

Task 8. Implement various text file operation.

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

Problem 8.1:

You need to write the sentence "Error objects are thrown when runtime errors occur. The error object can also be used as a base object for user-defined exceptions" into a text file named log.txt. Implement a function that performs this task.

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 thrown when runtime errors occur. The error object can also be used as a base object for user-defined exceptions
 - Close the file.

2. Read from a file:

- Define readfile(filename) function:
 - Open the file specified by filename in read mode using a with statement.
 - Read the entire content of the file
 - Print the content

3. Execute the program;

- Call writefile("text") to write the predefined text to "log.txt".
- Call readfile("text") to attempt to read from a file named "text" and print its content.

notifica si la test avviene tramite .E most

Output:

Just consider the following reporting routine is there is no error
Error objects are thrown when runtime errors occurs. The
Error objects can also be used as a base for user-defined
exceptions.

account on a void void "txt" operator with others of basic type
having balanced begin ends with integer errors number nodes
so that "andreas@x3" benefits - user of voids and so on
among them containing a template.txt got become if test
most diff

continu

diff o at stile .L

: without (error) definition info.

show errors in "txt.got" become diff .error

: diff of txt program diff errors

without errors result void void void

so others fields using the library errors

comparing behaviour of voids and so on basic

if it reads.

: diff o many basic .

: notusing (error) without info.

show basic in errorinfo but errors diff with respect

translating others given

diff of behavior with out basic

behavior diff with respect

: (and D019 diff standard)

behaving diff errors of ("error") definition info.

"txt.got" is first

basic basic of template of ("txt") without info .

behavior diff string base "txt" becomes diff o

Program: 8.1

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

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

f.write("Errors Object are thrown when runtime error occur. The Error object can also be used as a 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")
```

Problem 6.2: You have a text file log.txt containing logs of a system. Write a function that counts the number of lines containing the word "ERROR".

Algorithm:

1. Initialize Error Counter:

- Define the function count_error_lines(filename):

- 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 each line in the file:

- If the line contains the word "ERROR", increment error_count by 1.

4. Return Error Count:

- After reading all the lines, return the value of error_count

Output:

Number of time with "ERROR" is 2, file.pot = ?

: (margin) file.pot

: C:\00 ("pot", pot) = ?

Count of words which are error in file.pot

(margin) file.pot

: C:\00

: (margin) file.pot

: with in ("error") file.pot

(margin) file.pot

(total) file.pot

("error") file.pot

count of ("error")

method of doing print words in file.pot

It prints word frequency at other than count of words
with a file "file.pot" in row

: margin

: (margin) file.pot

: (margin) file.pot

o of two rows will print

: (margin) file.pot

print same word in memory of function if it is not

remembered then

"word" of in dict is

: (margin) file.pot

tammanit, "good" break with another with file.pot

: (margin) file.pot

now - now of another function, and it is no problem right.

5. Execute the Program:

- call count_error_lines("log.txt") to count the number of lines with the word "Error", increment error_count by 1.

Program 8.3:

```
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 occurs.

The Error Objects can also be used as a base object for user defined exceptions.

Problem 8.3

You need to write a report containing the details (name, departments) of the employee in list. Write a Python function that writes this report to a file named.

employee-report.txt

Algorithm:

1. Create Employee Data:

- Define the function write_employee_report(filename):
 - Create a list employees containing dictionaries, each with "name" and "department" keys for individual employees.

2. Open file for writing:

- Open the file specified by filename in write mode using a with Statement.

Skewness of column wise sum of ("txt.pdf") and "sum-trunc.htm".
LSD loadings remain, "word" know it.

Output:

Name: Alice, Department: HR

Name: Bob, Department: Engineering

Name: Charlie, Department: Finance

3. Write Employee data to file:

• Loop through each employee in the employee list:

• For each employee, format a string as "Name: {employee['name']}, Department: {employee['department']}".

• 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".

Program 8.3:

```
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']}
```

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

VEL TECH - CSE	
EX NO.	8
PERFORMANCE (5)	S
RESULT AND ANALYSIS (5)	S
VIVA VOCE (5)	S
RECORD (5)	
TOTAL (20)	15
SIGN WITH DATE	

Result: Thus, the Python program implements various textfile operations was successfully executed and the output was verified.