is text and mode “read”

“wt” write+text
“rt” read+text
“wb” write+binary
“rb” read+binary
“a” append+text
“w” write+text
“r” read+text

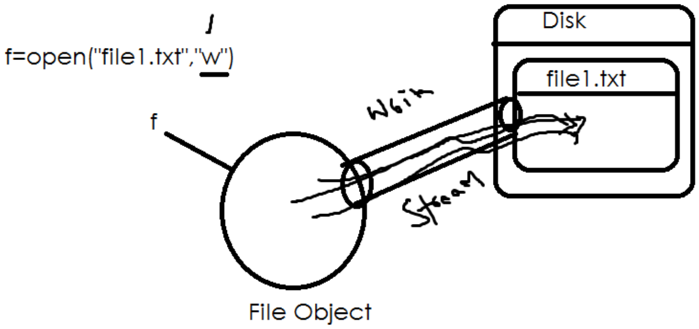
Working with Text Files

Text file is collection of characters. Text file allows writing and reading data in string format.

Writing data inside text file

The following functions are used to write string inside file.

1. write(str)
2. print(values,sep=’ ‘,end=’\n’,file=sys.stdout)



Example1:

```
def main():
    f=open("file1.txt","w")
    f.write("python")
    f.write("java")
    f.write("oracle")
    f.write(''' python is general purpose
    programming langauge''')
    f.write("101")
    f.close()
main()
```

Output:
Output is saved inside file1.txt, created inside project folder

Example2:

```
def main():
    f=open("f:\\python6pmaug\\file1.txt","w")
    f.write("python")
    f.write("java")
    f.write("oracle")
    f.write(''' python is general purpose
    programming langauge''')
    f.write("101")
    f.close()
main()
```

Output:
file1.txt file is created inside given location or path

Example:

```
def main():
    f=open("file1.txt","a")
    f.write('''This is new content''')
    f.close()
main()
```

Output:
More content is added inside file1.txt

Example:

```
# write a program to create student file
# to store the following details
# rollno,name,course

def main():
    f=open("student.txt","a")
    while True:
        rollno=int(input("Enter rollno"))
        name=input("Enter Name")
        course=input("Enter Course")
        print(rollno,name,course,file=f)
        ans=input("Add another Student?")
        if ans=="no":
            f.close()
            break
main()
```

Output:
Output is saved inside student.txt file

Example:

```
# write a program to create marks file to
# store student marks details

def main():
    f=open("marks.txt","a")
    while True:
        rollno,s1,s2=list(map(int,input("Enter rollno,s1,s2").split()))
        print(rollno,s1,s2,file=f)
        ans=input('Add another student marks?')
        if ans=="no":
            f.close()
            break
main()
```

Output:
Output is saved inside marks.txt file

Reading content of file

File object provides the following methods to read content of text file

1. read()
2. readlines()

Syntax: read(size=-1)

This method read the complete file content and return as one string if size=-1. If size is given, it read size characters from file.

Example:

```
# program to read content of file1.txt

def main():
    f=open("file1.txt","r")
    s=f.read()
    print(s)
    f.close()
main()
```

Output

```
pythonjavaoracle python is general purpose
    programming langauge101This is new content
```

Example

```
# write a program to count  how  many
# vowels exists in file1.txt

def main():
    f=open("file1.txt","r")
    c=0
    while True:
        ch=f.read(1)
        if ch=="":
            break
        if ch in "aeiouAEIOU":
            c+=1
    print(f'Vowel count {c}')
    f.close()
main()
```

Output

```
Vowel count 5
```

Example:

```
# write a program to copy content of one file inside another file
def main():
    src=input("Enter Source FileName")
    dest=input("Enter Dest FileName")
    f1=open(src,"r")
    f2=open(dest,"w")
    s=f1.read()
    f2.write(s)
    f1.close()
    f2.close()
main()
```

Output

```
Enter Source FileNamefile1.txt
Enter Dest FileNamefile2.txt
```

Example:

```
# write a program to student details from student.txt

def main():
    f=open("student.txt","r")
    while True:
        stud=f.readline()
        if stud=="":
            break
        rno,name,course=stud.split()
        print(f'{rno}\t{name}\t{course}')
    f.close()
main()
```

Output

```
101   naresh    python
102   suresh    java
103   rajesh    C++
104   kishore   C
```

Example:

```
# read marks.text file
# calculate tot,avg and result

def main():
    f=open("marks.txt","r")
    while True:
        stud=f.readline()
        if stud=='':
            break
        rno,s1,s2=map(int,stud.split())
        tot=s1+s2
        avg=tot/2
        res="pass" if s1>=40 and s2>=40 else "fail"
        print(f'{rno}\t{s1}\t{s2}\t{tot}\t{avg:.2f}\t{res}')
    f.close()
main()
```

Output

```
101   60    90    150   75.00 pass
102   60    30    90    45.00 fail
103   90    60    150   75.00 pass
```

CSV file or csv module

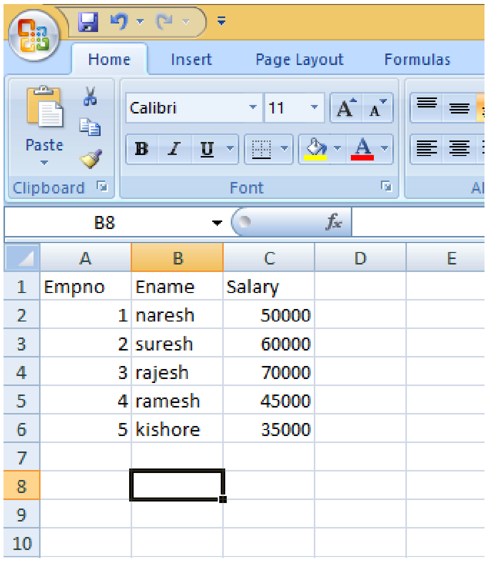
CSV stands for Comma Separated Values. The so-called CSV (Comma Separated Values) format is the most common import and export format for spreadsheets, databases and programming languages.

