JUNAID GIRKAR
60004190057
TE COMPS A4

EXPERIMENT - 6

Aim :

Demonstrate File handling and Directories
a. Python program to append data to existing file and then display the entire file. b. Python program to count number of lines, words and characters in a file. c. Python program to display file available in current directory

Theory

A. Python program to append data to existing file and then display the entire file. While reading or writing to a file, access mode governs the type of operations possible in the opened file. It refers to how the file will be used once it's opened. These modes also define the location of the File Handle in the file. File handle is like a cursor, which defines from where the data has to be read or written in the file.
   a. Append Only ('a'): Open the file for writing. The file is created if it does not exist. The handle is positioned at the end of the file. The data being written will be inserted at the end, after the existing data.
   b. Append and Read ('a+'): Open the file for reading and writing. The file is created if it does not exist. The handle is positioned at the end of the file. The data being written will be inserted at the end, after the existing data.

   When the file is opened in append mode, the handle is positioned at the end of the file. The data being written will be inserted at the end, after the existing data.

B. Python program to count number of lines, words and characters in a file. Counting the number of characters is important because almost all the text boxes that rely on user input, have a certain limit on the number of characters that can be inserted. For example, the character limit on a Facebook post is 63, 206 characters. Whereas, for a tweet on Twitter the character limit is 140 characters and the character limit is 80 per post for Snapchat. Counting the lines, word, and characters within a text file results in the line count, word count, and character count, which includes spaces. For instance the text file containing "Hello World\nHello Again\nGoodbye" results on lines: 3 words: 5 characters: 29. Call open(file, mode) with the pathname of a file as file and mode as "r" to open the file for reading. Use a for-loop to iterate through the file.

   At each iteration, use str.strip(characters) with "\n" as characters to strip it from each line str, and use str.split() to create a list containing all the words from the line str. Also, in each iteration, add 1 to the number of lines, use len(object) with the list containing the word from the line as object to add it to the number of words, and use len(object) with the stripped line as object to add it to the number of characters.

C. Python program to display file available in current directory

The following is a list of some of the important methods/functions in Python with descriptions required to implement the program to list the directories and files in a given folder.

- **str()** – It is used to transform data value(integers, floats, list) into string.
- **abspath()** – It returns the absolute path of the file/directory name passed as an argument.
- **enumerate**() – Returns an enumerate object for the passed iterable that can be used to iterate over the items of iterable with an access to their indexes.
- **list()** – It is used to create a list by using an existing iterable(list, tuple, dictionary, set).
- **listdir()** – It is used to list the directory contents. The path of directory is passed as an argument.
- **isfile()** – It checks whether the passed parameter denotes the path to a file. If yes then returns True otherwise False
- **isdir()** – It checks whether the passed parameter denotes the path to a directory. If yes then returns True otherwise False
- **append()** – It is used to append items on a list.

Code

a. Python program to append data to existing file and then display the entire file.

```
def AppendData(fname):
    """Python program to append data to existing file and then
display  the entire file. """
    """Before appending to "file.txt" its contents of the file were
as  follows:"""
    beforeAppending = open(fname)
    print("Before appending to 'file.txt' its contents were as
follows:")
    print(beforeAppending.read())
    beforeAppending.close()
    with open(fname, 'a', encoding='utf-8') as f:
        f.write("\nThis is the First Line.\n")
        f.write("This is the Second one\n")
        f.write("This is the final line.\n")
        print("*"*80)
    """Before appending to "file.txt" its contents were as
follows:"""
    afterAppending = open(fname)
    print("After appending to file.txt' its contents were as
follows:")
    print(afterAppending.read())
    afterAppending.close()
def main():
```

```
    AppendData("file.txt")
if __name__ == '__main__':
    main()


OUTPUT:
Before app
```

OUTPUT:

```
Before appending to 'file.txt' its contents were as  follows:

**********************************************************************
*************
After appending to file.txt' its contents were as  follows:

This is the First Line.
This is the Second one
This is the final line.
```

b. Python program to count number of lines, words and characters in a file.

```
def CountInFile(fname):
    """Python program to count number of lines, words and
characters in  a file."""
    with open(fname, 'r', encoding='utf-8') as f:
        lineCount = 0
        wordCount = 0
        characterCount = 0
        for line in f:
            line = line.strip("\n")

            words = line.split()
            lineCount += 1
            wordCount += len(words)
            characterCount += len(line)
        print(f"In the file {fname} we have:")
        print(f"Lines: {lineCount}")
        print(f"Words: {wordCount}")
```

```
        print(f"Characters: {characterCount}")
def main():
    CountInFile("file.txt")

main()
```

OUTPUT:

```
In the file file.txt we have:
Lines: 4
Words: 15
Characters: 68
```

---

c. Python program to display files available in the current directory.

```
import os

def countDirectories(directoryList):
    """ A function that lists the directories in a folder."""
    print("Directories in the given folder: ")
    for index, dir in enumerate(directoryList):
        print(f"{str(index+1)}. {dir}")

def countFiles(fileList):
    """ A function that lists the files in a folder."""
    print("Files in the given folder: ")
    for index, file in enumerate(fileList):
        print(f"{str(index+1)}. {file}")

def displayFilesAvailableInCurrentDirectory(path="."):
    """
    A function that calls 2 functions to separately listing out
directories and files.
    It takes a default argument as cwd(.). We can
    pass other paths too.
    """
    fileList = list()
    directoryList = list()
```

```
        try:
            for f in os.listdir(path):
                if os.path.isfile(os.path.join(path, f)):
                    fileList.append(f)
                else:
                    if os.path.isdir(os.path.join(path, f)):
                        directoryList.append(f)
        except:
            print("\nError, please check the path")
            pass
        print(f"Given folder: {os.path.abspath(path)}")
        countFiles(fileList)
        countDirectories(directoryList)

    def main():
        displayFilesAvailableInCurrentDirectory()

    main(
```

OUTPUT:

```
Files in the given folder:
1. appendData.py
2. CountDIrectories.py
3. countlines.py
4. file.txt
Directories in the given folder:
1. RandomFolder
2. RandomFolder2
```

Conclusion

In part a of this experiment, we learned how to append data to a pre-existing file in Python using the "with open file in 'a' mode" syntax, where 'a' stands for "append". In part b, we learned how to count the number of lines, words and characters in a file in Python by iterating over the file line by line and counting them. In part c, we listed all the files and directories (separately) in a given folder by calling the abspath function to display the absolute path of the chosen directory, the listdir function to list the files and folders in the chosen directory from the in-built os module. We also checked if a particular item was a directory or a file by calling the isdir and isfile functions and then displaying the files and folders separately.