

**ENCS, Computer Systems Engineering**

**ENCS3130, LINUX LABORATORY**

**Project 2 Python**

Sahar Hmidat ID: 1202038

Instructor’s Name: Dr. Mohammad Jubran

Assistant Name: Tareq

Section No: 1

Date: 12/6/2024

# **Introduction**

The goal of this project is to automate Python command execution so that diverse activities may be completed quickly and easily. This will lay the groundwork for automation testing scenarios in a variety of situations. The generated software can be used to execute scripts that have been predefined, confirming the resilience of the tested systems and looking for particular behaviors.

Table of Contents

[**Introduction** II](#_Toc169121087)

[List of figures IV](#_Toc169121088)

[**Program Implementation** V](#_Toc169121089)

[For Delete Cmd: V](#_Toc169121090)

[For List cmd: V](#_Toc169121091)

[For Categories cmd: VI](#_Toc169121092)

[For Count cmd: VII](#_Toc169121093)

[For Move cmd: VII](#_Toc169121094)

[For Rename cmd: VIII](#_Toc169121095)

[For Sort class: IX](#_Toc169121096)

[For main: IX](#_Toc169121097)

[**Running the Program** XIII](#_Toc169121098)

[After running: XIV](#_Toc169121099)

[Csv running: XV](#_Toc169121100)

# List of figures

[Figure 1 Grep class V](#_Toc169121041)

[Figure 2 Delete class V](#_Toc169121042)

[Figure 3 List class VI](#_Toc169121043)

[Figure 4 Category class VII](#_Toc169121044)

[Figure 5 Count class VII](#_Toc169121045)

[Figure 6 move class VIII](#_Toc169121046)

[Figure 7 Rename class IX](#_Toc169121047)

[Figure 8 sort class IX](#_Toc169121048)

[Figure 9 Main 1 X](#_Toc169121049)

[Figure 10 Main 2 X](#_Toc169121050)

[Figure 11 Main 3 X](#_Toc169121051)

[Figure 12 Main 4 XI](#_Toc169121052)

[Figure 13 Main 5 XI](#_Toc169121053)

[Figure 14 Main 6 XI](#_Toc169121054)

[Figure 15 Main 7 XII](#_Toc169121055)

[Figure 16 Main 8 XII](#_Toc169121056)

[Figure 17 Main 9 XII](#_Toc169121057)

[Figure 18 Main 10 XIII](#_Toc169121058)

[Figure 19 teatli folder XIII](#_Toc169121059)

[Figure 20 tealt2 folder XIII](#_Toc169121060)

[Figure 21 Script txt XIV](#_Toc169121061)

[Figure 22 JSON file XIV](#_Toc169121062)

[Figure 23 Run 1 XIV](#_Toc169121063)

[Figure 24 Run 2 XIV](#_Toc169121064)

[Figure 25 move run XV](#_Toc169121065)

[Figure 26 category run XV](#_Toc169121066)

[Figure 27 category run 2 XV](#_Toc169121067)

[Figure 28 rename run XV](#_Toc169121068)

[Figure 29 count run XV](#_Toc169121069)

[Figure 30 CSV run XV](#_Toc169121070)

# **Program Implementation**

We used PyCharm to write python language for this project. First, we wrote each command in class itself and call it to main for each case.

First of all, we wrote grep class to find specific word to run correct command, we used this class in each command after read Script.txt file.

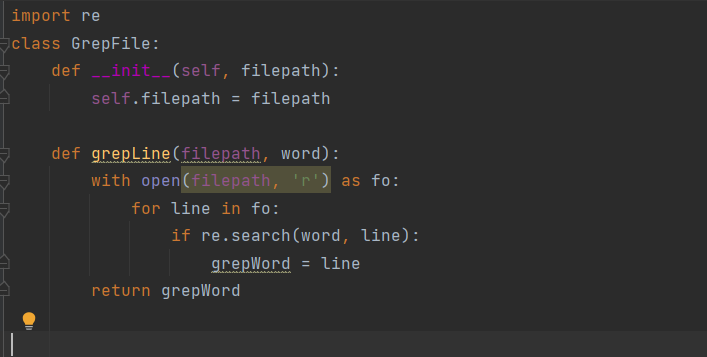


Figure 1 Grep class

Then we made class for each command as shown below.

## For Delete Cmd:

We read the file to find the delete command to know the source folder and destination file , then call grep class to read the line contains delete and split the line to got the source and destination, after that sure that folders is exist then remove the file.

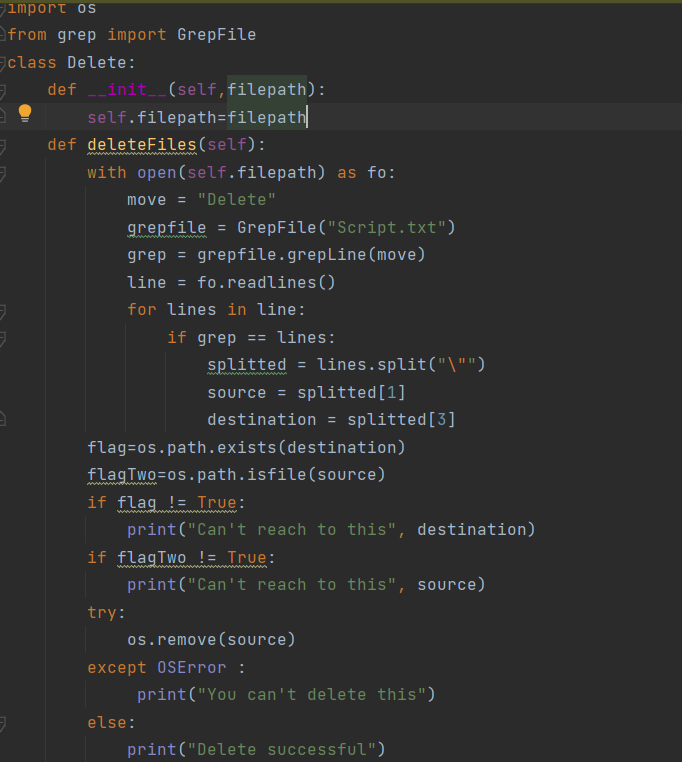


Figure 2 Delete class

## For List cmd:

We wrote this class as delete class, call grep class to search in List command after reading the file then split it to know the path of folder and then list all files in it by use os.listdir.

