CSC-40076: Design and Programming

Assignment 1

Preamble

In week 3 we studied Data Types and Data Structures and looked carefully at and practised reading and writing file in Python, particularly the CSV (Comma Separated Values) format that is a common import and export format for spreadsheets, databases, and other applications such as the data sciences. Recall that a CSV file stores tabular data in which each data field is separated by a delimiter, usually a comma – hence the name. CSV files are saved with the .csv file extension. We also studied sorting data.

Libraries like pandas exist to help with this problem but we are assessing **your** ability with handling files, so **we do not allow pandas to be used for this summative assignment.**

The Problem

A small college lending library run by volunteers has donated its books to the case study library. They utilised a simple but effective spreadsheet-based cataloguing and loan system. Their books and data files are thus available for some experimental work. We have been given 3 files, books.csv, bookloans.csv, and membership.csv. We are ignoring membership data for this assignment. The library has a standard loan period of 14 days. A borrowed book returned after 19 days is 5 days late.

The text file **books.csv** contains a subset list of their books in CSV format. It has a header row. The CSV files are encoded as UTF-8. You may assume it is free from errors i.e., no records are flawed.

The text file **bookloans.csv** contains data on borrowed books in CSV format. It does not have a header row. Be warned that this file has errors. Each row holds a *book_number*, *member_number*, *date_of_loan* and *date_of_return* separated by commas. Only transactions of books ***borrowed from our library and returned*** during the year of the transaction are valid. The order of the records is from the transaction of a member borrowing a book, so it is in the order of the date of the loan. Assume that all the books borrowed and returned were in 2023.

The year of the transaction is 2023.

Some observations already made are:

- Each book has only a single copy for a loan, so if it is on loan, it cannot be borrowed. In other words, it can only be borrowed if it is on the shelf.

- A date*_of_return* of 0 (zero) means that the book has not been returned yet so is not on the shelf and cannot be borrowed.

- All loans are returned within the year or not returned at all.

- The file contains thousands of valid loan transactions but many invalid ones.

- The *date_of_loan* is a single float number representing the date of the loan in Microsoft Excel Epoch Format that is stored for our processing as an integer or a string that evaluates to an integer. We are ignoring fractions of a day, so integers are fine. It represents the number of days elapsed since 1/1/1900.

- In the year 2023, the 1st of January 2023 had an Excel epoch date of 44927 and the 31st of December 2023 had an Excel epoch date 45291.

There is a separate notebook containing functions that may be useful that relate to Excel epoch dates and the more usual 'YYYY-MM-DD' format. It is possible to do the tasks without the 'YYYY-MM-DD' format, but you may wish to convert dates for your own convenience in understanding the data.

Statistics received from the library tell us some facts about the files provided by them. These are listed below for your information.

books.csv

- the highest book number in the collection is 120.
- the highest member ID number of a borrower is 200.

bookloans.csv

- The number of invalid loans (not returned) is 51.
- The maximum days borrowed: 20.
- The minimum days borrowed and returned: 1.
- The lowest book number borrowed: 1.
- The highest book number borrowed: 140.
- The lowest member number of borrowers: 1.
- The highest member id number of a borrower: 200.

We need to compute the lost data from the files.

Overview of assessment

This assessment has three main areas of design, the functionality of your code and the quality of documentation.

- *Design* assesses your use of functions, lists, and dictionaries (or any reasonable data structure), and evidence of function testing, connecting CSV files for analysis, and data checking and cleaning. Exception handling and validation are examples of testing.
- *Functionality* assesses tasks 1, 2, and 3.
- *Quality of documentation* assess the appropriate use of markdown in your Jupyter notebook (.ipynb) to explain work, code readability (e.g., commenting on code using # or docstring), and opinion of the quality of the data. Your opinion on the popularity measure of a book could be pertinent. Do you agree with us or disagree, and if so, why?

Tasks

Task 1.

This task requires reading books.csv and bookloans.csv into nested lists or dictionaries and ensuring you have a list of valid book loans. It requires processing to produce a printed report. You need to consider the validity of the data used in your report or your statistics will be wrong.

A report of **books borrowed in 2023** is required listing the returned book number, book title, author, and the number of days borrowed. This should be a valid list excluding the false books and non-returned borrows. We want to identify and show the least popular and most popular books.

What is the best measure for the popularity of a book in a lending library, the number of times it is borrowed in a year, or the number of days for which it was borrowed in a year?

That is a design issue.

I. The *number of times a book is borrowed* in 2023 provides a measure of its overall popularity and demand among library members. This measure can determine which books:

   a. we purchase additional copies of,
   b. we feature in promotions or displays, and
   c. we recommend to members looking for recommendations.

