CMPUT 175 LEC B1 B2 B3 B4 - Winter 2022

Dashboard / My courses / CMPUT 175 (LEC B1 B2 B3 B4 Winter 2022) / Assignments / Assignment 1

Assignment 1

Assignment 1 - File Input/Output and Data processing

Due Date: Monday January 31st, 2022 at 23:55

Percentage overall grade: 6%
Penalties: No late assignments allowed
Maximum Marks: 100

Goal: refresher of Python and hands-on experience with file input/output, string manipulation.

Assignment Specifications

Problem: Data Processing

An acquaintance of yours, a librarian at an elementary school in a rural area who is still maintaining the library holdings manually on paper, has contacted you for assistance. They have acquired a computer and knowing that you are taking CMPUT 175, they have asked you to help write a program to automate some of the processes. Using a spreadsheet, they have already started collecting data that they can provide you in text-formatted files. Your python program would use these text-formatted files as input and produce as output lists that can be printed.

The files that you have as input are as follows:

The first file, **books.txt**, contains the list of all the books in the library. Each line of the file contains information about a book: Book ID followed by a pound (#), followed by a title, followed by a pound, followed by the author names, followed again by a pound and finally followed by a value in dollars. For example, the content of the file could look like this:

```
B123456#The mouse and the motorcycle#Beverly Cleary#10.5
B234587#Charlotte’s Web#Elwyn Brooks White#15
B354759#Matilda#Roald Dahl#15
B654190#Harry Potter and the prisoner of Azkaban#Joanne Rowling#25
B58256#James and the giant peach#Roald Dahl#13
B246511#The witches#Roald Dahl#22
B998135#Pipi Longstocking#Astrid Lindgren#8.75
B100952#The battle of the labyrinth#Rick Riordan#25
```

The second file, **students.txt**, contains the list of students. Each line concerns one student and is formatted as follows: StudentID, student name, class, all separated by a comma (,). For example, the content of the file could look like this:

```
S983342,Suzanne Foster,2B
S981123,Eric Xu,2B
S980127,Hua Li,3A
S986654,Myriam Salah,2A
S982234,Ali Rabbany,3B
S987654,Kevin Kelly,3A
```

The third file, **borrowers.txt**, contains the history of all loans. Each line has a Book ID followed by a Student ID, followed by a date the book was borrowed, followed by the expected date the book is supposed to be returned. The dates are formatted in strings of 6 characters, the year followed by the month followed by the day. For instance the 5th of January 2022 would be “220105”. This way dates can be compared. Indeed 220105 < 220131. All fields are separated by a semi column (;). For example the content of the borrowers.txt content could be like this:

```
B123456;S983342;220105;220201
B234587;S983342;220110;220215
```

A fourth file, **returns.txt**, contains information about the returns. Each line has a Book ID followed by a Student ID, followed by a date the book was registered as returned, followed by the state of the book. The date is formatted as in the previous file YYMMDD and the state of the book is either “0” meaning the book is in good state, “1” if the book was lost, “2” if the book was lost and paid, “3” if the book was damaged and paid, otherwise it means the book was damaged and needs replacement but hasn’t been paid yet. All fields are separated by a semi column (;). Here is an example of how the file could look like:

```
B123456;S983342;220201;0
B234587;S983342;220215;2
```

For all four input files, you can assume that there will not be whitespace at the beginning of each line, nor will there be blank lines in the file. You can also assume that all Book IDs used in borrowers.txt and returns.txt are defined in books.txt and all Student IDs in borrowers.txt and returns.txt are defined in students.txt.

Your Python program must generate a file, **standing.txt**, that contains for each classroom two tables. The first table is listing all of the books that haven’t been returned yet and the name of the borrower sorted by the borrowers name, with the due date. This table should only contain the books that haven't been returned. this information can be obtained from borrowers.txt and returns.txt. The due date is the date the book is supposed to be returned. The second table, also sorted by the students’ names, contains the amount due by students. The student owes the cost of the book (from the books.txt file) if the book was lost (i.e. the state of the book is 2 in the returns.txt file) or was damaged but not paid back (i.e. status is not 0, 1, or 3 in the returns.txt file).

