



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

Date of Performance:

Date of Submission:

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



Vidyavardhini's College of Engineering & Technology

Department of Computer Engineering

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

```
File Edit Selection View Go Run ... Search
exp1.py exp2.py exp3.py exp4.py exp5.py x exp6.py exp7.py
C:\Users\Student\Documents> exp5.py > ...
1 def append_data_to_file(file_name, data):
2
3     with open(file_name, 'a') as file:
4         file.write(data)
5         print("Data appended successfully to", file_name)
6
7
8 def display_file_content(file_name):
9
10    with open(file_name, 'r') as file:
11        content = file.read()
12        print("Content of", file_name, ":\n", content)
13
14
15
16 file_name = "example.txt"
17
18
19 append_data_to_file(file_name, "This is some additional data.\n")
20
21
22 display_file_content(file_name)
23
```

Ln 23, Col 1 Spaces: 4 UTF-8 CRLF Python 3.11.2 64-bit GO LIVE PREVIEW

26°C Smoke

Sharing to Desktop-123
Screenshot (15).png
Successfully sent

09:29 13-02-2024



Vidyavardhini's College of Engineering & Technology

Department of Computer Engineering

```
File Edit Selection View Go Run ... Search
exp1.py exp2.py exp3.py exp4.py exp5.py x exp6.py exp7.py
C:\Users\Student\Documents> exp5.py
1 def append_data_to_file(file_name, data):
2
3     with open(file_name, 'a') as file:
4         file.write(data)
5         print("Data appended successfully to", file_name)
6
7
8 def display_file_content(file_name):
9
10    with open(file_name, 'r') as file:
11        content = file.read()
12        print("Content of", file_name, ":\n", content)
13
14
15
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS Python
PS C:\Users\Student> & C:\Users\Student\AppData\Local\Programs\Python\Python311\python.exe c:/Users/Student/Documents/exp5.py
Data appended successfully to example.txt
Content of example.txt :
This is some additional data.
This is some additional data.
This is some additional data.
This is some additional data.
This is some additional data.
This is some additional data.
PS C:\Users\Student>
```

Conclusion: Directories and files have been explored.

A directory is a collection of files and subdirectories. A directory inside a directory is known as a subdirectory. Python has the os module that provides us with many useful methods to work with directories (and files as well).