

ITSC 1213 – Project 3 – Bookstore Management System

In this checkpoint, you will be expanding the previous assignment in two significant ways. This assumes you have a working version of the Bookstore project from the previous assignment. If you were not able to complete that assignment, or if your project had significant issues that affected functionality, please make sure to get in touch with someone on the instructional team to help get your project caught up so you are ready for this one.

Preliminaries

I can not iterate the importance of having a working implementation that meets the requirements of programming checkpoint 2.

You do not need to start a new NetBeans project for this submission. You can use your existing Bookstore project as a starting point. If you choose to create a new project it should be named **lastNameFirstNameProject3**.

Part A – Inventory Tracking (15 points)

In this part we want to change the way our inventory tracking works. Likely, you have some sort of starting inventory hard coded into your Bookstore class, but this creates a situation where every day of your bookstore's operation is completely identical (i.e., every time we run the simulation we get the same output). In order to make this both more interesting and realistic, in this section you will be altering the execution of your bookstore such that rather than generating a fixed inventory when run, the inventory will be created by parsing through an external file at execution time.

A simple way used to store tabular data, such as a spreadsheet or database, in a text file is using a format known as Comma Separated Values (CSV). In this format the comma in this format is a delimiter, which is a character that separates the columns of a table. Using the comma character to separate (or delimit) data is the most common, but sometimes we use other characters, like semicolons. This is important if the actual data contains commas since that will interfere with the file structure.

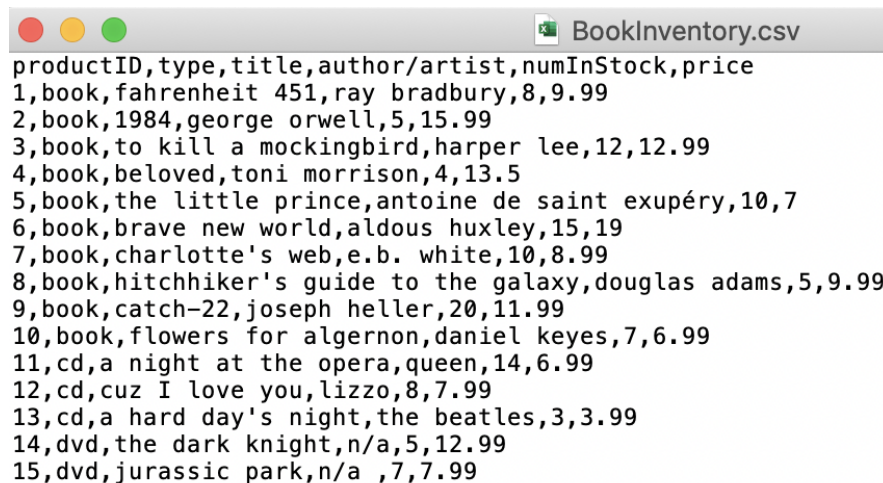
For example, let's say you had a spreadsheet containing the following data we can create a CSV file using a text editor such as Notepad or TextEdit.

Email	Name	Hometown
John@company.com	John Newman	Chicago
Marie@company.com	Marie Robinson	San Francisco
Cali@company.com	Cali Williams	Seattle

The contents of the text file would look like:

```
Email, Name, Hometown
John@company.com, John Newman, Chicago
Marie@company.com, Marie Robinson, San Francisco
Cali@company.com, Cali Williams, Seattle
```

Take a look at the format and content of this file



```
productID,type,title,author/artist,numInStock,price
1,book,fahrenheit 451,ray bradbury,8,9.99
2,book,1984,george orwell,5,15.99
3,book,to kill a mockingbird,harper lee,12,12.99
4,book,beloved,toni morrison,4,13.5
5,book,the little prince,antoine de saint exupéry,10,7
6,book,brave new world,aldous huxley,15,19
7,book,charlotte's web,e.b. white,10,8.99
8,book,hitchhiker's guide to the galaxy,douglas adams,5,9.99
9,book,catch-22,joseph heller,20,11.99
10,book,flowers for algernon,daniel keyes,7,6.99
11,cd,a night at the opera,queen,14,6.99
12,cd,cuz I love you,lizzo,8,7.99
13,cd,a hard day's night,the beatles,3,3.99
14,dvd,the dark knight,n/a,5,12.99
15,dvd,jurassic park,n/a ,7,7.99
```

The file format of this file is .csv. This is an example of an inventory file that can be used to populate the inventory of our bookstore by using the relevant information to create the different Product objects. You can use this file or create one with information relevant to your solution and products. ****NOTE**** - If you create your own .csv inventory list keep in mind that you will need to submit that along with the rest of your assignment files.

Once your inventory is populated, the program should proceed as before. When the day ends (i.e., user selects exit), the following should happen:

- An end of day report should be generated as a .txt file.
 - This file should include what products were purchased, how many new members were registered, total sales and revenue, and any other information you think is relevant.
 - It is up to you to figure out the proper formatting to this file. It should be clear, readable, and make sense.
- An updated inventory file. This file should follow the same format of the input file , but have updated stock numbers. This file should have a slightly different name, such as BookInventoryDay2.csv.

Point distribution - inventory loaded from file (5 points), end of day report (5 points), updated inventory file (5 points)

Part B – Changing the Test Harness (5 points)

The other change we'll be making to the project is a potentially small alteration to the test harness code. Our goal with these changes is to alter the flow of the test harness such that the execution of your program represents one day of our bookstore being open. The most important aspect of this is the idea of a starting inventory (the products available for purchase at the start of the day) and an ending inventory (purchased products subtracted from the inventory).

Update the test harness to reflect the updates made to the inventory tracking such that before the program exits the end of day report and updated inventory file are generated.

Part C – Handle Exceptions (7 points)

There is a lot of user input using the scanner class in this project. Usage of the scanner class can lead to lots of different exceptions being thrown (i.e., What if the user enters a letter when you are expecting a number? or the other way around?). There is also the use of the File I/O classes and methods when the read from and write to files. You should catch these exceptions in the program and write out an error message.

1. Catch InputMismatchException when you ask the user for a number using nextInt() or nextDouble().
2. Catch general Exception in a try block around all scanner input calls, as a way to check for the many possible errors that can occur when asking the user for input.
3. Catch FileNotFoundException when you have open/close file operations
4. Catch IOException

Suggested Additions for Extra Credit!

- Add in one additional function as described in Programming Checkpoint 2
- A scanner which asks the user to enter the inventory filename (allows you to progress day to day)
- Institute a daily sale! One way to do this would be to randomly select one item to receive a discount. How this affects sales and revenues should be noted in the end of day report.
- Have something in mind? suggest it for possible consideration.

Coding Style – (3 points)

This grade is awarded for proper coding styles. This includes:

- Appropriate prompts and informative output
- Appropriate method, variable and object names
- Proper indentation
- Good commenting (explains what code is doing)
- Well-organized, elegant solution

Submission Requirements:

1. Submit your NetBeans project export that includes your program to fulfill the requirements. (**lastNameFirstNameProject3.zip** or **lastNameFirstNameProject3.zip**)
2. Your initial inventory file if different than what was provided.
3. Submit a PDF document with the name **project3Info.pdf** that includes the following assignment information:
4. Explanation of any special or additional features you added, if any.
 - If your solution to this problem is incomplete, you may explain the status as of your submission and describe any issues that kept you from completing the assignment.
 - Discuss the easy and challenging parts of the assignment. How did you overcome all or some of the challenges?
 - Did you learn anything that will help you perform better on future assignments?