Assignment 2 (A2)

Due date: Thursday, February 29, 2024 (11:59pm)
Submission via Git only

Contents

Programming environment

For this assignment you must ensure your code executes correctly on the *reference platform* (i.e., computers on ELW B238, which can be accessed using ssh) you configured as part of Lab 01. This same environment will also be used by the teaching team when evaluating your submitted work. You will have to make sure that your program executes perfectly on the reference platform. If your programs do not run on reference platform, your submission will receive 0 marks.

All test files and sample code for this assignment will be available in the ‘a2’ folder of the ‘assignments’ branch of your git repository. To avoid conflicts with A1 submissions, **the files for A2 will be available on Friday, February 16th**. To retrieve the files, you need to run:

`git pull`, if you already cloned the ‘assignments’ branch as indicated in the Step 3 of the step-by-step guide for submissions. Of course, you’ll need to be inside the repository folder to execute this command.

Individual work

This assignment is to be completed by each individual student (i.e., no group work). You are encouraged to discuss aspects of the problem with your fellow students. However, sharing of code fragments is strictly forbidden. Note SENG 265 uses highly effective plagiarism detection tools to discover copied code in your submitted work. Both, using code from others and providing code to others, are considered cheating. If cheating is detected, we'll abide by the strict UVic policies on academic integrity: https://www.uvic.ca/library/help/citation/plagiarism/

Learning objectives

- Learn or review basic features of the Python 3 programming language.
- Use Git to manage changes in your source code and annotate the evolution of your solution with messages provided during commits. Remember, the only acceptable way of submitting your work is using Git.
- Test your code against the provided test cases.

song_analyzer.py: Descriptive analytics about songs data

a) In A2, song_analyzer is a small program that uses relevant Python structures and libraries to process songs data to produce descriptive analytics.

b) Based on provided arguments and data files (i.e., datasets), song_analyzer will help us answer the following questions:

    Q1: What were the 6 least streamed songs by Dua Lipa during 2023?
    Q2: What were the most streamed songs by Drake during 2023?
    Q3: What were the top 20 most streamed songs during 2023?
    Q4: What were the top 5 songs released in 2023 with most appearances in Spotify playlists during 2023?
    Q5: What were the top 7 songs released in 2023 with most appearances in Apple Music playlists during 2023?

c) After each execution, your program **must produce a file named output.csv** that represents the answer to the question asked to program (i.e., arguments passed).

d) The most reliable way (and the only one encouraged) to test your program is to use the provided **tester** file which will validate the output produced by your program, given a particular question.

Restrictions

- Your program must run as a script.
- Your program must be decomposed into easy-to-understand components. Good program decomposition is required. Methods or functions must be short and effectively parameterized.
- Unwieldy functions are not accepted. The main function should especially be easy to understand and represent a decomposition of the problem at hand.
- Typing (i.e., type hints) must be used in variables and functions defined in your program.
- Do not use global variables.

- **You can only use Python modules and libraries that DO NOT require additional installations on the reference platform**. As part of the installation and configuration of the reference platform, we included relevant libraries for managing data such as numpy and pandas. **Failing to comply with this restriction will result in 0 marks for the assignment.**
- Keep all your code in one file (**song_analyzer.py**) for this assignment. In Assignment 4 we will use the multi-module features of Python.

Testing your solution

- This time the **tester** file will execute your program automatically. You'll only need to pass the number of the question as an argument (e.g., `./tester 1`). If no arguments are passed to the tester file (i.e., `./tester`), it will run the tests for all the questions. Using a specialized library that compares the differences between .csv files, the **tester** file will describe the differences between the expected output and the one provided by your program.
- Refer to the example commands in the file "TESTS.md" for appropriate command line input. Your solution must accommodate all specified command line inputs.
- Make sure to use the test files provided as part of this assignment.
- Use the test files and listed test cases to guide your implementation effort. Develop your program incrementally. Save stages of your incremental development in your Git repository.
- For this assignment you can assume all test inputs are well-formed (i.e., exception handling is not required).
- **DO NOT rely on visual inspection**. You can use the provided **tester** file so you can verify the validity of your outputs in a simpler manner.