II. The *number of days* for which a book is borrowed in a year provides a measure of its sustained popularity and engagement among library members. This measure can be useful in determining which books to keep in the library's collection over time, as books that are consistently checked out for longer periods may be more valuable to the library's members.

To measure the *number of times* a book is borrowed, you can simply count the number of times each book appears on the loan list in 2023. This measure is relatively easy to obtain as it only requires tallying the number of times each book is checked out for a loan.

To measure the *number of days* for which a book is borrowed, you need to consult the record of the dates on which each book is checked out and returned, and then calculate the number of days between those dates, and the period of the loan. This measure requires arguably more effort as it involves tracking individual loans and calculating the length of time for each loan. The file bookloans.csv is a transaction file in order of the date a book is borrowed. We wish you to process that file to compute the ***number of days for which each book was borrowed***.

Popularity is measured by the ***number of days for which each book was*** borrowed in 2023.

You may disagree with this. That is okay. You are welcome to argue your countercase under the quality section of the assessment. But for this task the *number of days for which each book is borrowed*

Note: If you are doing a reassessment, this measure may be different from the presentation that you studied and were assessed for this assignment. This measure supersedes the previous requirement.

We require the listing of books sorted by **reverse order** of their popularity.

Write code to accomplish this. Use functions where appropriate and comment on your code often to show your assessor what you are doing and why.

Design is sometimes not clear-cut like programming can be. As said above, it's arguable that our measure of the popularity of books is flawed. If so, show us why. You may use Markdown cells in the notebook to report this part. See the external link.

Task 2.

The library is keen to know the interests of its readers to influence purchasing decisions. The books have different genres. You cannot assume all genres were borrowed; we are only interested in those that were. Is this sensible do you think? Or would a genre not borrowed at all be significant? Bear such issues in mind for the quality section. We decided to ignore sub-genres this year.

- Write code to produce a popularity report of all **genres** of books borrowed in 2023 and how many books are in that genre.
- Write code to sort the report. Default or reverse order is okay.

Task 3.

Statistics on loans and late loans are lost.

For the library books, we need you to calculate the:

- Number of loans.
- Number of days borrowed.
- Number of loans returned late.
- Number of days late.
- Average number of days a book was borrowed.
- The percentage proportion of books returned late.
- The average late period of books returned late.

A book borrowed and returned on the same day was borrowed for one day. If a book is returned after 19 days, it is 5 days late.

The report should show:

a) **The average number of days that a book was borrowed**.

   Assume that if books were borrowed for 20,000 days and the number of loans was 2,000, the average number of days that a book was borrowed was 10 days.

b) **The percentage proportion of books returned late**.

   Assume that if the number of loans returned late was 500 when the number of loans was 2,000, the proportion of books returned late was 25%.

c) **The average late period of books returned late**. i.e., a book returned late is on average this many days after the grace period.

   Assume that if the number of days late was 3,000 and the number of loans returned late was 500, the average late period of books returned late was 6 days.

Code your own documented functions to compute these statistics. Do not use statistical packages because we need to see that you can design the solution yourself. Moreover, packages make false assumptions.

You could print like below:

```
print(f"{number_of_loans:.0f}") # e.g., 2000
print(f"{average_loan_days_a_book_was_borrowed:.1f}") # e.g., 10.1
```

However, tabulation of some kind is encouraged.

Quality Section

Comment on your work using markdown in addition to docstrings or other comments in the code.

Use docstrings and comments to make your code clearer to your assessor.

What is your opinion on the quality of data you were provided and its limitations?

Submission

Your code and documentation should be submitted as a Jupyter Notebook. Not .py files, which cannot be assessed because they are in the wrong format.

When creating your solutions, it's important to think about how to break down the system into functional parts and determine the most suitable data structures for achieving an efficient solution, such as utilising lists and dictionaries. Additionally, it's crucial to thoroughly test your functions and document the testing process using cell markdown and comments marked with either # or """ to demonstrate how you've accomplished this. The use of docstrings with functions is assessed. Import of statistics packages or pandas will be penalised.

Design aspects of your solution will attract a maximum of 15 Marks of the 40 available. Achieving the correct functionality will attract a further maximum of 15 Marks and the remaining 10 Marks will be awarded to reflect the quality of your documentation. Your documentation should describe any ‘sanity checks’ e.g., printing a data structure to confirm it contains what you expected but then commenting that out of your function processing data. Also, describe your opinion of the quality of the data and how going forward the case-study project might benefit from knowing that. Please describe the assumptions you made and the extent to which you feel your completed tasks have met the requirements.

We don’t need the given files back! Ensure that your code functions with the original data files present in the same folder as the Jupyter Notebook because that is how they will be run. You should submit a **single** Notebook (.ipynb) file.