Lab Assignment 2:
Library Catalog System

Due Date (a two-week LA)	
Wednesday Labs	**2/12/20 @ 11:59pm**
Thursday Lab	**2/13/20 @ 11:59pm**

Objectives

- Review classes and OOP
- Review inheritance and polymorphism
- Review overriding superclass method(s) in subclasses

Problem Specification

Write a Python application to keep track of a library's collection of materials. The collection will be read from an **input** file; users should be able to display a list of materials and check out materials using the materials' call numbers.

The library materials are either books or periodicals. The information stored for each type of library material is shown below.

- Books (they are checked out for 21 days at a time)
  - Book Title
  - Author
  - Genre
- Periodicals (they are checked out for 7 days at a time)
  - Periodical Title
  - Volume
  - Issue
  - Subject

All library materials have a call number, and can be checked out. The date checked out should be the current date, and the due date depends on the type of material (21 days after current date for **books**; 7 days for **periodicals**). When displaying the item, show the specific information, along with the call number, whether or not the item is checked out, and if checked out, the check-out and due dates (see example output below).

The input file with data about available library materials is organized as follows:

- The first line is the number of **books** in the library's collection.

- The second line is the number of **periodicals** in the library's collection.
- Each line after that contains a single library item (fields separated by commas)
    - Books (indicated by the letter B)
        - Fields included: *Call number, Book title, Author, Genre*
    - Periodicals (indicated by the letter P)
        - Fields included: *Call number, Periodical title, Volume, Issue, Subject*

Example Input:

2
2
B,C124.S17,The Cat in the Hat,Dr. Seuss,Children's Literature
P,QJ072.C23.37.4,Computational Linguistics,37,4,Computational Linguistics
P,QJ015.C42.55.2,Communications of the ACM,55,2,Computer Science
B,F380.M1,A Game of Thrones,George R. R. Martin,Fantasy Literature

The application should exhibit the following functionality (see the sample output below):

- Read the contents of the library's collection from the input file.
    - The file name should be **hardcoded** in the application.
- Allow the user to choose from a menu of 3 options:
    - ***Display collection***
        - Display the full list of library materials including the **check-out status** (i.e. YES or NO);
            - If checked out, display the check-out date and the due date.
    - ***Check out materials***
        - Ask the user for the call number, and then find the matching item.
        - If the item is already checked out, do not allow the user to check it out; display a message indicating it is not available.
            - Display the menu again to the user.
        - If the item is **not** checked out, check the material out to the user and set the **check-out** date and **due date** accordingly.
        - Display the item information
    - ***Quit***
        - Exit the application
- Allow the user to continue making requests (selecting options) until s/he selects the ***Quit*** option.

Example Output:

---------- Menu ----------
1) Display collection
2) Check out materials
3) Quit

Please choose an option: 1

Book Title: The Cat in the Hat
Author: Dr. Seuss
Genre: Children's Literature
Call Number: C124.S17

Checked Out: NO

Periodical Title: Computational Linguistics
Volume: 37
Issue: 4
Subject: Computational Linguistics
Call Number: QJ072.C23.37.4
Checked Out: NO

Periodical Title: Communications of the ACM
Volume: 55
Issue: 2
Subject: Computer Science
Call Number: QJ015.C42.55.2
Checked Out: NO

Book Title: A Game of Thrones
Author: George R. R. Martin
Genre: Fantasy Literature
Call Number: F380.M1
Checked Out: NO

---------- Menu ----------
1) Display collection
2) Check out materials
3) Quit

Please choose an option: 2

Enter the call number: F380.M1

Book Title: A Game of Thrones
Author: George R. R. Martin
Genre: Fantasy Literature
Call Number: F380.M1
Checked Out: YES
Date Out: 02/08/16
Date Due: 02/29/16

---------- Menu ----------
1) Display collection
2) Check out materials
3) Quit

Please choose an option: 2

Enter the call number: F380.M1

```
Item is not available.
---------- Menu ----------
   1) Display collection
   2) Check out materials
   3) Quit
------------------------------
Please choose an option: 3
```

Design Requirements

Your application **MUST** make use of a proper **inheritance** relationship. It should have a **superclass** representing a general library item, and **subclasses** representing the specific types of library items included in the library collection (**books** and **periodicals**).

