|  |
| --- |
| Experiment No. 5 |
| Exploring Files and directories: Python program to append data to existing file and then display the entire file |
| Date of Performance:19/2/2024 |
| Date of Submission:26/2/2024 |

**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:**

**A)**

# Writing to a file with open('filenew.txt', 'w') as f: user\_input = input("Enter text: ")

f.write(user\_input) # Reading from the file

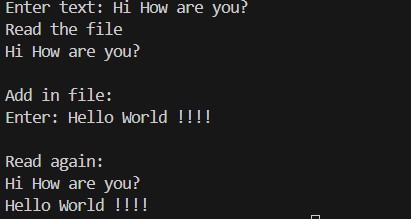
print('Read the file') with open('filenew.txt', 'r') as f:

file\_content = f.read() print(file\_content) # Appending to the file

print('\nAdd in file:') with open('filenew.txt', 'a') as f: user\_input = input('Enter: ')

f.write("\n" + user\_input) # Reading from the file again print("\nRead again:") with open('filenew.txt', 'r') as f: file\_content = f.read() print(file\_content)

**Output A):**



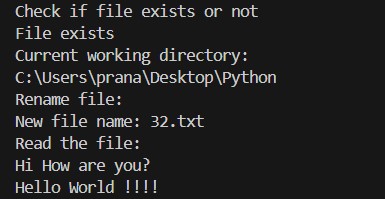
**B)** import os print("Check if file exists or not") if os.path.exists("filenew.txt"):

print("File exists") else:

print("File doesn't exist") print("Current working directory:") dirnow = os.getcwd()

print(dirnow) dir\_path = r"C:\Users\prana\Desktop\Python" os.makedirs(dir\_path, exist\_ok=True) print("Rename file:") old\_file\_name = 'filenew.txt' new\_file\_name = '32.txt' os.rename(old\_file\_name, new\_file\_name) print(f"New file name: {new\_file\_name}") print("Read the file:") with open('32.txt', 'r') as f: file\_content = f.read() print(file\_content)

**Output B):**



**C)** import re print("search method") str1 = "cat mat mop mit"

result1 = re.search(r'm\w\w', str1) print(result1.group()) print("\nfindall method") str2 = "one two five seven three 4 5 9" print("\nWord with length 4") result2 = re.findall(r'\b\w{4}\b' , str2) print(result2)

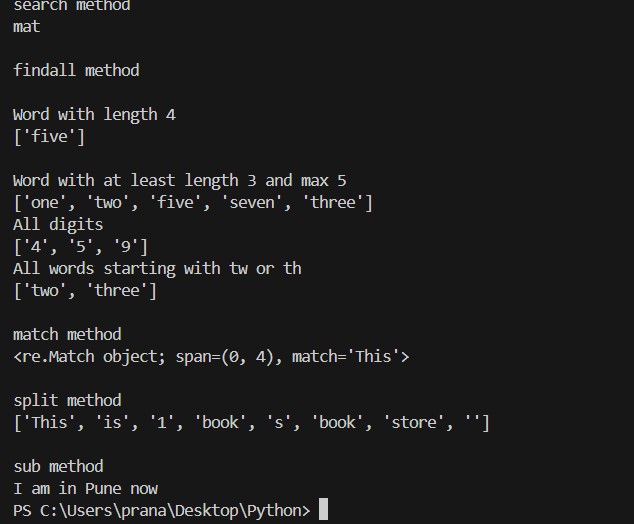
print("\nWord with at least length 3 and max 5") result3 = re.findall(r'\b\w{3,5}\b' , str2) print(result3) print("All digits") result4 = re.findall(r'\b\d\b',str2) print(result4) print("All words starting with tw or th") result5 = re.findall(r't[wh][\w]\*' , str2) print(result5) print("\nmatch method") str3 = ("This is new!") result6= re.match(r'T\w\w\w' , str3) print(result6) print("\nsplit method")

str4 = ("This is 1$ book\s @ book store?")

result7 = re.split(r'\W+' , str4) print(result7) print("\nsub method") str5 = ("I am in Mumbai now")

result8 = re.sub(r'Mumbai','Pune' , str5) print(result8))

**Output C):**



# Department of Computer Engineering

**Conclusion:** This Python code exhibits a comprehensive utilization of both the `os` module for file handling operations and regular expressions (regex) for text manipulation. Through functions such as checking file existence, creating directories, renaming files, and displaying renamed file names, it underscores the versatility of Python in managing files and directories efficiently. Furthermore, by employing regex for pattern matching and manipulation tasks, the code showcases Python's prowess in handling complex textual data processing operations, thereby demonstrating its adaptability for diverse tasks encompassing data manipulation, text processing, and file management with ease and precision.