The csv module implements classes to read and write tabular data in CSV format. It allows programmers to say, “write this data in the format preferred by Excel,” or “read data from this file which was generated by Excel,” without knowing the precise details of the CSV format used by Excel. Programmers can also describe the CSV formats understood by other applications or define their own special-purpose CSV formats.

The csv module’s reader and writer objects read and write sequences. Programmers can also read and write data in dictionary form using the DictReader and DictWriter classes.

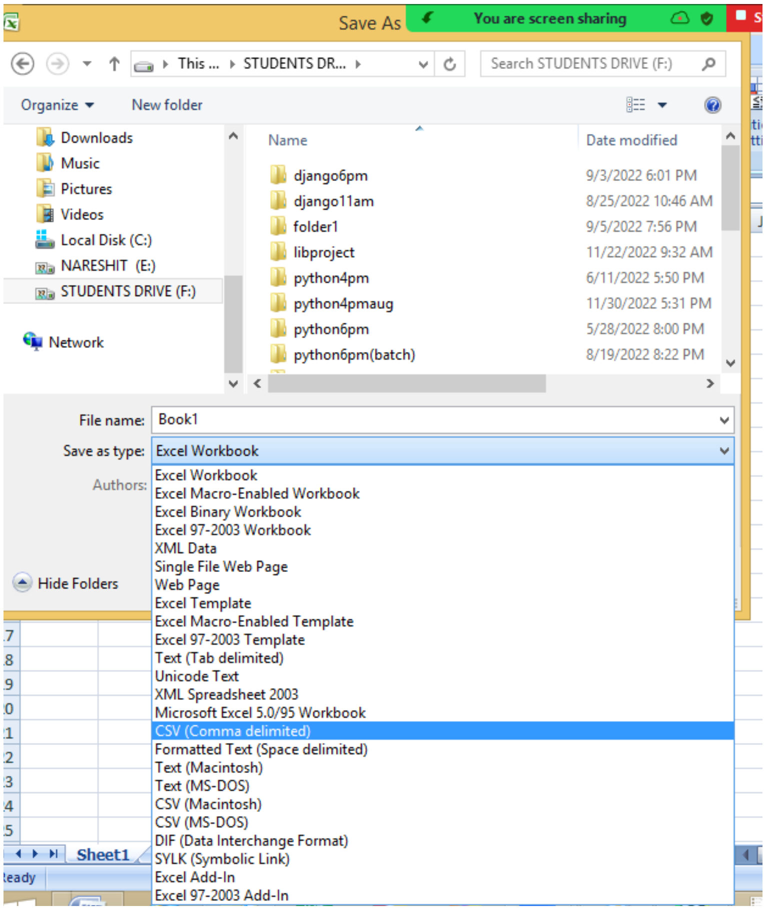
How to create CSV file in MS-Excel?

1. Open Ms-Excel

2. 

	A	B	C
1	Empno	Ename	Salary
2	1	naresh	50000
3	2	suresh	60000
4	3	rajesh	70000
5	4	ramesh	45000
6	5	kishore	35000

3. File→Save



Name	Date modified
django6pm	9/3/2022 6:01 PM
django11am	8/25/2022 10:46 AM
folder1	9/5/2022 7:56 PM
libproject	11/22/2022 9:32 AM
python4pm	6/11/2022 5:50 PM
python4pmaug	11/30/2022 5:31 PM
python6pm	5/28/2022 8:00 PM
python6pm(batch)	8/19/2022 8:22 PM

File name: Book1

Save as type: Excel Workbook

Authors:

4. Employee.csv

reader object
csv.reader(csvfile)

Return a reader object which will iterate over lines in the given csvfile. csvfile can be any object which supports the iterator protocol and returns a string each time its __next__() method is called — file objects and list objects are both suitable.

```
import csv
def main():
    f=open("f:\\employee.csv","r")
    cr=csv.reader(f)
    for row in cr:
        print(row)
    f.close()
    f=open("f:\\employee.csv","r")
    r=csv.reader(f)
    employeeList=list(r)
    print(employeeList)
    for emp in employeeList:
        print(emp[0],emp[1],emp[2])
main()
```

Output

```
['Empno', 'Ename', 'Salary']
['1', 'naresh', '50000']
['2', 'suresh', '60000']
['3', 'rajesh', '70000']
['4', 'ramesh', '45000']
['5', 'kishore', '35000']
[['Empno', 'Ename', 'Salary'], ['1', 'naresh', '50000'], ['2', 'suresh', '60000'],
['3', 'rajesh', '70000'], ['4', 'ramesh', '45000'], ['5', 'kishore', '35000']]
Empno Ename Salary
1 naresh 50000
2 suresh 60000
3 rajesh 70000
4 ramesh 45000
5 kishore 35000

Process finished with exit code 0
```

class csv.DictReader(f, fieldnames=None)

Create an object that operates like a regular reader but maps the information in each row to a dict whose keys are given by the optional fieldnames parameter.

Example:

```
import csv
def main():
    f=open("f:\\employee.csv","r")
    dr=csv.DictReader(f)
    tot=0
    for row in dr:
        print(row['Empno'],row['Ename'],row['Salary'])
        tot=tot+float(row['Salary'])
    print(f'Total Salary {tot}')
    f.close()
main()
```

Output:

```
1 naresh 50000
2 suresh 60000
3 rajesh 70000
4 ramesh 45000
5 kishore 35000
Total Salary 260000.0
```

writer