Practical Learning #5 - 2
File Handling

Text Files

- Until now, you have been reading and writing to the standard input and output. Now, we will see how to use actual data files.
- Python provides basic functions and methods necessary to manipulate files by default. You can do most of the file manipulation using a file object.
- File handling is an important part of any application.
- Python has several functions for creating, reading, updating, and deleting files

File Handling

The key function for working with files in Python is the `open()` function.

The `open()` function takes two parameters; *filename*, and *mode*.

There are four different methods (modes) for opening a file:

- `"r"` - Read - Default value. Opens a file for reading, error if the file does not exist
- `"a"` - Append - Opens a file for appending, creates the file if it does not exist
- `"w"` - Write - Opens a file for writing, creates the file if it does not exist
- `"x"` - 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)

9. Assume we have the following file, located in your demo folder: Filename: demofile.txt

```
Hello! Welcome to demofile.txt
This file is for testing purposes.
Good Luck!
```

10. Create a new python file name as PL5-2_LastnameFirstname.Type the following

```
#To open the file, use the built-in open() function.
#The open() function returns a file object, which has a
#read() method for reading the content of the file

f = open("demofile.txt","r")
print(f.read())
```

OUTPUT:

```
Hello! Welcome to demofile.txt
This is for testing purposes.
Good Luck!
```

11.

```
#Read Only Parts of the file
#By default the read() method returns the whole text,
#but you can also specify how many character you wan to return
f = open("demofile.txt","r")
print(f.read(5))
```

OUTPUT:

```
Hello
```

```
#Read Lines
#You can return one line by using the readline() method
f = open("demofile.txt","r")
print(f.readline())
```

OUTPUT:

```
Hello! Welcome to demofile.txt
```

```
#by calling readline() two times, you can read
#the first two lines

f=open("demofile.txt","r")
print(f.readline())
print(f.readline())
```

OUTPUT:

```
Hello! Welcome to demofile.txt

This is for testing purposes.
```

14.

```
#by looping through the lines of the file,
#you can read the whole file, line by line:

f=open("demofile.txt","r")
for x in f:
    print(x)
```

OUTPUT:

```
Hello! Welcome to demofile.txt

This is for testing purposes.

Good Luck!
```

Write to an Existing File

To write to an existing file, you must add a parameter to the `open()` function:

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

`"w"` - Write - will overwrite any existing content

15. Comment previous codes.

```
f = open("demofile.txt","a")
f.write("\nNow the file has one more line!")
f.close()
f = open("demofile.txt","r")
print(f.read())
```

OUTPUT:

```
Hello! Welcome to demofile.txt
This file is for testing purposes.
Good Luck!
Now the file has one more line!
```

Every time you run the program this line will be added in the demofile.

16. Comment previous codes:

```
f = open("demofile.txt","w")
f.write("Woops!Ihave deleted the content!")
f.close()
f = open("demofile.txt","r")
print(f.read())
```

OUTPUT:

```
Woops!Ihave deleted the content!
```

The close() method of a file object flushes any unwritten information and closes the file object, after which no more writing can be done.

Python automatically closes a file when the reference object of a file is reassigned to another file. It is a good practice to use the close() method to close a file.

17. Save and upload your work.