CHAPTER- 3 FILE HANDLING

Learning Outcomes: Learn file handling in Python.

INTRODUCTION:

File:- A file is a collection of related data stored in computer storage for future data retrieval.

Stream: - It refers to a sequence of bytes.

File handling refres to reading and writing contents from a file.

Data Files:

Data files can be stored in two ways:

1. **Text Files**: Text files are structured as a sequence of lines, where each line includes a sequence of characters.
2. **Binary Files** : A binary file is any type of file that is not a text file.

S. No.	Text Files	Binary Files
1.	Stores information in ASCII characters.	Stores information in the same format which the information is held in memory.
2.	Each line of text is terminated with a special character known as EOL (End of Line)	No delimiters are used for a line.
3.	Some internal translations take place when this EOL character is read or written.	No translation occurs in binary files.
4.	Slower than binary files.	Binary files are faster and easier for a program to read and write the text files.

Opening and closing a file:

Opening a file: To work with a file, first of all you have to open the file. To open a file in python, we use open( ) function.

The **open( )** function takes two parameters; *filename*, and *mode*. **open( )** function returns a file object.

Syntax: file_objectname= open(filename, mode)

Example:

To open a file for reading it is enough to specify the name of the file: f = open("book.txt")

The code above is the same as:

f = open("book.txt", "rt")

Where "r" for read mode, and "t" for text are the default values, you do not need to specify them.

Closing a file:

After performing the operations, the file has to be closed. For this, a close( ) function is used to close a file.

Syntax:

file-objectname.close( )

File Modes:

Text file mode	**Binary File Mode**	**Description**
'r'	'rb'	Read - Default value. Opens a file for reading, error if the file does not exist.
'w'	'wb'	Write - Opens a file for writing, creates the file if it does not exist
'a'	'ab'	Append - Opens a file for appending, creates the file if it does not exist
'r+'	'rb+'	Read and Write-File must exist, otherwise error is raised.
'w+'	'wb+'	Read and Write-File is created if does not exist.
'a+'	'ab+'	Read and write-Append new data
'x'	'xb'	Create - Creates the specified file, returns an error if the file exists

In addition you can specify if the file should be handled as binary or text mode

"t" – Text-Default value. Text mode

"b" – Binary- Binary Mode (e.g. images)

WORKING WITH TEXT FILES:

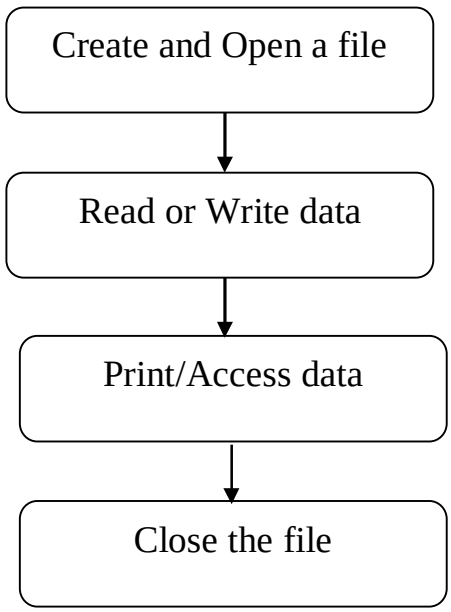
Basic operations with files:

a. **Read** the data from a file

b. **Write** the data to a file

c. **Append** the data to a file

d. **Delete** a file



a. Read the data from a file:

There are **3 types** of functions to read data from a file.

- **read( ) :** reads n bytes. if no n is specified, reads the entire file.
- **readline( ) :** Reads a line. if n is specified, reads n bytes.
- **readlines( ):** Reads all lines and returns a list.

Steps to read data from a file:

- Create and Open a file using open( ) function
- use the read( ), readline( ) or readlines( ) function
- print/access the data
- Close the file.

Let a text file **"Book.txt"** has the following text:

"Python is interactive language. It is case sensitive language.
It makes the difference between uppercase and lowercase letters. It is official language of google."

Example-1: Program using read( ) function:

fin=open("D:\\python programs\\Book.txt",'r') str=fin.read( ) print(str) fin.close( )	fin=open("D:\\python programs\\Book.txt",'r') str=fin.read(10) print(str) fin.close( )
OUTPUT: Python is interactive language. It is case sensitive language. It makes the difference between uppercase and lowercase letters. It is official language of google.	**OUTPUT:** Python is

Example-2: using readline( ) function:

```
fin=open("D:\\python programs\\Book.txt",'r')
 str=fin.readline( )
print(str)
fin.close( )
```

OUTPUT:

Python is interactive language. It is case sensitive language.

