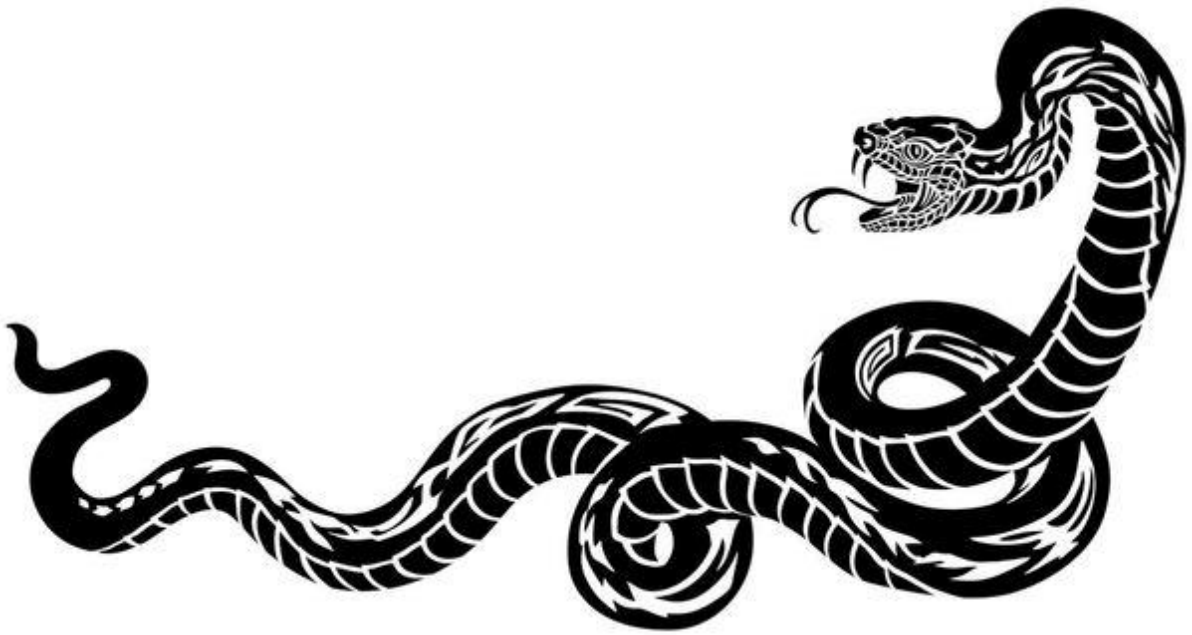
Python – Bottom Up

~ Practical Project ~



GENERAL INFORMATION

Project Name: File Analyzer

Project Submission Due Date: 31.10.2024

IMPLEMENTATION NOTES AND GUIDELINES

1. The solution must take advantage of Python's Standard Library, without any 3rd-party helper libraries or frameworks.
2. The solution should have one main script (explicitly named *main.py*) and the rest of the logic can be implemented either within that same script or within other scripts.
3. The project structure is completely based on free choice.
4. The solution must be runnable not only in PyCharm, Visual Studio Code or any other development environment, but also through Command Prompt and Git Bash.
5. The solution may be implemented with use of classes and objects or through use of scripting paradigm, with functions only (no classes, no objects) - the choice is yours.
6. The solution must take into consideration non-existing file paths.
7. The solution must take into consideration empty files by providing a single line in the analysis report file:
   No content to be analyzed in the file on <FILE_PATH>.
8. The solution must take into consideration files of formats different than those of textual, JSON and CSV by providing a single line in the analysis report file:
   The file on <FILE_PATH> is of format <FILE_FORMAT>, which is not supported by this application.
9. **You do not need to implement all the requirements proposed by specification in order to pass the practical test** – the core functionality of loading the content and writing the analysis report is sufficient for passing the practical test. Additional requirements proposed by specification will bring you bonus points and a more complete project.

WHAT WILL BE CHECKED WHEN RATING AND TESTING THE SOLUTION

1. Following of the Python naming conventions (variables, functions, classes, etc.).
2. Addition of type hints (optional for variables, but mandatory for function definitions).
3. Addition of docstrings (i.e. code documentation) for each function/method.

SPECIFICATION

The main idea is for you to implement a Python application which is capable of loading the contents of textual, JSON or CSV files.

Furthermore, the solution should take the loaded content, count specific elements and provide an analysis report in form of a textual file.

When it comes to counting specific elements, the solution should count:

- Number of words in case of a textual file
- Number of root (i.e. 1st-level) elements in case of a JSON file
- Number of rows and columns in case of a CSV file

The analysis report file should be saved in *reports* directory, which should be **dynamically** created on the same level as the main script.

Every analysis report file should be following the naming convention:

analysis_report_ <FILE_NAME>.txt

Every analysis report file should follow the content structure as proposed:

File Name: <FILE_NAME>
Number Of Words / Root Elements / Rows And Columns: <COUNTED_NUMBER>
Analysis Performed At: <DATE_TIME_TIMESTAMP>

Example:

A file named *users.csv* is loaded, it has 20 rows and 5 columns.

The content of *analysis_report_users.txt* file should look like this:

File Name: users.csv
Number Of Rows And Columns: 20 rows, 5 columns
Analysis Performed At: 2024-09-20 15:28:26.661038

Textual file contents should be first cleaned from the following special characters: **.,:;!?=+-_***