Project 2 – Programming Project

CS3330 Data Structures and Algorithms
Term 5 2016: May 23 – July 24
Dr. Jack Davault

Overview

This assignment consists of implementing an application using the techniques learned in the first half of the course. Examples on how to use file operations, the standard template library, and other features are provided in the *Sample Programs (for Project 2)* section located the *Learning Activities* area in Blackboard. These examples provide a starting point for your implementation.

Your program must compile and you must provide all of the source code files so that I can also compile and run your program (i.e. provide me with all files ending in **.h** and **.cpp** that are necessary to compile your program). You are responsible for submitting your program correctly. Please ask if you have questions.

You may use any resources, books, or notes. **Apart from your textbook and provided examples, you must mention all resources that you use as comments next to the applicable lines of code or the appropriate functions within your source code or within the main comment at the top of your source code file.** You may also work with only <u>one</u> other classmate on this assignment if you want to, but you must mention all of your names as comments in the applicable sections of the source code showing where you collaborated. If you collaborate on the entire project, no problem, both of your names must be in the comment at the top of all of the source code files, and only one person needs to submit the assignment. **I do highly recommend that you work with one other person on this project, and begin working on this assignment as soon as possible. Don't wait until the last minute to begin working on this assignment.** Consider posting a message in the *Student Lounge* or sending out an email to your fellow classmates asking if anyone is willing to partner with you on this project. Two person teams only.

Note: Again, you may use outside resources or books, but you are not to post questions, comments, or source code related to this assignment in any public open forum other than the ones specified for this course in Blackboard (e.g. *Ask the Instructor* or *Student Lounge*). Also, note that pay for websites for programming solutions are not an acceptable method for completing this assignment. If you have a question, don't sit on it for too long; ask me in the *Ask the Instructor* forum or via e-mail. I am also available by appointment via *Blackboard IM*.

This assignment weighs 15% of your final course grade. When submitting your work, be sure to Zip up all source code files and documentation (if any) into a single file.

Implementation (25 points)

Implement a streaming data servicer management program can be used as a prototype to schedule the availability of media for a streaming data service provider. Your program will allow the scheduler (or user) to enter the properties associated with various types of streaming media (e.g. movies, music, etc.). The program will manage a list of properties for each streaming data product as a node within a single linked list stored in memory (this can be a global variable for simplicity). The program <u>must</u> use the "list" API in the C++ standard template library (STL). The program must implement at least one class that will hold the following variables (input validation is only required where indicated):

• `entryID –`	A string variable that will hold an automatically generated 10-character string as the entry identifier. The string must be in the form **LLL-NNN-LL**, where **L** is a character from A to Z, and **N** is a number from 0 to 9. For example: HEJ-268-UY. The `entryID` is a unique identifier for each streaming product and repeats are not allowed.
• `productType –`	A string variable that indicate the type of streaming content. Valid content items are: movie, music, television, news, and radio. *Input Validation Required:* The program must re-prompt the user if the entered product type does not match one of the valid content items.
• `title –`	A string variable to hold the title of the streaming entry. Note that spaces between words must be allowed so that all of this information can be entered.
• `description -`	A string variable to hold a brief description of the streaming entry. Again that spaces between words must be allowed so that all of this information can be entered.
• `durationTime -`	A string variable to hold the playing time for the streaming data product. The provided time must be in the form **hh:mm:ss**, where **hh** is the number of hours (00 - 12) and **mm** is the number of minutes (00 – 60), and **ss** is the number of second (00 – 60). *Input Validation Required:* The program must re-prompt the user if the entered duration time does not match the above format.
• `rentalPrice –`	A float variable to hold the price of the streaming product.
• `dateAvailable -`	A string variable to hold the date that the media will be available. The entered date will be in the form **mm/dd/yyyy**.
• `daysAvailable -`	An integer variable to hold the number of days that the media will be available.

Provide the appropriate methods to set and get the data for each of these class variables. For example `setTitle(string title)` and `string getTitle()`. The BookInventory provides an example on how this can be done. In addition, the main program must provide the following functionality:

1. When the program is first started, it must read a data file called **media_list.dat**. The program will not prompt the user for the name of the data file. The name of the file must be hard-coded in the program. If the file exists, the program will load the data for each media type into the global linked list. If the file does not exist, the program will start with an empty linked list.

2. The program will provide a simple text-based user interface that manages the all of the schedule media within a linked list. Each entry must be placed in linked list as an object that holds all of the information associated with it as mentioned above. The user interface will allow the user to perform the following:

    (a) Display Content – displays all of the scheduled media within the linked list along with their associated attributes. In addition, this option must print the total number of scheduled entries within the linked list and the total rental prices for all of the scheduled.

(b) Enter Media Item – allows the user to enter all of the fields associated with a given media item, except for the `entryID`, which will be automatically generated by the program as previously mentioned. After the fields are entered, the program will place the media object into the global linked list.

(c) Delete Media Item – allows the user to delete a media item from the linked list using the `entryID` as the key. The program must display a message if the provided `entryID` is not found in the linked list.

(d) Search for Media Item – allows the user to find a media item. The program will prompt the user to enter the `entryID` and will display all of the fields associated with the given media item if it is found. The program must display a message if the item is not found in the linked list.

(e) Edit Media Item – allows the user to edit the fields for a given media item that is in the linked list (except for the `entryID`). The program must prompt the user to enter the `entryID` as the key to find the item to edit. Print a message if the media item is not found in the linked list. For simplicity, the program may re-prompt the user to re-enter all of the fields associated with the given media item; but again, it must reuse the `entryID` value.

(f) Exit System – before the program exits, it must save all of the data in the linked list to the data file. I recommend using a standard text file with one field per line. At this point, if the file does not exist, the program will create it.

Hints and Tips

The Book Inventory program provides a good place to start and I recommend using it as a model. It provides a simple text based interface, shows how to create a class, and how to use the C++ STL library's linked list API. You can rename its files, the class, and methods in the as a starting point for your program.

Start by breaking the program down into small pieces. First, work on the feature that allows the user to enter a property along with all of its fields. Next, work on the display feature that will show all of the items and their associated fields contained within the linked list. Next, make sure your program can read and write one item and its fields to and from the data file. Then expand the program to read and write all of the items to and from the data file. The data file functions for this can be modeled after those in File Operations example and the display and enter functions that you will create. Finally, work on the search, modify, and delete features. Be sure to review the sample code in the *Sample Programs (for Project 2)* section in the *Learning Activities* area in Blackboard.

There are no tricks or special techniques required for this assignment. Everything you need to successfully create this program is available in the provided sample code.

Other Comments

Please do not hesitate to let me know if you have questions in the *Ask the Instructor* forum or via e-mail. I also encourage you to discuss this project among yourselves in the *Student Lounge* area.