Both tables, as shown below, end with a total. In one case, it is the count of books borrowed by students in the classroom, and in the second case the total of money due by all students in the classroom.

```
Class: 2B
+------------------+--------------------------------------+--------------+
| Student Name     | Book                                 | Due Date     |
+------------------+--------------------------------------+--------------+
| Eric Xu          | Harry Potter and the prisoner of Az  | Feb 15, 2022 |
| Suzanne Foster   | Pipi Longstocking                    | Feb 15, 2022 |
| Suzanne Foster   | National Geographic Little Kids Fir  | Feb 27, 2022 |
 ...
| Zac de wall      | The Kids book of Canada              | Feb 10, 2022 |
+------------------+--------------------------------------+--------------+
| Total Books                                             |           35 |
+---------------------------------------------------------+--------------+

+------------------+----------+
| Student Name     | Due      |
+------------------+----------+
| Bill Bryson      |   $15.00 |
| Suzanne Foster   |    $5.75 |
...
| Zac de Wall      |   $20.50 |
+------------------+----------+
| Total Books      |  $145.75 |
+------------------+----------+

Class: 3A
```

```
...
```

The first table for each class has 3 columns:

- The first column is the student name. The name is padded with one space in the start and as many spaces as necessary after the name. The name is truncated after the 16th character.
- The second column is the title of the borrowed book truncated after the 35th character. So the column has to be 37 characters wide. That is, If the book title is longer than 35 characters, the title is truncated to fit.
- The third column is the due date. Notice that the due date format is converted to mmm dd, yyyy with the month in letters Jan, Feb, Mar, etc.
- Finally, a line at the end of the first table should indicate the total number of borrowed books by the pupils of the class. The table should be formatted as illustrated above.

The second table for each class has 2 columns:

- The first column is the student name. The name is padded with one space in the start and as many spaces as necessary after the name. The name is truncated after the 16th character.
- The second column is 10 positions wide and contains the total amount due by that student for all books returned damaged or lost and that are unpaid. The amount should be formatted with 2 digits after the decimal point.
- Finally, a line at the end of the second table should indicate the total amount still due by the pupils of that class.

In addition to the output file, the same output should also be printed on the screen without the list of students but only the totals and should look like:

```
Class: 2B
Total books currently borrowed: 35
Total amout due for books: $145.75

Class: 3A
...
```

Note that the example output table is not exactly related to the input examples provided here. Indeed, the output depends on 4 input files. The input and sample output provided here are just examples to indicate the format of all of the files. **You should create your own files and stick to the structure described here.** The TAs will evaluate your assignment with different input files.

Stick to what the customer has requested as format. Do not try to make it "nicer" or "more practical", or anything else. This is what the customer wants.

General Guidelines

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; use concise but descriptive variable names; define constants instead of hardcoding literal values throughout your code; include meaningful comments to document your code, as well as docstrings for all functions; 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.

Assignment Deliverables

- You are going to submit one Python program that performs the specified tasks.

Submission Instructions

Please follow these instructions to correctly submit your solution.

- Name your solution **assignment1.py**
- Submit your program at the end of this page
- Note that late submissions ***will not be accepted.*** You are able to submit your work as often as you like and only the last submission will be marked. So submit early and submit often.

books.txt ✚	14 January 2022, 6:47 PM
borrowers.txt ✚	14 January 2022, 6:47 PM
returns.txt ✚	14 January 2022, 6:47 PM
students.txt ✚	14 January 2022, 6:47 PM

Submission status

Attempt number	This is attempt 1.
Submission status	No attempt
Grading status	Not graded
Due date	Monday, 31 January 2022, 11:55 PM
Time remaining	14 days 12 hours
Last modified	-
Submission comments	▸ Comments (0)

Add submission

You have not made a submission yet.