What to submit

A single Python source file named "song_analyzer.py" submitted (i.e., Git push to the remote 'assignments' branch of your repository) to the a2 folder in your repository (i.e., using the 'assignments' branch). To achieve this, you'll need to follow the steps described in the step-by-step guide for submissions.

Grading

Assignment 2 grading scheme is as follows. In general, straying from the assignment requirements might result in zero marks due to automated grading.

Scheme

- **(50%) # Tests Passed**
- **(50%) Code Qualitative Assessment**
    - **(20%) Functional decomposition, program-scope or file-scope variables:** quality coding requires the good use of functions. Code that relies on few large functions to accomplish its goals is considered poor-quality code. Typically, a good program has a main function that does some basic tasks and calls other functions that do most of the work.

- **(20%) Code structure:** The code style must be consistent, uniform, and facilitate readability throughout the file.
- **(15%) Proper naming conventions:** You must use proper names for functions and variables. Using random or single character variables is considered improper coding and significantly reduces code readability. Single character variables as loop variables is fine.
- **(15%) Typing:** to facilitate readability and maintainability in your solution, typing (i.e., type hints) is required for this assignment when declaring variables and functions.
- **(10%) Debugging/Comment artifacts:** You must submit a clean file with no residual commented lines of code or unintended text.
- **(10%) Documentation and commenting:** the purpose of documentation and commenting is to write information so that anyone other than yourself (with knowledge of coding) can review your program and quickly understand how it works. In terms of marking, documentation is not a large mark, but it will be part of the quality assessment.
- **(10%) Quality of solution:** marker will access the submission for logical and functional quality of the solution. Some examples that would result in a reduction of marks: solutions that read the input files several times, solutions that represent the data in inappropriate data structures, solutions which scale unreasonably with the size of the input.

Scale

A grade		B grade		C grade		D grade		F grade		
grade	marks	grade	marks	grade	marks	grade	marks	grade	marks	description
A+	90-100	B+	77-79							Either no submission given, or submission represents little work or none of the tests pass. No submission, 0 marks. Submissions that do not run, 0 marks. Submissions that fail all tests and show a poor to no effort (as assessed by the marker) are given 0 marks. Submissions that fail all tests, but represent a sincere effort (as assessed by the marker) may be given a few marks
A	85-89	B	73-76	C+	65-69					
A-	80-84	B-	70-72	C	60-64	D	50-59	F	0-49	

Input specification

- All input test files are in the CSV format and are based on the “Most Streamed Spotify Songs 2023” dataset in Kaggle.com.
- The metadata (i.e., description of each column) for the input datasets can be found online in the ‘Key Features’ section of the dataset description.

Program Arguments

Argument	Argument Description	Optional	Value Example(s)	Value Description
--data	The path of the .csv file containing the input data.	No	data.csv	N\A
--filter	The column used to filter rows. If not provided, the argument --*value* won't be provided. When not provided, rows will only need to be sorted as no filters will be applied.	Yes	ARTIST	Refers to the column '*artist(s)_name*'
			STREAMS	Refers to the column '*streams*'
			YEAR	Refers to the column '*released_year*'
--value	The specific value to filter rows using the column specified by --filter. If not provided, the argument --*filter* won't be provided. When not provided, rows will only need to be sorted as no filters will be applied.	Yes	Dua Lipa	For example, as in Test 01, all the rows **containing** the value 'Dua Lipa' for the column '*artist(s)_name*' will be selected. Rows that do not include the provided value should be dismissed.
--order_by	The column used to sort the rows.	No	STREAMS	Refers to the column '*streams*'
			NO_SPOTIFY_PLAYLISTS	Refers to the column 'in_spotify_playlists'
			NO_APPLE_PLAYLISTS	Refers to the column 'in_apple_playlists'
--order	The type of order the data will follow.	No	ASC	Ascending order.
			DES	Descending order.
--limit	The number of songs that will be displayed. If not provided, all songs meeting the filtering criteria should be displayed.	Yes	6	N\A

