Assignment 1 – Library Reading Club

Due Date: Wednesday, October 14th 2020 at 23:55 (Edmonton time)
Percentage overall grade: 6%
Penalties: No late assignments allowed
Maximum marks: 100

Goal: refresher of Python and hands-on experience with file input/output, built-in data structures (especially dictionaries), and string manipulation.

Assignment Problem

Every summer, Edmonton Public Library (EPL) has a Summer Reading Club that aims to encourage children and tweens to read more during the summer months. Every book that a participant reads is recorded; the more books that a participant reads, the higher the level of that participant, and the more prizes s/he qualifies for.

EPL has hired you to write a Python program that will allow librarians to:

- enter the books read by each participant;
- generate a report for a given participant that summarizes how many books they have read and what level they are currently on; and
- see a summary of the club activity so far.

Input

EPL will provide you with **THREE** text files. The content of these files will be updated periodically, so your program must be able to read the most up-to-date information from them every time it is run.

The first text file is called *clubLevels.txt*, and it contains information about different themes for the club. Each summer, EPL chooses a theme based on information in this file. For example, one year the theme may be all about dinosaurs, and another year it may be all about exploring the safari. This file also contains information about level names that correspond to a specific theme. Each line of the text file will contain the theme name, level name, and maximum number of books read associated with that level. All of these items are separated by a colon (":"), as can be seen in the sample clubLevels.txt provided with the assignment description. You can assume that any participant who reads more than the maximum number of books associated with the highest level will be put into that highest level. For example, in the sample clubLevels.txt provided, if Safari Summer is selected as the theme, anyone who reads 17, 18, or more books will achieve "lion" status.

The second text file is called *participants.txt*, and it contains information about all of the people currently registered in the reading club. Each line of the text file will contain:

- the participant’s first name;
- the participant’s last name;

- the participant's library card number (a 7 digit sequence); and
- the participant's birthday. The birthday will be formatted to start with the first three letters of the birthday month, followed by a single space, followed by the day integer which could contain either one or two digits (e.g. 3 or 27), followed by a single space, followed by the four digit birth year. You can assume that any birthdays entered in this file are valid and correct.

All of these items will be separated by a comma and single space (", "), as can be seen in the sample participants.txt provided with the assignment description.

The third text file is called *booksRead.txt*, and it contains information about all of the books read so far by each participant. Each line of the text file will contain the library branch name where the book was loaned, the reader's library card number, and the name of the book. All of these items will be separated by a hash symbol ("#"), as can be seen in the sample booksRead.txt provided with the assignment description. You can assume that any library cards recorded in this file belong to a participant from participants.txt. However, you cannot assume that the lines in this file are in any sort of order.

Output

Initialization:
When the program first starts, it should display a welcome banner, followed by a prompt asking the librarian to enter in the club's start date. It should look exactly like this:

```
************************************
WELCOME TO EPL's SUMMER READING CLUB
************************************
Please enter the club's start date (YYYYMMDD):
```

The program should check that a valid date is entered in the format YYYYMMDD. In other words, the date should contain 8 digits (no letters or other characters), the month should be from 01 to 12, and the day should not exceed the number of days for the entered month. Remember to check if the entered year is a leap year when checking for the maximum number of days in February. Continue to re-prompt until a valid date is entered. (See **sampleOutput.txt** included with this assignment description for more details on format of re-prompt.)

*Reminder: a leap year occurs when the year is an integer multiple of 4, except for years that are also divisible by 100 but not 400. For example, the year 2000 was a leap year because it was divisible by 4 and 400. The year 2004 was also a leap year because it was divisible by 4 and not 100. However, the year 1900 was NOT a leap year (divisible by 4 and 100, but not 400). The year 1901 was NOT a leap year because it was not divisible by 4.

Once a valid start date has been entered, the librarian should be prompted to select a theme. These themes must be read in from clubLevels.txt and displayed on the screen in alphabetical order. (See sampleOutput.txt.) If the librarian enters something that does not exactly match any of the themes displayed, they should be re-prompted until they enter a valid theme.

Menu:
After the initialization has been completed, a menu should be displayed that will allow the librarian to keep selecting options until quit is chosen. It should look exactly like this:

```
What would you like to do?
1. Record a book that has been read.
2. Generate a participant report.
3. Summarize club activity.
4. Quit
>
```

The only valid options for the librarian to enter are the numbers 1, 2, 3, or 4. If the user enters an invalid choice, your program should display an error message and then re-prompt the librarian to enter a valid choice. It should continue to re-prompt until a valid choice is entered. For example:

```
What would you like to do?
1. Record a book that has been read.
2. Generate a participant report.
3. Summarize club activity.
4. Quit
> 0
Sorry, invalid entry. Please enter a choice from 1 to 4.
> abc
Sorry, invalid entry. Please enter a choice from 1 to 4.
>
```

Option 1:
When the librarian chooses option 1, s/he will be prompted to enter all of the required information pertaining to the book that has been read and the participant that read it. The librarian will first be prompted to enter the reader's library card number. Your program should check to make sure that the entered card number matches that of a registered club participant (i.e. it exists in participants.txt), and continue to re-prompt until a valid registered library card number is entered. The librarian will then be prompted to enter the title of the book (no validation required). Finally, the librarian will be prompted to enter their library branch (no validation required). See sampleOutput.txt for formatting and wording.