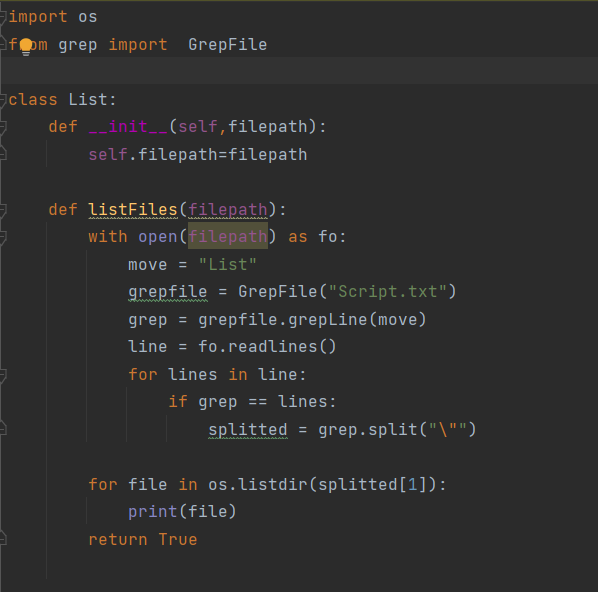
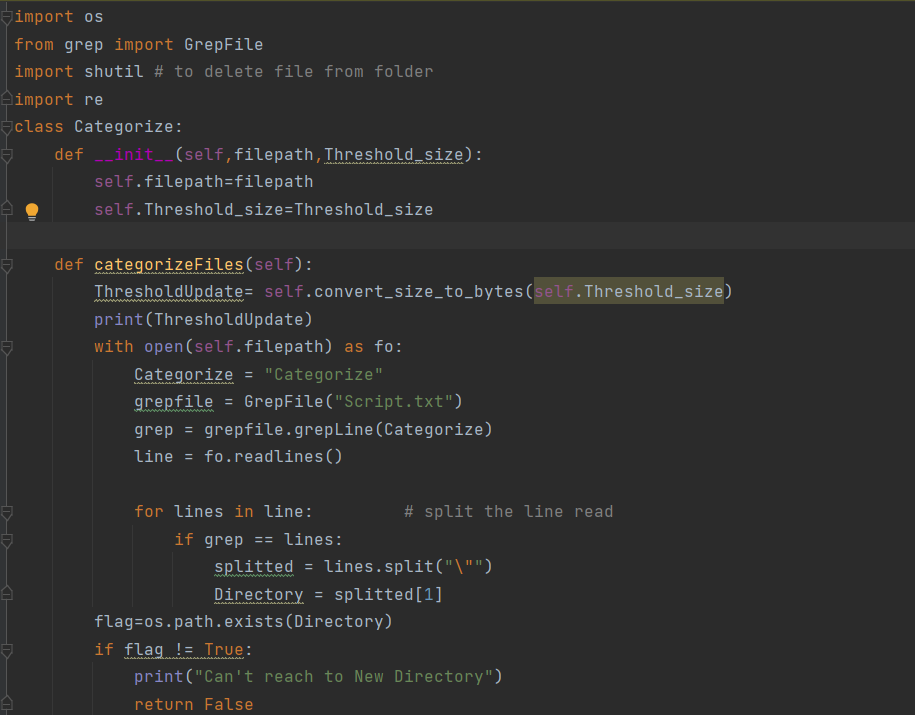
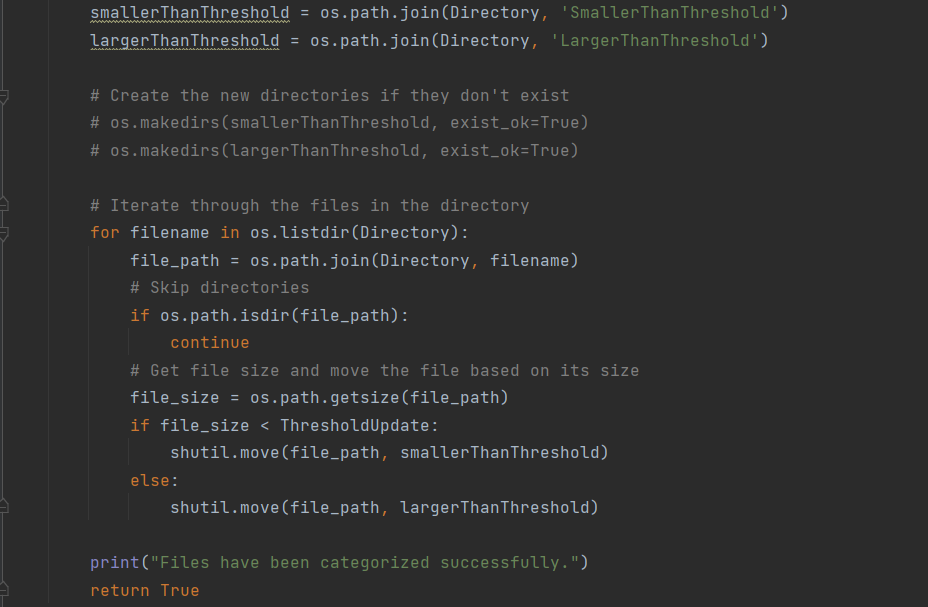


Figure 3 List class

## For Categories cmd:

As before I call grep class to find Categories command and read the folder. We make two folders for smaller and larger and sort it in these based on threshold wrote from json file, and because there are K we use function to convert each K byte to int.





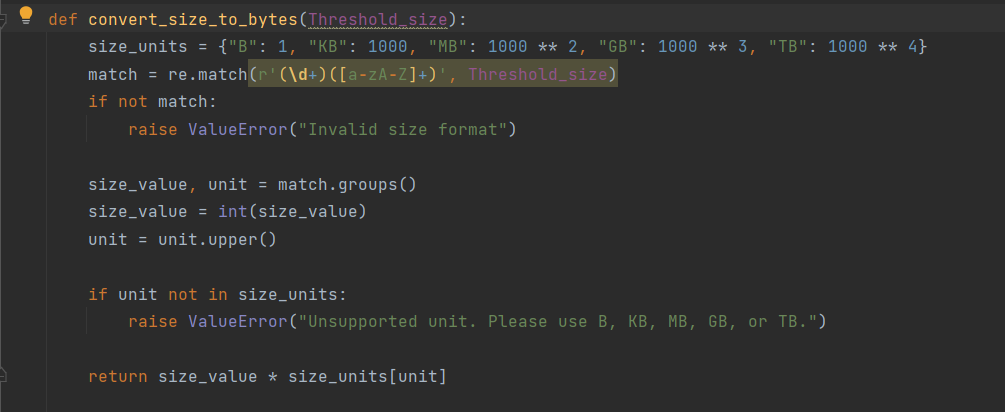


Figure 4 Category class

## For Count cmd:

We wrote count class as previous classes, call grep to find Count word then split the line to find the folder and list all file in this folder and count it, then print in file and in terminal the total count.