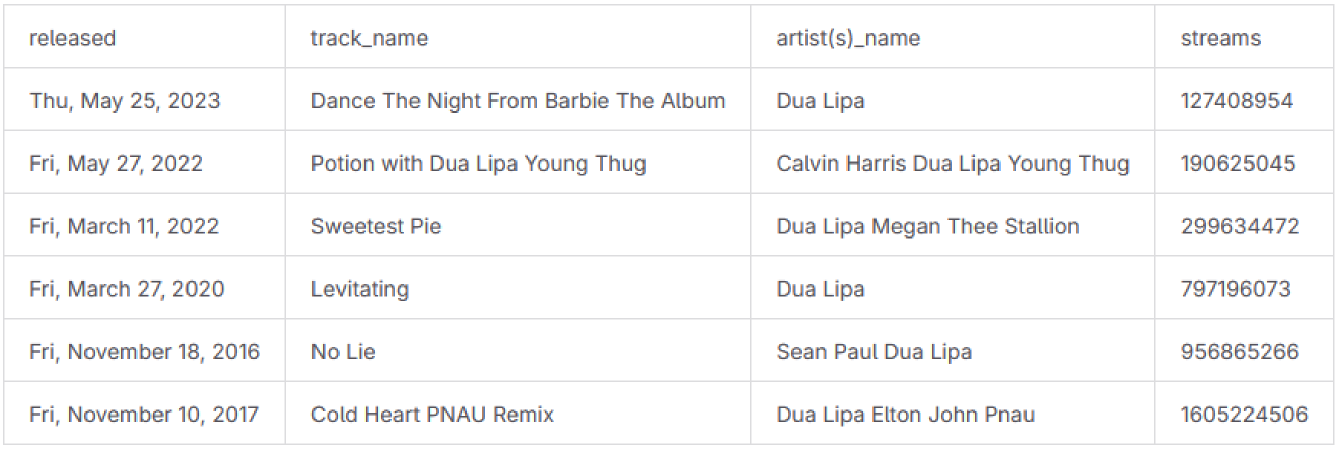
Input Example (Test 01)

```
./song_analyzer.py --data="data.csv" --filter="ARTIST" --value="Dua Lipa" --
order_by="STREAMS" --order="ASC" --limit="6"
```

Output specification

- After execution, your program must produce\create the following file: **output.csv**. This file will contain multi-dimensional data (i.e., a table with several columns) that represents the answer to the question passed as an argument to the program (e.g., **Q1**, **Q2**, **Q3**, **Q4**, **Q5**).
    - The first column (i.e., *released*) always refers to the song release date. It follows the format: *weekday*, *month_name day*, *year*
    - The second column (i.e., *track_name*) always refers to the name of the song.
    - The third column (i.e., *artist(s)_name*) always refers to song's artist(s).
    - The fourth column refers to the column used to sort the data (e.g., streams, in_spotify_playlists, in_apple_playlists).

Example of expected output for Test 01 (test01.csv)



released	track_name	artist(s)_name	streams
Thu, May 25, 2023	Dance The Night From Barbie The Album	Dua Lipa	127408954
Fri, May 27, 2022	Potion with Dua Lipa Young Thug	Calvin Harris Dua Lipa Young Thug	190625045
Fri, March 11, 2022	Sweetest Pie	Dua Lipa Megan Thee Stallion	299634472
Fri, March 27, 2020	Levitating	Dua Lipa	797196073
Fri, November 18, 2016	No Lie	Sean Paul Dua Lipa	956865266
Fri, November 10, 2017	Cold Heart PNAU Remix	Dua Lipa Elton John Pnau	1605224506