Example-3: using readlines( ) function:

```
fin=open("D:\\python programs\\Book.txt",'r')
str=fin.readlines( )
print(str)
fin.close( )
```

OUTPUT:

['Python is interactive language. It is case sensitive language.\n', 'It makes the difference between uppercase and lowercase letters.\n', 'It is official language of google.'**]**

Some important programs related to read data from text files:

Program-a: Count the number of characters from a file. (Don't count white spaces)

```
fin=open("Book.txt",'r')
str=fin.read()
L=str.split()
count_char=0
for i in L:
    count_char=count_char+len(i)
print(count_char)
fin.close( )
```

Program-b: Count the number of words in a file.

```
fin=open("Book.txt",'r')
str=fin.read()
L=str.split()
count_words=0
for i in L:
    count_words=count_words+1
print(count_words)
 fin.close( )
```

Program-c: Count number of lines in a text file.

```
fin=open("Book.txt",'r')
str=fin.readlines()
count_line=0
for i in str:
    count_line=count_line+1
print(count_line)
fin.close( )
```

Program-d: Count number of vowels in a text file.

```
fin=open("D:\\python programs\\Book.txt",'r')
str=fin.read()
count=0
 for i in str:
            if i=='a' or i=='e' or i=='i' or i=='o' or i=='u':
            count=count+1
print(count)
 fin.close( )
```

Program-e : Count the number of 'is' word in a text file.

```
fin=open("D:\\python programs\\Book.txt",'r')
str=fin.read()
L=str.split()
count=0
for i in L:
```

```
            if i=='is':
      count=count+1
print(count)
fin.close( )
```

b. Write data to a file:

There are **2 types** of functions to write the data to a file.

- **write( ):**Write the data to a file.

Syntax:write(string) (for text files) write(byte_string) (for binary files)

- **writelines( )**: Write all strings in a list L as lines to file.

To write the data to an existing file, you have to use the following mode:

"w" - Write - will overwrite any existing content

Example: Open the file "Book.txt" and append content to the file:

```
fout= open("Book.txt", "a")

fout.write("Welcome to the world of Programmers")
```

Example: Open the file "Book.txt" and overwrite the content:

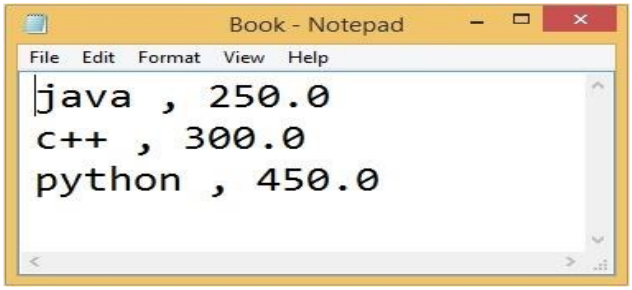
```
fout = open("Book.txt", "w")
fout.write("It is creative and innovative")
```

Program: Write a program to take the details of book from the user and write the record in text file.

```
fout=open("D:\\python programs\\Book.txt",'w')
n=int(input("How many records you want to write in a file ? :"))
for i in range(n):
        print("Enter details of record :", i+1)
        title=input("Enter the title of book : ")
        price=float(input("Enter price of the book: "))
        record=title+" , "+str(price)+'\n'
        fout.write(record)
fout.close( )
```

OUTPUT:

```
How many records you want to write in a file ? :3
Enter details of record : 1
Enter the title of book : java
Enter price of the book: 250
Enter details of record : 2
Enter the title of book : c++
Enter price of the book: 300
Enter details of record : 3
Enter the title of book : python
Enter price of the book: 450
```



c. Append the data to a file:

This operation is used to add the data in the end of the file. It doesn't overwrite the existing data in a file. To write the data in the end of the file, you have to use the following mode:

"a" - Append - will append to the end of the file.

Program: Write a program to take the details of book from the user and write the record in the end of the text file.

```
fout=open("D:\\python programs\\Book.txt",'a')
n=int(input("How many records you want to write in a file ? :"))
 for i in range(n):
        print("Enter details of record :", i+1)
        title=input("Enter the title of book : ")
        price=float(input("Enter price of the book: "))
        record=title+" , "+str(price)+'\n'
        fout.write(record)
```

OUTPUT:

How many records you want to write in a file ? :2

Enter details of record : 1

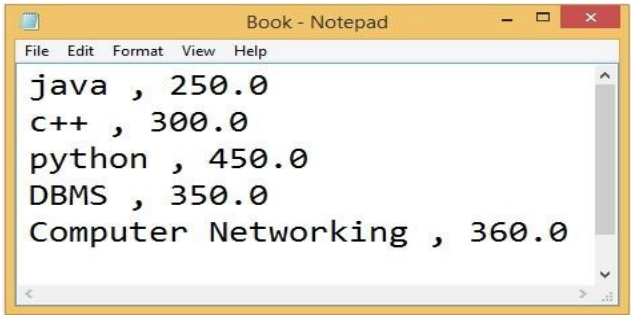
Enter the title of book : DBMS

Enter price of the book: 350

Enter details of record : 2

Enter the title of book : Computer Networking

Enter price of the book: 360



WORKING WITH BINARY FILES:

Binary files are used to store binary data such as images,video files, audio files etc. They store data in the binary format (0's and 1's) .In Binary files there is no delimiter for a line. To open files in binary mode, when specifying a mode, add 'b' to it. Pickle module can be imported to write or read data in a binary file.