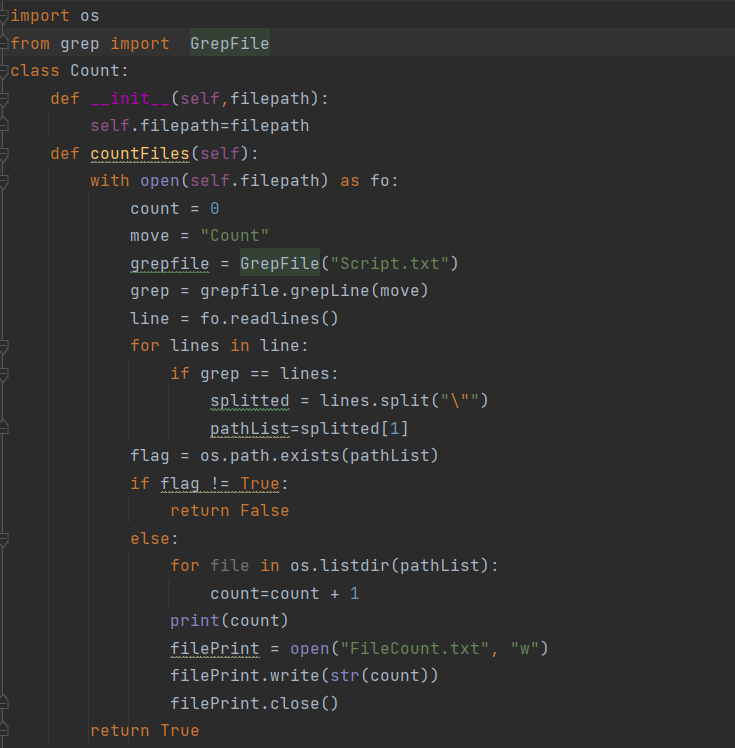
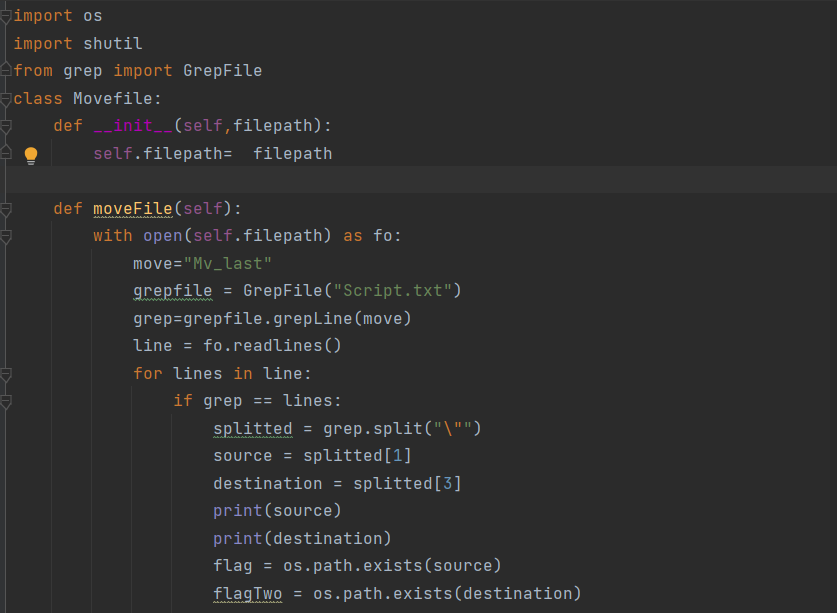


Figure 5 Count class

## For Move cmd:

Also there, we call grep class to find the Mv\_last word from Script.txt file then split to know the source and destination folder then make sure that folders exist and then move the specific file from source to destination and then check if already exist in this destination or not.



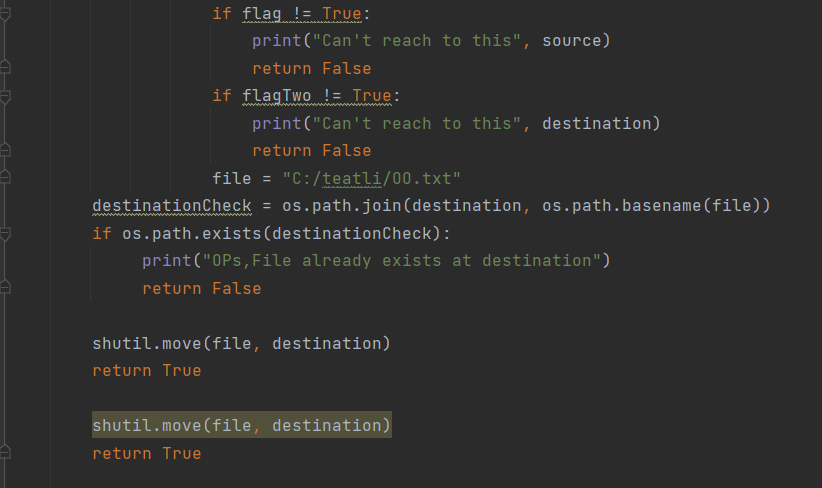
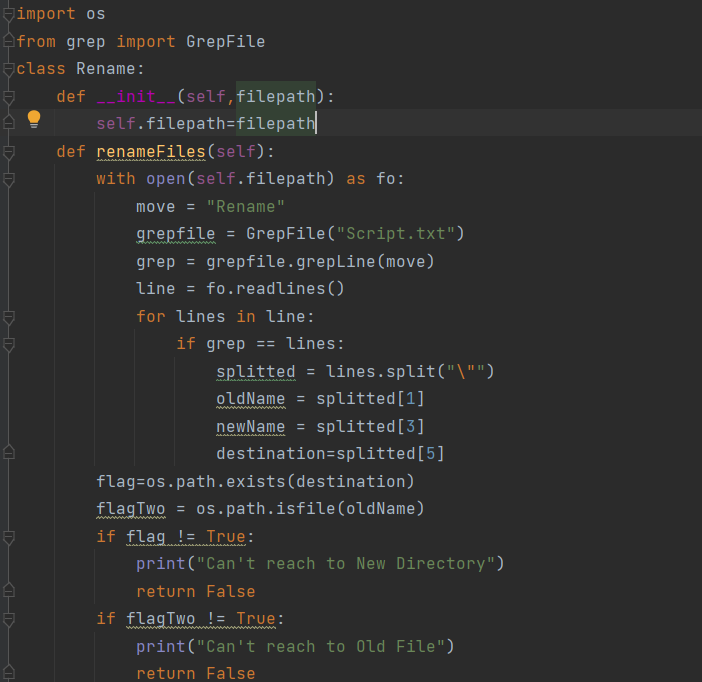


Figure 6 move class

## For Rename cmd:

We find the line contains Rename word by grep then split the line to get the source file, old name and new name and check if it’s exist then by os.rename we convert the name.



