CMPE 150 - Homework 3

deniz.acar1@std.bogazici.edu.tr
yigit.ozel@std.bogazici.edu.tr

Library Management Simulation

You are a new employee in Bogazici University Library and have recently been assigned with taking care of book exchanges. Until now, the library has dealt with borrowing books and later on keeping track of their return date, as it was taken care of by the same person who was very familiar with the job for many years. However, after the resignation of that person, there has been a complication in the next employee's term and this has resulted in the loss of some ancient and precious books, making the university go through a hard time trying to find replacements. Now your boss has decided to switch to a manual system that deals with transactions on its own so that human error is minimized. You have been assigned with designing the program that deals with this system.

Implementation Details:

There will be input files provided to you that you'll need to work on. The input files can have 3 different types of lines.

Input line 1:

The first type is the indicator of a person borrowing or returning a book shown as:

2024-11-18-09-00-00 Ali Borrow B123

2024-11-18-09-05-00 Emre Borrow B234

2024-11-18-09-06-00 Emre Return B342

2024-11-18-09-10-00 Mustafa Borrow B432

These lines (when taken a close look at the first line) indicate the date "2024-11-18" in YYYY-MM-DD format, time "09-00-00" in Hour-Minute-Second format, user_name "Ali", book_status "Borrow", and book_id "B123" in 1 letter 3 numbers.

It is assumed that there are no users with the same name so when someone named Ali borrows a book and the same name is also indicated to return that book, you are to assume that these two are the same person. This will be important for the future parts as you'll be asked to work with the usernames so keep that in mind.

Input line 2:

The second type of input line that you'll encounter is:

Currently missing books.

Every book, given with the book code “B***” exists uniquely in the library, and the book codes start from “B111” and go up to “B444” so in your implementation you should store every value in this scope as a library. The line above asks for the books that are currently not in the library, meaning the books that have been borrowed but not returned. When you encounter this line in the input file, you should write the names of the missing books to the output file in increasing order according to their codes, shown below:

Missing books are: B123, B156, B234, B256, B345

Input line 3:

Third and final type of input line is about which person has borrowed which book. When the line goes as:

Books borrowed by people.

You should show every person that has borrowed a book up until that point and the number of books they’ve borrowed alongside of the book codes. An example output line segment could be:

Mehmet borrowed 4 books: B123, B234, B235, B345

Selin borrowed 3 books: B342, B324, B321

In this part, it’s important to show the names of the people in the order they’ve been introduced to the system. From the output above, you can understand that Mehmet has interacted with the library system (by borrowing or returning a book) before Selin, so keep that in mind.

General form of output lines for book borrowing and returning:

If a book that exists in the library is requested by a person indicated by the word “Borrow”, the output line should be similar to the example:

Mehmet borrowed B234 on 2024/11/18 at 09:23:00

Also, if a book is being returned by the user, the line should be similar to:

Ali returned B342 on 2024/11/19 at 09:24:00

You will have to deal with extra cases as showed under previous titles and alongside those you should deal with the possibility that the book doesn’t currently exist in the library.

A book can be borrowed by only one person, this means that if the book has already been borrowed by a user and there's another person requesting for it, they can't borrow a non-existing book. This case needs to be shown in the output file as well. You can check from below:

Mustafa requested B123 on 2024/11/18 at 09:10:00 - book not available

Ela requested B123 on 2024/11/18 at 09:23:00 - book not available

Also, if a user asks for the same book for more than one times, indicated in the following lines:

2024-11-18-09-05-00 47 Emre Borrow B123

2024-11-18-09-06-00 47 Emre Borrow B123

The output should show all the times that the user has requested for the book (they might ask multiple times to check whether the book has returned or not). If it still hasn't returned, you should print their request again regardless. Also there might be users that accidentally ask for the book that they've borrowed forgetting they did, you should print that accordingly. Check the following 2 examples:

Emre requested B123 on 2024/11/18 at 09:05:00 — book not available

Emre requested B123 on 2024/11/18 at 09:06:00 — book not available

(Ex1. The book doesn't exist prior to Emre asking for it)

OR

Emre requested B123 on 2024/11/18 at 09:05:00

Emre requested B123 on 2024/11/18 at 09:06:00 — book not available

(Ex2. The book is in the library when Emre asks for it but later when he borrows it, it doesn't exist anymore)

IO Handling:

Input file reading and output file creations are dealt with in the code file that's been provided to you. You are supposed to add all the output you've created to the *output_lines* variable and that will be turned into the output file. In your implementation you're required to complete the function named library_system() and then call it for once at the end of your code. You can compare your output files and the correct output files given in the material using the terminal in your pc.

The creation of output files is done in the code given to you and you need to give the input and output file paths to the INPUT_FILE and OUTPUT_FILE variables which you can check in the code yourself. For testing different input files, you can write different input file paths for your program to read. (Writing the paths for Windows and macOS are different so check that carefully.)

For windows, file comparison can be made with command:

```
fc <your_output_file> <file_we_provided>
```

For mac, file comparison can be made with the command:

```
diff <your_output_file> <file_we_provided>
```

You should write the full paths for the files for the terminal to be able to read them.

Submission Details and Penalties:

- You are expected to submit one .py file under the name of “your_student_number.py”.
- Mistakes in submission will result in penalties so carefully check your file before sending it.
- You are supposed to use the structures you’ve learned in the lecture until this point and NO imports or builtin libraries are allowed. Usage of forbidden structures will give you penalties.
- Don’t use large language models to write your full code, and don’t share code with your peers. Plagiarism and copying will be harshly evaluated and possibly taken disciplinary actions against, so just try to write the code yourself.

Updates and new files (if necessary) will be shared in the moodle page so don’t forget to constantly check it.