

TASK-67

Implement various text file operation

Aim: To write a Python Program implement various text file operations.

Algorithm:

1. Write to a file:

- Define writefile (filename) function

- open a file named log.txt in write mode

- write the following text to the file

- Error objects are known when runtime error occurs.

- close the file.

2. Read from a file:

- Define readfile (filename) function.

- open the files specified by filename in read using a with statement.

- Read the entire content of the file.

- Print the content.

3. Execute the Program

- call writefile ("write") to write the predefined text to "log.txt".

- call readfile (text) to attempt to read from a file named "text" and print its content.

Implement various text file operations

Implement various text file operations

Implement

Output

Error objects are thrown when runtime errors occur. The Error object can also be used as a base object for user-defined exceptions.

Error objects are thrown when runtime errors occur.

Error occurs.

Close the file.

Read from file.

Write to file (FileWriter).

Open the file specified by filename in

mode and a given base

Read the entire content of the file

Write the content

Example: The program

will write the 'write' to write the program

to 'log.txt'

will write the 'write' to write the program

to 'log.txt' and print the

output

Program:

```
def writefile (filename):
```

```
    f=open("log.txt","w")
```

```
    f.write ("Error objects are thrown when  
runtime error occur. The error object  
can also be used as base object  
for user-defined exceptions")
```

```
    f.close()
```

```
def readfile (filename):
```

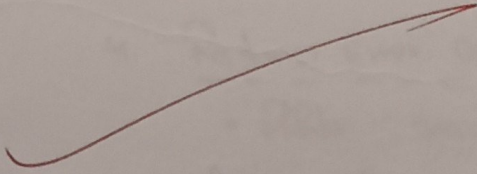
```
    with open (filename,"r") as file:
```

```
        content = file.read()
```

```
        print (content)
```

```
writefile ("write")
```

```
readfile ("text")
```



TASK - Q2 7.2

Aim: To write a Python function that counts the number of lines containing the word 'error' in a log file

Algorithm:

1. Initialize Error counter:

- Define the function count-error-lines
- Initialize error-count to 0.

2. Open and Read file:

- open the file specified by filename in read mode using a with statement

3. check each line for "ERROR":

- loop through line the file
- If line contains the word "ERROR" increment by 1.

4. Return Error count:

- After reading all the lines return the value of error-count.

5. Execute the Program

- call count-error-lines ("log.txt") to count the number with the word "ERROR" in the file "log.txt".

lines to write a given function etc. count
the number of lines containing the word

output in a file

RESTART : c/users/91979/Desktop/5125/69, 100

Number of files with ERROR is 2

- 1. Initialize error count to 0
- 2. Iterate the function count error lines
- 3. Initialize error count to 0
- 4. Iterate the function count error lines
- 5. Iterate the function count error lines
- 6. Iterate the function count error lines
- 7. Iterate the function count error lines
- 8. Iterate the function count error lines
- 9. Iterate the function count error lines
- 10. Iterate the function count error lines

- 1. Iterate the function count error lines
- 2. Iterate the function count error lines
- 3. Iterate the function count error lines
- 4. Iterate the function count error lines
- 5. Iterate the function count error lines
- 6. Iterate the function count error lines
- 7. Iterate the function count error lines
- 8. Iterate the function count error lines
- 9. Iterate the function count error lines
- 10. Iterate the function count error lines

- Print the result with the message: "Number of lines the "ERROR": {error-lines}"

Program:

```
def count_error_lines(filename):
```

```
    error_count = 0
```

```
    with open(filename, "r") as file
```

```
        for line in file
```

```
            if error in line
```

```
                error_count += 1
```

```
    return error_count
```

```
error_lines = count_error_lines("log.txt")
```

```
Print(f"Number of lines with ERROR: {error_lines}").
```

log.txt

"Error objects are thrown when runtime error occur"

The error object can also be used as base object for user defined exceptions.

Task 7.3

Aim: To write a function that counts the no. of lines containing word ERROR.

Algorithm:

1. Create Employee Data.
 - > Define the function write_employee_report
 - > create a list employee containing dictionaries each with "name" and development keys for individual employee.
2. Open file for writing
 - open the file specified by filename in write mode using a with statement.
3. write Employee Data to file
 - loop through each employee in the employee list
 - > for each employee format a string as name
 - > write the formatted string to the file followed by a newline character (\n).
4. Execute the Program
 - call write_employee_report("employee_report.txt") to write the employee data to the file "employee_report.txt".

It is with a function that reads the
input of lines containing word errors

output

Number of lines with ERROR = 2.

Algorithm

- 1. Create empty list
- 2. Iterate the function write - empty - output
- 3. Create a list of empty containing dictionary
- 4. Each with 'name' and 'email' keys for individual employee
- 5. Open file for writing
- 6. Open the file specified by filename in write mode using a with statement
- 7. Write employee data to file
- 8. Loop through each employee in the employee list
- 9. For each employee format a string as name
- 10. Write the formatted string to the file
- 11. Followed by a newline character (\n)
- 12. Execute the program
- 13. Write - empty - output

Program:

```
def write_employee_report(filename):
```

```
    employees = [
```

```
        {"name": "Alice", "department": "HR"},
```

```
        {"name": "Bob", "department": "Engineering"},
```

```
        {"name": "Charlie", "department": "Finance"}]
```

```
]
```

```
    with open(filename, "w") as file:
```

```
        for employee in employees:
```

```
            line = f"Name: {employee['name']} Department:
```

```
            {employee['department']} \n"
```

```
            file.write(line)
```

```
    # Example usage
```

```
    write_employee_report("employees_report.txt")
```

VEL TECH - CSE	
EX NO.	7
PERFORMANCE (5)	5
RESULT AND ANALYSIS (3)	3
VIVA VOCE (3)	3
RECORD (4)	4
TOTAL (15)	15
SIGN WITH DATE	

Result: ✓ Thus the Python Program implemented various text file operations was successfully executed and the output was verified.