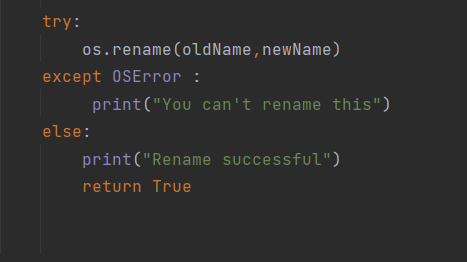
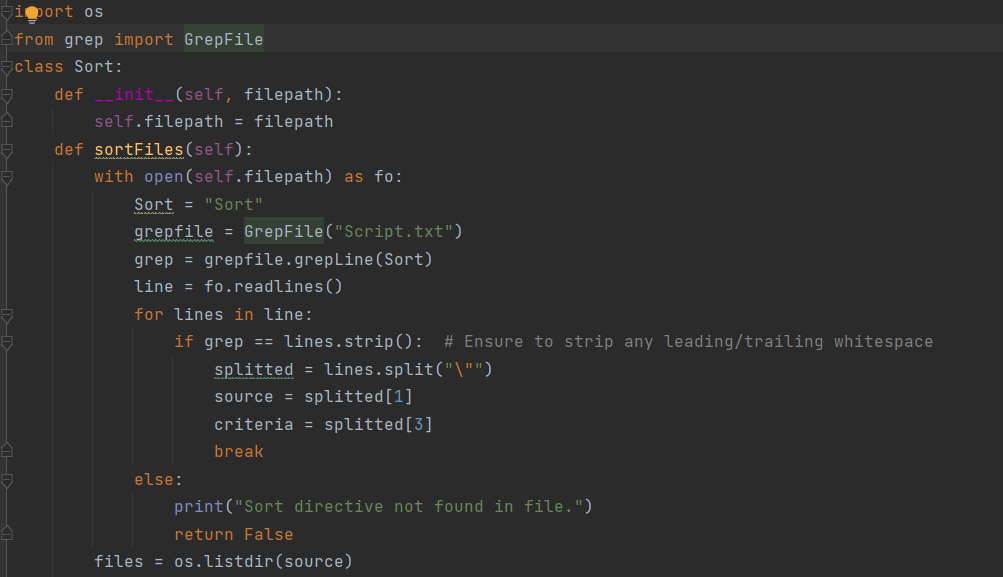


Figure 7 Rename class

## For Sort class:

We call grep class then after reading the name of folder and specific criteria then make the sort based on it.



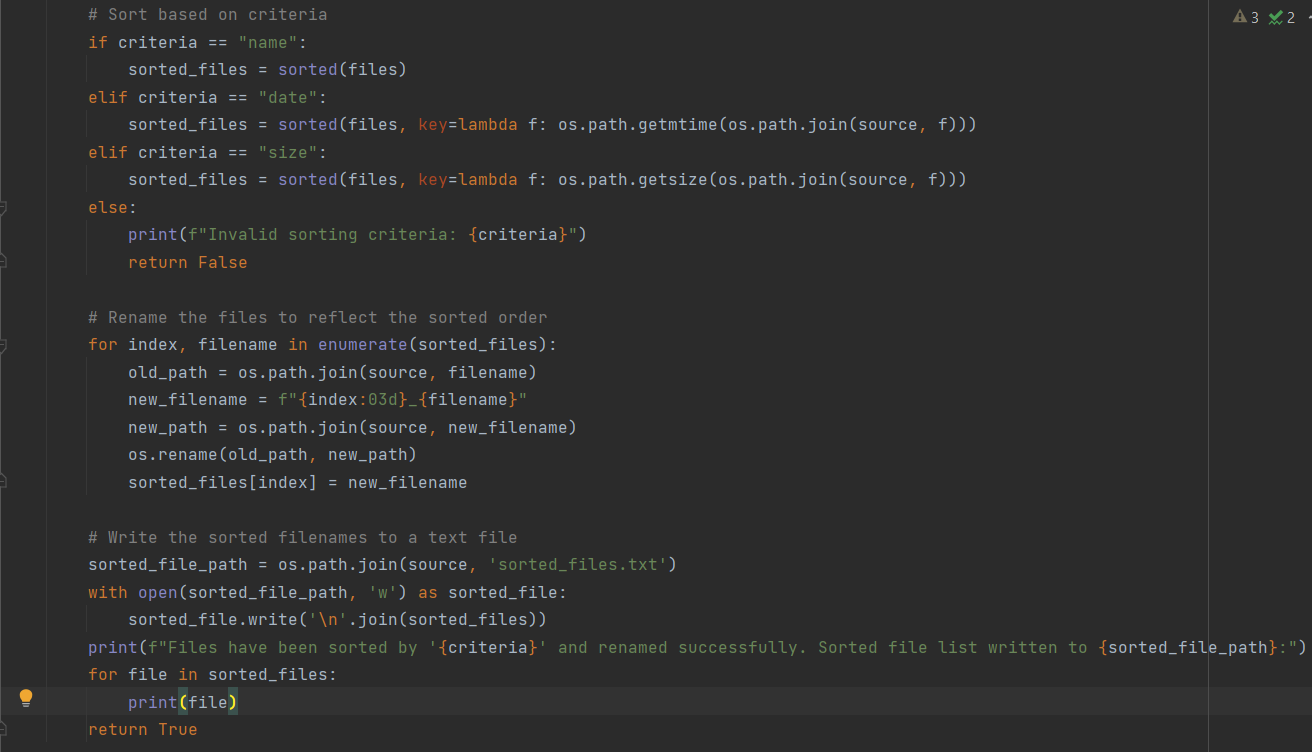


Figure 8 sort class

## For main:

First, we read the attributes from json file to use it in my project, then make function to write in csv files the output.