Specifically, your project should have a class **LibraryItem** with the methods listed below.

class LibraryItem:

```
    # Initializer method – Updates the call number, checkedOut, dateCheckedOut and dueDate fields.
    __init__(self, call_num)

    # Sets the boolean value checkedOut to true, and
    # initializes the dateChecked out attribute (with
    # the current date – date on which item was checked out).
    check_out()

    # Getter for the call number of this object.
    # @return - the callNumber
    get_call_number()

    # Returns true or false depending on if this item has been checked out.
    # @return - the boolean value for isCheckedOut
    is_checked_out()

    # Returns the date this item was checked out.
    # @return - the date the item was checked out
    get_date_checked_out()

    # Returns the date this item is due to be returned.
    # @return - the due date
    get_date_due()

    # Sets the dateDue to the parameter received.
    # @parameter dateDue - the new due date
    set_date_due(self, due_date)

    # Generates a string with the details of the library item
    # whose call number has been input by the user (see example output)
    # and returns that string.
    # If the user wants to check out the library item, the string to be
    # returned also includes information that the item has been checked out,
```

```
    # the date it was checked out, and the due date by which the item should be returned.
    # @return – The generated string.
    __str__();
```

Class **LibraryItem** has two subclasses - **Book** and **Periodical**. Your application should store all library items (both Books and Periodicals) in a list.

There is also a **Controller** class which is used by the **main**() method to run the program.

The code for the ***main()*** method) is provided and **should not be modified**. This serves as the test class for your program.

NOTE: ***Any fields or methods common to the subclasses should be defined in the superclass. Specific functionality, such as additional data members or overriding methods, should be defined in the appropriate subclass(es).***

The methods in class Controller, and the code for the main method are provided below.

class Controller:-

```
    # Initializer method: - creates an empty list of library items. This will be used to store the library items in the input
    # file.
    __init__()

    # Displays the menu options to the user.
      show_menu()

    # Displays the collection of library items on the screen
    display_collection()

    # Searches in both the array of books and the array of periodicals
    # for the book with the call number received as a parameter.
    # @parameter callNum - The call number of the item requested by the user (of type “string”)
    # @return - The requested item (if found)
    find_item(callNum)

    # Requests for the call number from the user, uses the findItem()
    # method to check if that item exists in the library, and if it does
    # calls the checkOut() method for that item and prints out the item
    # that has been checked out.
    check_out_materials()

    # Reads data from the input file and stores the items in the
    # appropriate array.
    # @parameter filename - The name of the input file (of type “string”)
    read_input (fileName)

# End of class Controller
```

```
def main():
    control = Controller()
    control.read_input("input.txt")
    response = ""
    quit_flag = False
    print('Before while-loop')
    while quit_flag is False:
        control.show_menu()
        response = input("Please choose an option: ")

        if response == "1":
            control.display_collection()
        elif response == "2":
            control.check_out_materials()
        elif response == "3":
            quit_flag = True
        else:
            print("Invalid response!")

    print("Good bye!")
```

End of **main**()

Hints

1) Working with dates in Python is very easy! Visit the links below for examples of how to get the current date, and how to get the new date after adding a specific number of days to the current date (this will be useful for getting the due date for a library item that is checked out).

   https://www.codespeedy.com/how-to-add-days-to-date-in-python/
   https://www.programiz.com/python-programming/datetime/current-datetime

2) When reading from the input file, it may be helpful to strip the newline character from each line of input. Also, to "process" the comma-separated data in the input file, you can split each line read from the file, using the comma as the **delimiter**.

3) Recall that the three steps used for working with input files are: open file, read data, close file.

Additional Requirements

A proper *design* (with detailed pseudocode) and proper *testing* are essential.

*Note: **Correct pseudocode development** will be worth **40%** of the total LA grade.*

You will also need to install **SPHINX** and use it to **generate HTML documentation** for your projects in PyCharm. This will be covered in the Labs. Accurately following the steps shown in class (a copy of which will be made

available in Elearning) will result in the creation of a file "**index.html**" for your project; this file can be opened in a browser and will contain your project's documentation.

Coding Standards

You must adhere to all conventions applicable to writing programs. This includes the use of white spaces and indentations for readability, the use of comments to explain the meaning of various methods and attributes, and the conventions for naming classes, variables, method parameters and methods.

Assignment Submission

- Generate a .zip file that contains all your files including:
    - Program Files
    - Any input or output files
    - The document containing your pseudocode

- Submit the .zip file to the appropriate folder on ELearning.

NOTE: The Elearning folder for LA submission will remain open beyond the due date but will indicate how many days late an assignment was submitted where applicable. The dropbox will be inaccessible seven days after the due date by which time no more credit can be received for the assignment.

The penalty for late submissions as stated in the course syllabus will be applied in grading any assignment submitted late.