(a) Write data to a Binary File:

Example:

```
import pickle
e={'Namita':25000,'Manya':30000,'Tanu':20000}
f1=open('emp.dat','wb')
pickle.dump(e,f1)
f1.close()
```

Output:

A file named emp.dat will be created in current working directory.

(b) Read the data from Binary File:

Example:

```
import pickle
```

```
f1=open('emp.dat','rb')
e=pickle.load(f1)
for x in e:
   print(x)
f1.close()
```

Output:

```
Namita
Manya
Tanu
```

Example :

```
import pickle
f1=open('emp.dat','rb')
e=pickle.load(f1)
for x in e:
   if(e[x]>=25000 and e[x]<=30000):
      print(x)
f1.close()
```

Output:

```
Employees having salary between 25000 to 30000
Namita
Manya
```

tell( ) and seek( ) methods:

- **tell( ):** It returns the current position of cursor in file.

Example:

```
fout=open("story.txt","w")
fout.write("Welcome Python")
 print(fout.tell( ))
fout.close( )
```

Output:

14

- **seek(offset) :** Change the cursor position by bytes as specified by the offset.

Example:

```
fout=open("story.txt","w")
fout.write("Welcome Python")
fout.seek(5)
 print(fout.tell( ))
 fout.close( )
```

Output:

5

File I/O Attributes

Attribute	Description
name	Returns the name of the file (Including path)
mode	Returns mode of the file. (r or w etc.)
encoding	Returns the encoding format of the file
closed	Returns True if the file closed else returns False

Example:

```
f = open("D:\\story.txt", "r")
print("Name of the File:  ", f.name)
 print("File-Mode :  ", f.mode)
print("File encoding format : ", f.encoding)
print("Is File closed? ", f.closed)
f.close()
print("Is File closed? ", f.closed)
```

OUTPUT:

```
Name of the File:   D:\story.txt File-Mode :   r
File encoding format :  cp1252
Is File closed?  False
Is File closed?  True
```

Relative and Absolute Paths :

- We all know that the files are kept in directory which are also known as folders.
- Every running program has a current directory. Which is generally a default directory and python always see the default directory first.
- **The absolute paths are from the topmost level of the directory structure. The relative paths are relative to the current working directory denoted as a dot(.) while its parent directory is denoted with two dots(..).**
- OS module provides many such functions which can be used to work with files and directories. OS means Operating System.
- getcwd( ) is a very function which can be used to identify the current working directory

```
>>> import os
>>> cwd=os.getcwd()
>>> print(cwd)
C:\Users\kv2kkdSrSec\AppData\Local\Programs\Python\Python36-32
```

STANDARD FILE STREAMS :

• We use standard I/O Streams to get better performance from different I/O devices.

• Some Standard Streams in python are as follows –

– Standard input Stream sys.stdin

– Standard output Stream sys.stdout

– Standard error Stream sys.stderr

```
import sys
f=open(r"hello.txt")
line1=f.readline()
line2=f.readline()
line3=f.readline()
sys.stdout.write(line1)
sys.stdout.write(line2)
sys.stdout.write(line3)
sys.stderr.write("\nNo errors occured\n")
f.close()
```

OUTPUT:

Comp Science
Informatics Practices
Kendriya Vidyalaya AFS Salua
No errors occured

Very Short Answer Type Questions (1-Mark)

Q.1 What is a data file in python?

Ans: A bunch of bytes / data stores on some storage device referred by the filename.

Q2. What is the difference between "w" and "a" modes?
Ans: "w" mode opens file in write mode but "a" mode opens file in append mode.

Short Answer Type Questions (2-Mark)

Q.1. Differentiate between a text file and a binary file.

Ans: A text file stores data as ASCII/UNICODE characters where as a binary file stores data in binary format (as it is stored in memory). Internal conversion is required in text file and hence slower but binary file does not need any translation and faster.

Q2. **Differentiate between Absolute path and relative path.**

The absolute paths are from the topmost level of the directory structure. The relative paths are relative to the current working directory denoted as a dot(.) while its parent directory is denoted with two dots(..).

Q3. **What does stdin, stdout represent?**
Ans : *stdin* represent standard input device and *stdout* represent standard output device which are represented as files.

Application Based Questions (3 Marks)

Q1. **Write a python code to find the size of the file in bytes, number of lines and number of words.**

```
# reading data from a file and find size, lines, words
f=open('Lines.txt','r')
str=f.read( )
size=len(str)
print('size of file n bytes',size)
f.seek(0)
L=f.readlines( )
word=L.split( )
print('Number of lines ',len(L))
print('Number of words ',len(word))
f.close( )
```

Q2. **Write code to print just the last line of a text file "data.txt".**

Ans:
```
fin=open("data.txt","r")
lineList=fin.readlines()
print("Last line = ", lineList[-1])
```