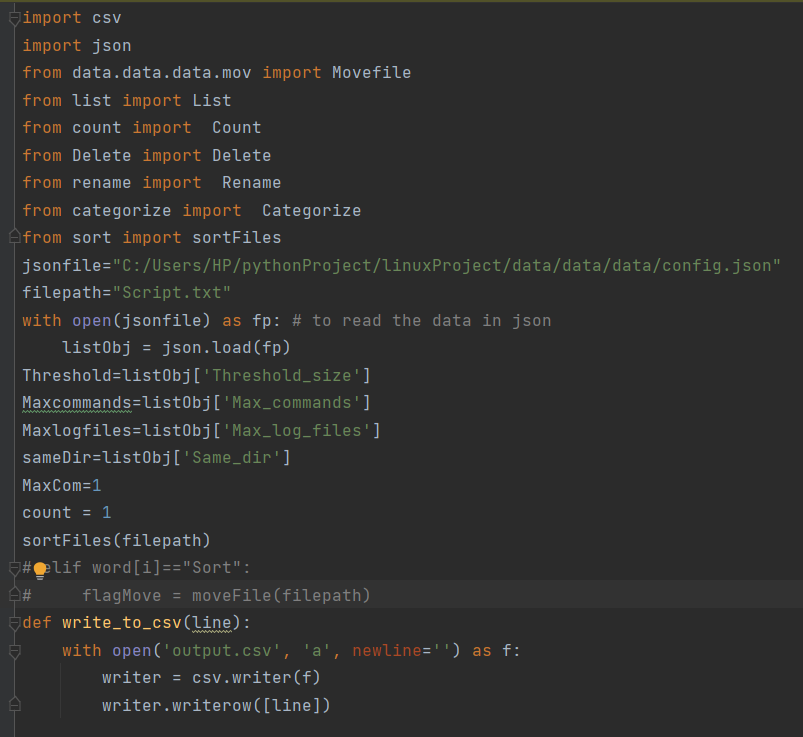


Figure 9 Main 1

The second function, write in log file when the output must be in log file.

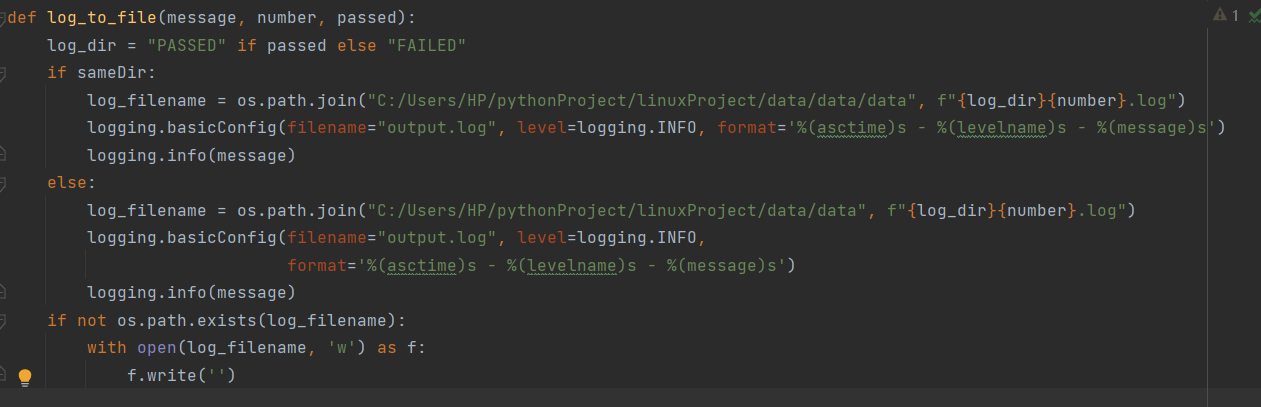


Figure 10 Main 2

Because there are limit number of csv or log file, we made values to count each file, then read Script.txt file and when read the first command will search to the if statement suit to this line and call the class for this line to execute. Based on value output will wrote in specific file.