Using the information that the librarian has entered, your program should add a new line to the booksRead.txt file. Follow the established format of that file: library_branch#library_card#book_title Be sure that you keep all of the data that was already in booksRead.txt (i.e. do not overwrite), and also be sure not to include any empty lines in your file. Display "`Record added successfully.`" on the screen once booksRead.txt has been updated.

After booksRead.txt has successfully been updated, the main menu is displayed again for the librarian to make another choice.

Option 2:
When the librarian chooses option 2, s/he will be prompted to enter the library card number of a registered participant. If an invalid library card number is entered (i.e. not that of a registered participant), an error message is displayed and the main menu is displayed again for the librarian to make another choice. See sampleOutput.txt.

If a valid library card number is entered, a report for that participant is displayed on the screen. See sampleOutput.txt. The width of each line in the report must be 25 characters wide.

- The name of the participant should be right-aligned, and should consist of the participant's first name (all in capital letters), followed by a single space, followed by the first character of the participant's last name (capitalized). If the participant's first name is longer than 8 characters, only the first 7 characters of that first name should be displayed, followed by an asterisk ("*").
- The age of the participant should be right-aligned, and calculated to show their age as of the start of the summer reading club (entered during the initialization phase).
- All books read by the participant so far should be listed (in the order that they appear in booksRead.txt). The displayed book title should not exceed the width of the report. If it does, the title should be truncated to only show the first 23 characters, followed by an asterisk ("*"). If the participant has not yet read any books, "`None yet...`" should be displayed.
- The total number of books read should be right-aligned. You can assume that no one will read more than 999 books in one summer.
- The participant's level (corresponding to the selected theme) should be right-aligned and in all capital letters.

After the participant's report is displayed, the main menu is displayed again for the librarian to make another choice.

Option 3:
When the librarian chooses option 3, a table summarizing the club's activity to date will be displayed on the screen. It will have three columns, and each of the column titles are centered. The table is 47 characters wide.

- Column 1 will be the age group. There are three age categories: 5 and under, 6 to 9, and 10 to 13. You can assume that individuals older than 13 cannot register to be part of this particular reading club. (Instead, the library has other reading clubs for older library patrons that run all year long.) The contents of column 1 (except for its title) should be right-aligned.
- Column 2 will be the total number of books read by all participants belonging to the respective age group. Remember that a participant's age is calculated as their age on the club's start date – participants will not move to a different age group during the summer. You can safely assume that the total number of books read will not exceed 1 billion. The contents of column 2 (including its title) should be centered.

- Column 3 will be the name of the participant who has read the most books for their age category. Their name will be displayed as in the participant's report (option 2). The contents of column 3 (except for its title) should be left-aligned. If there is a tie for the participant who has read the most for their age category (and that number of books read is greater than 0), display `"Tied!"` If no one in that age category has read any books yet, display '`N/A`'.

See sampleOutput.txt – pay close attention to the table format; you're formatting must match exactly.

After the summary has been displayed, the main menu is displayed again for the librarian to make another choice.

Note that the librarian can select the first 3 options in any order, and as many times as desired.

Option 4:
When the librarian chooses option 4, a goodbye message is displayed and the program will end:

```
What would you like to do?
1. Record a book that has been read.
2. Generate a participant report.
3. Summarize club activity.
4. Quit
> 4
Goodbye
```

Testing
Use the sample input text files to test your code. However, keep in mind that the markers will test your code with DIFFERENT input data in those input text files. So you should also test your code by changing the content of clubLevels.txt, participants.txt, and booksRead.txt.

Assessment
In addition to making sure that your code runs properly, we will also check that you follow good programming practices. For example, divide the problem into smaller sub-problems, and write functions to solve those sub-problems so that each function has a single purpose; use the most appropriate data structures for your algorithm (dictionaries will be your friend!); use concise but descriptive variable names; define constants instead of hardcoding literal values throughout your code; include meaningful comments and docstrings to document your code; and be sure to acknowledge any collaborators/references in a header comment at the top of your Python file.

Restrictions for this assignment: you cannot use break/continue, and you cannot import any modules. Doing so will result in deductions.

Rubric

Code quality and adherence to specification: 20%
File handling (input) and use of appropriate data structures: 10%
Program initialization and flow control: 10%
Option 1: 10%
Option 2: 25%
Option 3: 25%

Submission Instructions

Please follow these instructions to correctly submit your solution:

- All of your code should be contained in a single Python file: **assignment1.py**.
- Make sure that you include your name (as author) in a header comment at the top of assignment1.py, along with an acknowledgement of any collaborators/references.
- Please submit your assignment1.py file via eClass before the due date/time.
- Do not include any other files in your submission.
- Note that late submissions ***will not be accepted***. You can make as many submissions as you would like before the deadline – only your last submission will be marked. So submit early, and submit often.