DATA 601: Project I

You will create a software 'product' (Call it FilePro), which will be a python class that analyzes the folder1 provided to you (and all sub-folders and files within those folders) and does some operations on it.

This would be accessed from the file “main_notebook.ipynb”. You may have (one or more) .py or .ipynb files to host your code, but it’s the “main_notebook.ipynb”, where the code will be accessed, demonstrated and also documented (using markdown cells). Also, use comments in your code to document the functionality of your code and make it more readable.

Required functionality:

Note: All methods are mentioned with the suffix parenthesis (…). But the implementation of a given method is up to you (including what arguments to pass, unless explicitly specified). You can also create more methods to support the required methods.

DOM : Date of Modification

Cwd : Current Working Directory

1. get_info(…): For every file inside folder1 (including files in sub-folders), calculate the extension of the file (.txt, .csv, .pdf etc), the size (in bytes), a hash value based on a checksum hash technique (MD5,SHA-256, your choice), the filename, the filepath and the date the file was modified on. Store this data in a file "file_info", the type of which is your choice [json,csv,txt etc].
2. __init__(…): The constructor should take the path where folder1 is located. Then, if “file_info” file is not present in cwd or if “file_info”’s DOM is older than 2 days, run get_info(…) and get the most updated version of “file_info”.
3. generate_report(…) : When this is run, it generates a report on the screen with an option to save it to file (txt or csv). It would contain the following information:
    a. different types of files present in the folders (including all sub-folders)
    b. mean and median size of the files (in general and by type)
    c. the folder name (not including subfolders) with the largest amount of files by (i) number and (ii) size

    All the data required for this method will come from the “file_info” file. When this functionality is executed, check if the “file_info”’s date last modified is greater than 2 days, Call get_info() and overwrite the old “file_info” (and then generate the report).

Original functionality :

1. Apart from the above, you need to come up with and implement a functionality that’s not mentioned above. A few suggestions are : interactive CLI, simple GUI, produce graphs from the data (during generate_report()), a plot for showing the tree of all the folders inside folders1, adding more features to generate_report() (For example mean and median date of modification) or any other thing not mentioned here. You are free to choose.

This functionality can be a separate method in your class, or be a part of an existing method, or even outside the class

Your project will be graded based on the following criteria:

1. **Functionality [25 pts]** Whether the required functionality is implemented properly.
2. **Documentation [25 pts]** Your "main_notebook.ipynb" and comments in the code are going to be graded for the clarity of your documentation and presentation.
3. **Usability [15 pts]** Regardless of you making a CLI/GUI/ or just accessing your code through functions or classes written by you, the project will be graded on how "usable" your code is.
4. **Elegance [10 pts]** Points for the elegance of your implementation, proper organization, and the ease of readability.
5. **Originality [15 pts]** Points for the original functionality that you implement.
6. **Robustness [10 pts]** Make sure your code isn't error prone while executing it.
7. **BONUS [15 pts]:** Implement one additional original feature.

There is not going to be a software demonstration or presentation. Therefore, how you demonstrate and document your product in the "main_notebook.ipynb" becomes even more important.