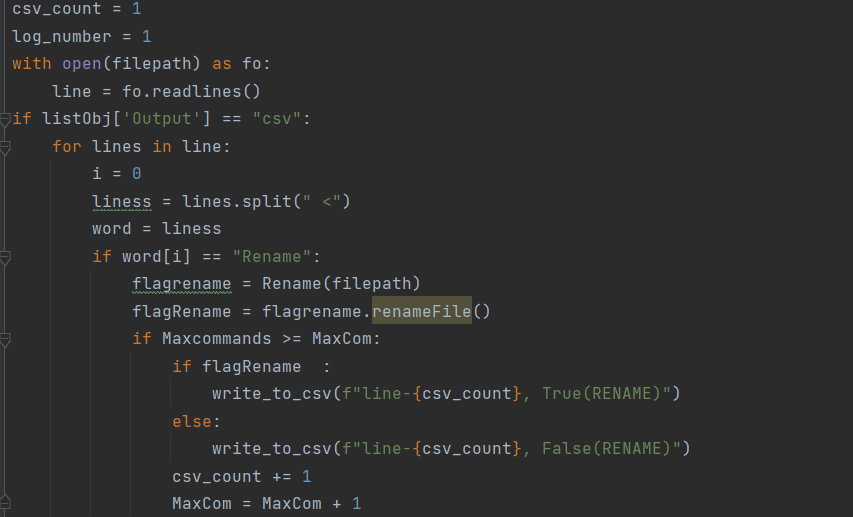


Figure 11 Main 3

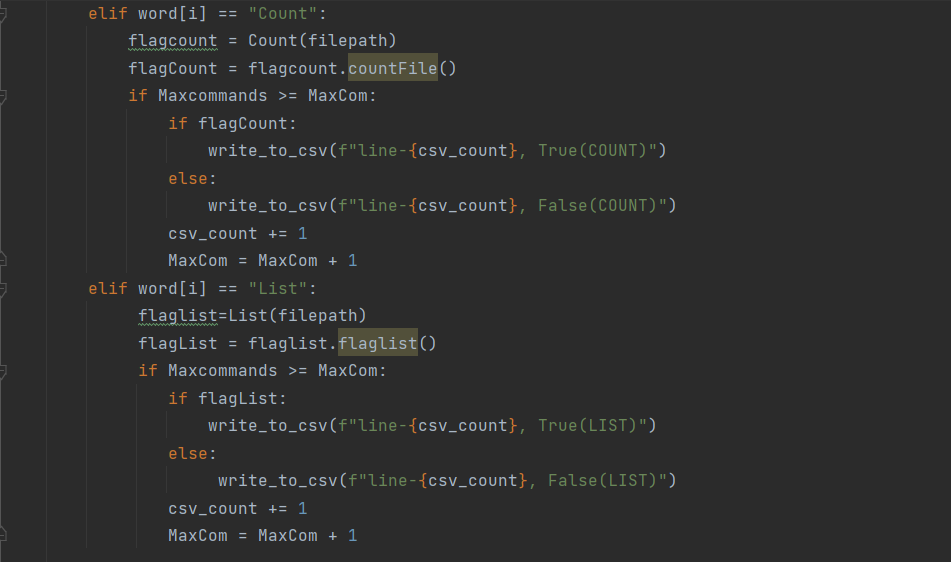


Figure 12 Main 4

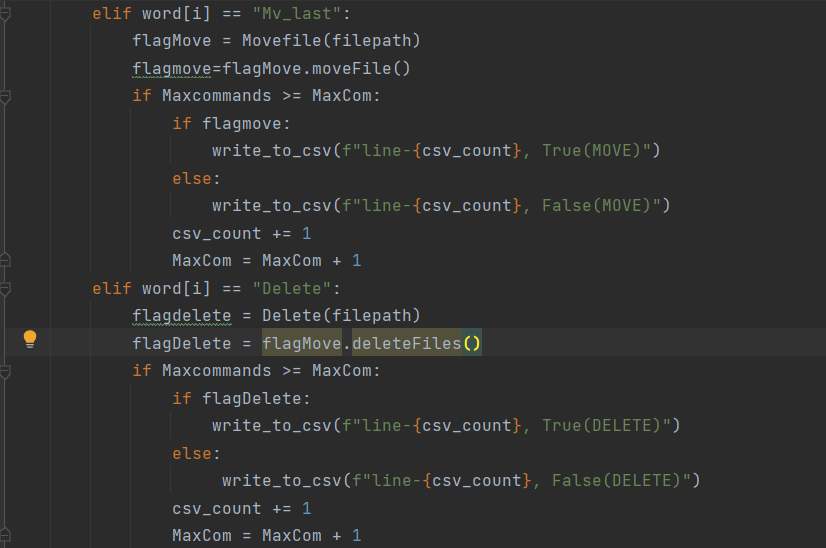


Figure 13 Main 5

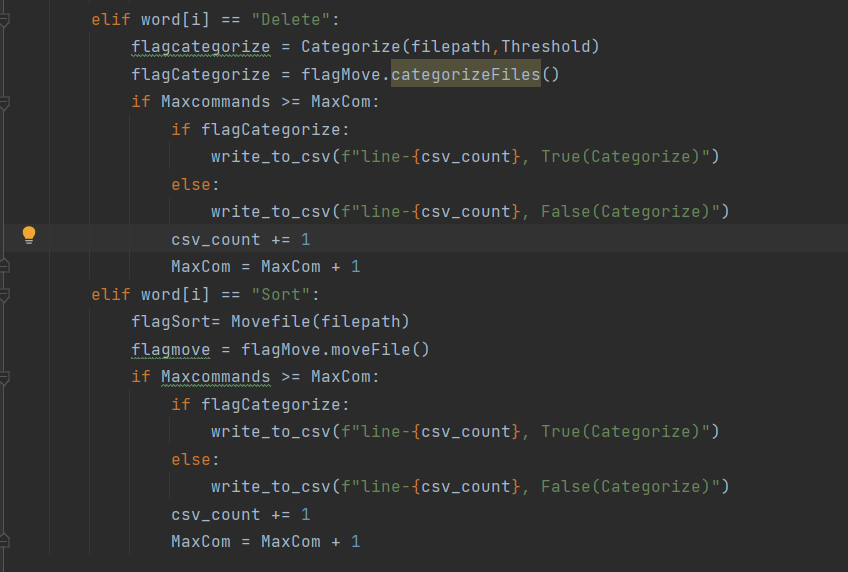


Figure 14 Main 6

When the output be log file

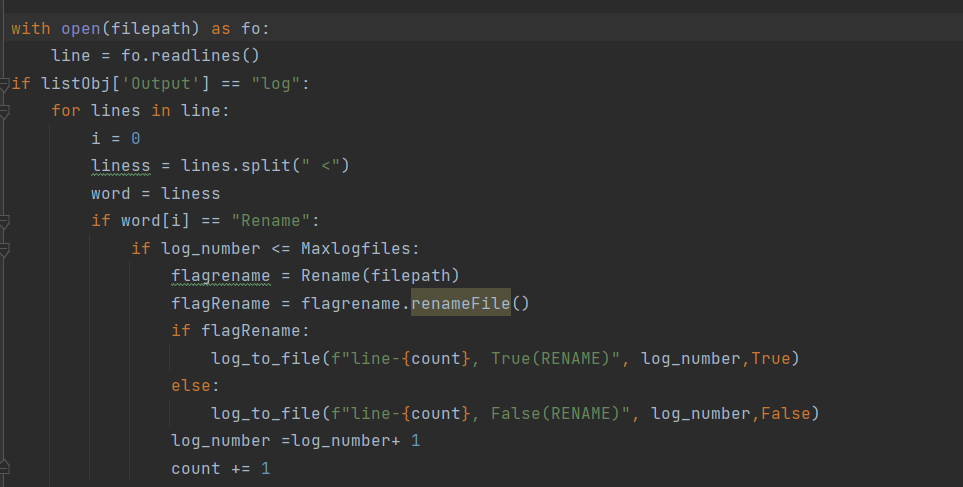


Figure 15 Main 7

