Experiment No. 5
Exploring Files and directories: Python program to append data to existing file and then display the entire file
Date of Performance: 12/04/2024
Date of Submission: 26/04/2024

Experiment No. 5

Title: Exploring Files and directories: Python program to append data to existing file and then display the entire file

Aim: To Exploring Files and directories: Python program to append data to existing file and then display the entire file

Objective: To Exploring Files and directories **Theory:**

Directory also sometimes known as a folder are unit organizational structure in computer's file system for storing and locating files or more folders. Python now supports a number of APIs to list the directory contents. For instance, we can use the Path.iterdir, os.scandir, os.walk, Path.rglob, or os.listdir functions.

Python too supports file handling and allows users to handle files i.e., to read and write files, along with many other file handling options, to operate on files. The concept of file handling has stretched over various other languages, but the implementation is either complicated or lengthy, but alike other concepts of Python, this concept here is also easy and short. Python treats file differently as text or binary and this is important. Each line of code includes a sequence of characters and they form text file. Each line of a file is terminated with a special character, called the EOL or End of Line characters like comma {,} or newline character. It ends the current line and tells the interpreter a new one has begun. Let's start with Reading and Writing files.

Working of open() function

We use open () function in Python to open a file in read or write mode. As explained above, open ( ) will return a file object. To return a file object we use open() function along with two arguments, that accepts file name and the mode, whether to read or write. So, the syntax being: open(filename, mode). There are three kinds of mode, that Python provides and how files can be opened:

" r ", for reading.

" w ", for writing.

" a ", for appending. " r+ ", for

both reading and writing **Code:**

```
#file read f = open("demofile.txt",

"r") str = f.read() print(str) #file

append f = open("demofile.txt",

"a")

f.write("Now the file has more content!")

f.close() f =

open("demofile.txt", "r")

print(f.read()) #file write f =

open("demofile.txt", "w")

f.write("Woops! I have deleted the content!")
```

f.close() f = open("demofile.txt", "r")

print(f.read()) def

count_words_lines_chars(file_name):

word_count = 0

line_count = 0 char_count = 0

with open(file_name, 'r') as

file:

for line in file:

words = line.split() word_count += len(words) line_count += 1

char_count += sum(len(word) for word in words) return word_count,

line_count, char_count file_name = 'demofile.txt' word_count, line_count,

char_count = count_words_lines_chars(file_name) print("Number of

words:", word_count) print("Number of lines:", line_count) print("Number

of characters:", char_count) **Output:**

File read,write and append

```
C:\Users\Student\PycharmProjects\pythonProject2\venv\Scripts\python.exe C:\Users\Student\PycharmProjects\pythonProject2\main.py
hello vcet!!!
this is the vcet lab
Woops! I have deleted the content!
hello vcet!!!
this is the vcet lab
Woops! I have deleted the content!Now the file has more content!
Woops! I have deleted the content!

Process finished with exit code 0
```

Count the lines,words and characters of string

```
C:\Users\Student\PycharmProjects\pythonProject2\venv\Scripts\python.exe C:\Users\Student\PycharmProjects\pythonProject2\main.py
Number of words: 6
Number of lines: 1
Number of characters: 29

Process finished with exit code 0
```

Conclusion:

Appending Data to File: The append_to_file function takes the file name and data to append as arguments.It opens the file in append mode ('a') and writes the provided data to the end of the file.If successful, it prints a success message; otherwise, it prints an error message.

Displaying File Contents: The display_file_contents function takes the file name as an argument.It opens the file in read mode ('r') and prints its contents.If the file is not found or an error occurs, appropriate error messages are displayed.

This program illustrates how to append data to an existing file and then display its contents using Python's file handling capabilities. Understanding file operations in Python is crucial for tasks involving reading from and writing to files, which are common in various applications.