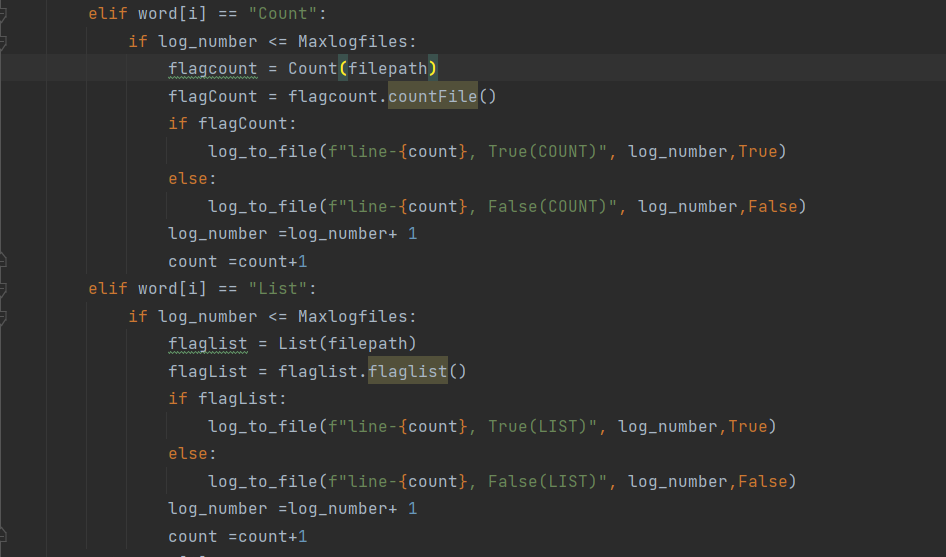


Figure 16 Main 8

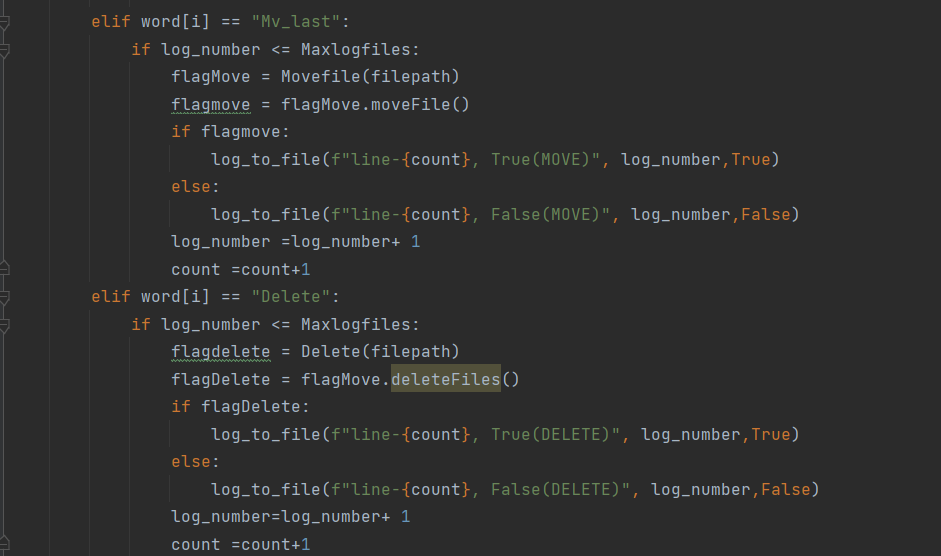


Figure 17 Main 9

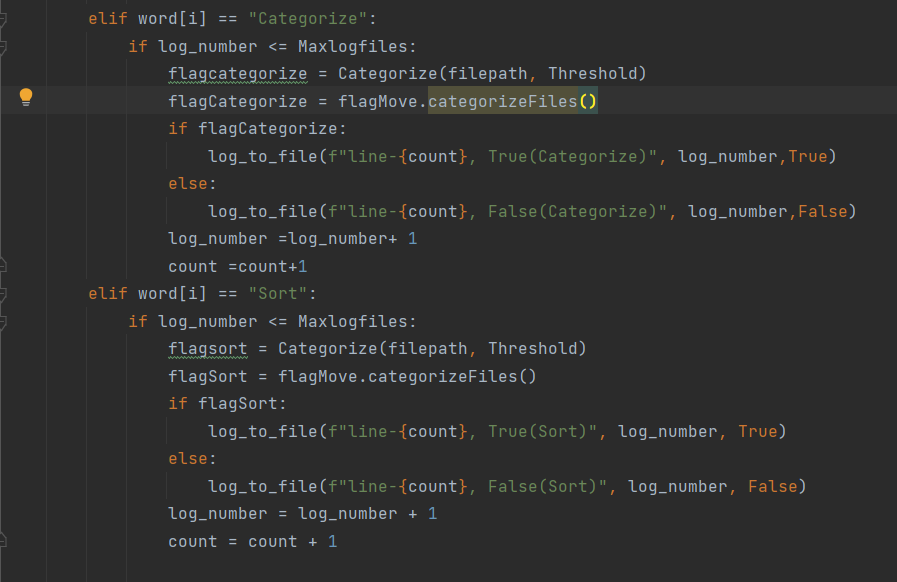


Figure 18 Main 10

# **Running the Program**

First, we make folder that contains four files.

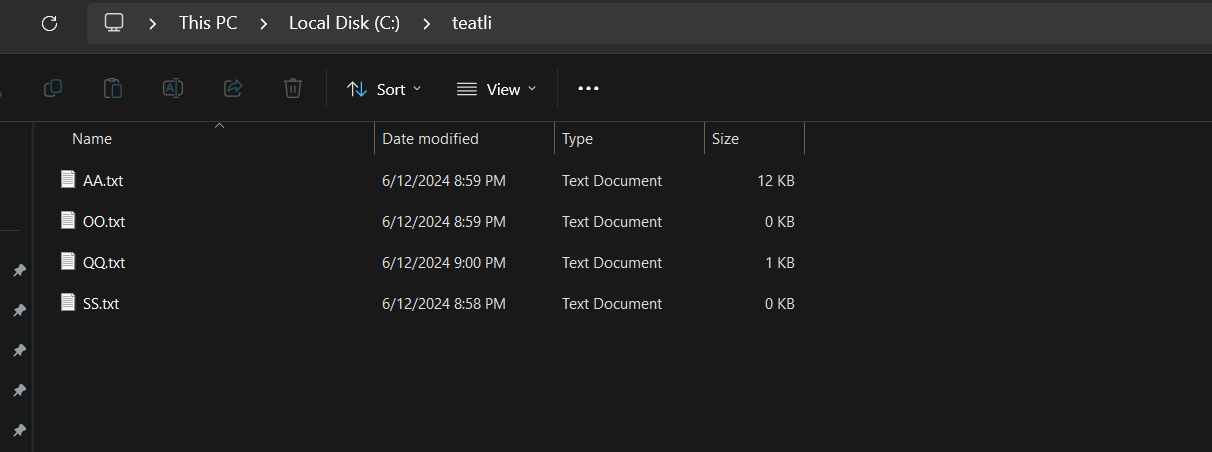


Figure 19 teatli folder

And we make empty second folder.

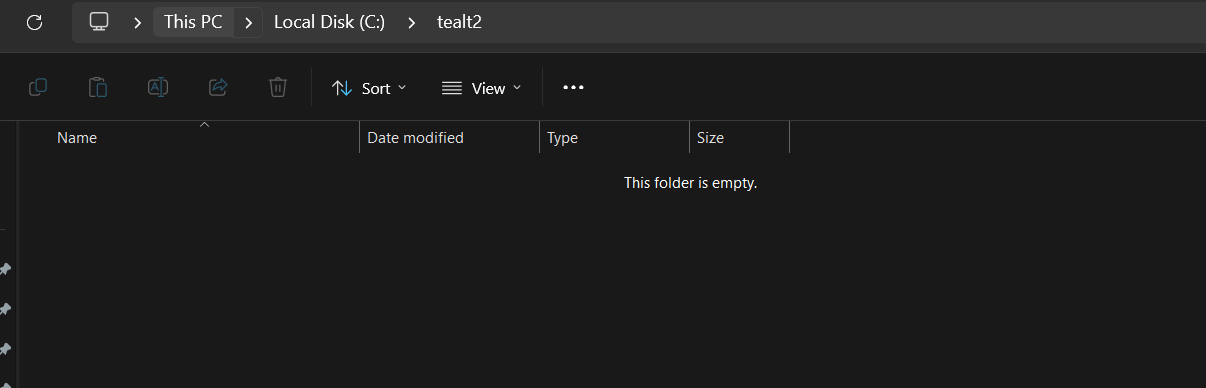


Figure 20 tealt2 folder

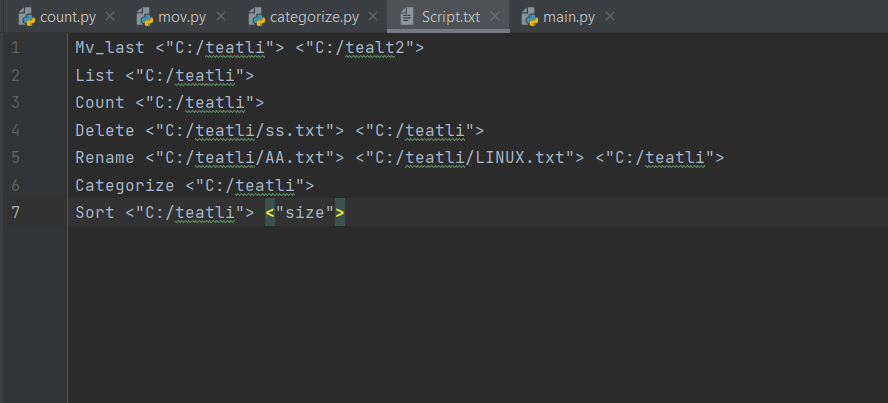


Figure 21 Script txt



Figure 22 JSON file

## After running:

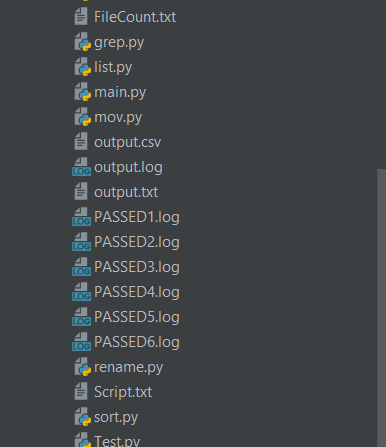


Figure 23 Run 1

In log file.

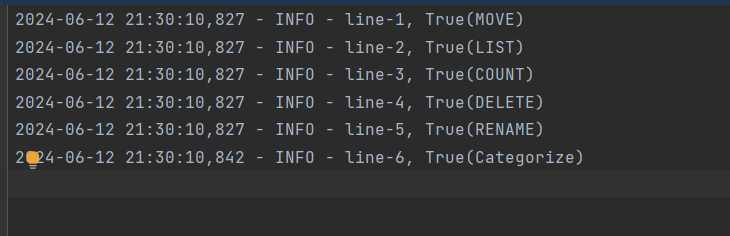


Figure 24 Run 2

The OO file move to destination folder.

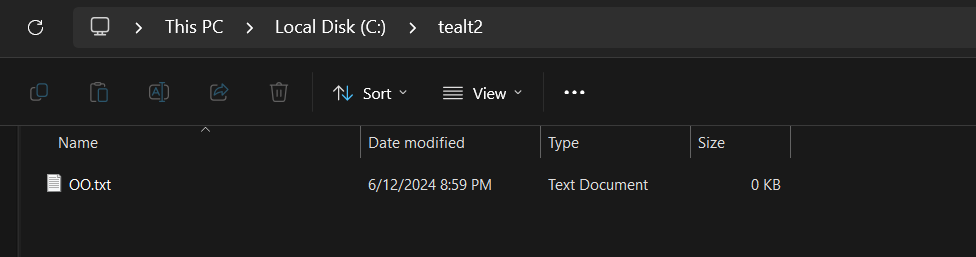


Figure 25 move run

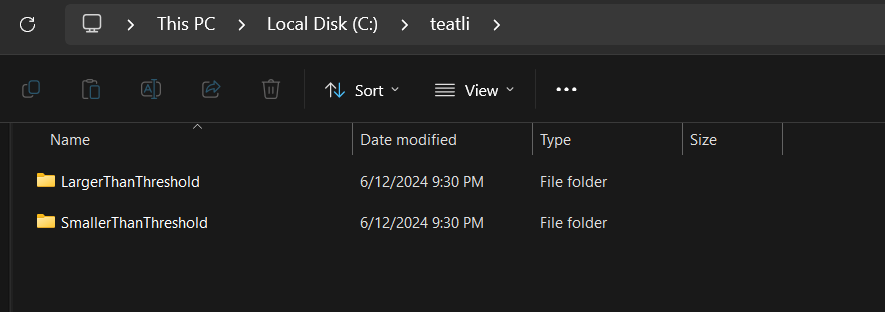


Figure 26 category run

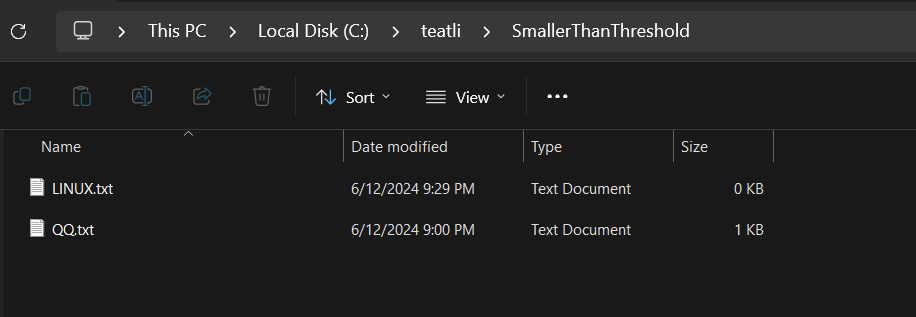


Figure 27 category run 2



Figure 28 rename run



Figure 29 count run

## Csv running:

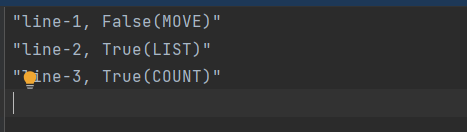


Figure 30 CSV run