Concordia University
Computer Science & Software Engineering
Faculty of Engineering and Computer Science

Concordia University
Comp 249 - Winter 2012
Object-Oriented Programming II
Assignment # 1 – Due 11:59 PM – January 22, 2012

Purpose: Practically, this assignment should be considered as *Assignment # **0***! The purpose of this assignment is to help you review some of the main topics covered in previous courses, including classes, loops, arrays, arrays of objects, static attributes and static methods.

General Guidelines When Writing Programs:

- Include the following comments at the top of your source codes
```
// -------------------------------------------------------
// Assignment (include number)
// Question: (include question/part number, if applicable)
// Written by: (include your name and student id)
// -------------------------------------------------------
```
- In a comment, give a general explanation of what your program does. As the programming questions get more complex, the explanations will get lengthier.
- Include comments in your program describing the main steps in your program.
- Display a welcome message which includes your name(s).
- Display clear prompts for users when you are expecting the user to enter data from the keyboard.
- All output should be displayed with clear messages and in an easy to read format.
- End your program with a closing message so that the user knows that the program has terminated.

Part I)

A **book** object has four attributes, a title (String), an author (String), an ISBN (long), and a price (double).

For this part, you are required to design and implement the **Book** class according to the following specifications:

- Upon the creation of a book object, the object must immediately be initialized with valid values; that is title, author, ISBN and price. (Hint: Constructors.)

- The design should allow enough flexibility so that the value of any of these attributes can be modified later on. For example, it should be possible to create a book object with a given price then change its price later on. The design should also allow the user to obtain the value of any of the attributes.
 (Hint: Accessors & Mutators.)

- The design should allow all information of an object to be displayed at once through the utilization of **System.out.print()** method. (Hint: **toString()** method)

- It is required to know how many Book objects have been created. For that, you need to add a method, called **findNumberOfCreatedBooks**(), to the class. This method must return the number of created Book objects prior to the time this method is called. The method would simply return 0 if no books has been created by the time the method is called. (Hint: Static – You are allowed to add other attributes to the class.)

- It is required to compare two Book objects for equality. Two Book objects are considered equal if they have the same ISBN and price. (Hint: **equals()** method)

Part II)

You are the owner of a bookstore and need help in keeping track of your books. Write a driver program that will contain the following methods at least. **Note**: You can have the main function in a separate driver file, or in the same file if you prefer.

1. **a main method**, that will:
   a. Display a welcome message
   b. Prompt the bookstore owner for the maximum number of books (maxBooks) his/her bookstore can contain. Create an empty array, called inventory, that will have the potential of keeping track of the created Book objects.
   c. Display a main menu (fig 1) with the following choices and keep prompting the user until they enter a number between 1 and 5 inclusive:

```
What do you want to do?
    1. Enter new books (password required)
    2. Change information of a book (password required)
    3. Display all books by a specific author
    4. Display all books under a certain a price.
    5. Quit
Please enter your choice >
```

Fig 1. Main menu

   d. When option 1 is entered:
      i. Prompt the bookstore owner for his/her password. (Make sure you have a constant variable containing the password "password" – do not use any other password as it will be easier for the marker to check your assignments). The bookstore owner has 3 tries to enter the correct password. After the 3rd illegal entry, the main menu in figure 1 is re-displayed again.

      ii. If the correct password is entered, ask the owner how many books s/he wants to enter. Check to make sure that there is enough space in the bookstore (array of Book) to add that many books. If so, add them; otherwise inform the owner that he/she can only add the number of remaining places in the array. (How the book information will be input/entered by the user, is up to you).

   e. When option 2 is entered :
      i. Prompt the bookstore owner for his/her password. (Make sure you have a constant containing the password "password" as a constant – do not use any other password as it will be easier for the marker to check your assignments). Again the bookstore owner has 3 tries to enter the correct password. After the 3rd illegal entry, the main menu in figure 1 is re-displayed again.

      ii. Ask the user which book number s/he wishes to update. The book number is the index in the array inventory. If there is no Book object at the specified index location, display a message asking the user if he/she wishes to enter another book, or quit this operation and go back to the main menu. If the entered index has a valid book, display the current information of that book in the following format:

```
Book # X
Author: name of author
Title: title of book
ISBN: isbn #
Price: $price
```

Then ask the user which attribute they wish to change by displaying the following menu.

```
What information would you like to
change?
     1.  author
     2.  title
     3.  ISBN
     4.  price
     5.  Quit
Enter your choice >
```

Fig 2. Update menu

Once the user has entered a correct choice, make the changes to the attribute then display again all of the attributes on the screen to show that the attribute has been changed. Keep prompting the user for additional changes until the user enters 5. Each time the user is prompted for a choice make sure that a number from 1 to 5 is entered, otherwise keep prompting until a valid number is entered.

f. When **option 3** is entered, prompt the user to enter an author name. You then need to display the information of all books by that requested author. (Hint: You may use a static method, for instance called findBooksBy , which accepts a string for an author name then performs the needed search).

g. When **option 4** is entered, promtp the user to enter a value (representing a price). You them need to display all books that have a vlaue smaller than that entered value. (Hint: You may use a static method, for instance called findCheaperThan, which accepts a double value, for a price, then performs the needed search).

h. When **option 5** is entered, display a closing message and end the driver.

Submitting Assignment 1

- Zip together the source codes. (Please use WINZIP).
- Naming convention for zip file: Create one zip file, containing all source files for your assignment using the following naming convention:
  The zip file should be called *a#_studentID*, where # is the number of the assignment *studentID* is your student ID number. For example, for the first assignment, student 123456 would submit a zip file named *a1_123456.zip*

- Submit your zip file at: https://fis.encs.concordia.ca/eas/ as **Programming Assignment** and submission **#1**. Assignments submitted to the wrong directory would be discarded and no replacement submission will be allowed.
- Submit only ONE version of an assignment. If more than one version is submitted the first one will be graded and all others will be disregarded.

Evaluation Criteria for Assignment 1 (10 points)

Part 1 (Class Book3)	**4 pts**
Default & copy constructors	1 pt
Accessor/mutator method for static attribute	1 pt
equals, toString and static attributes/methods	2 pts
Part 2 (Driver & other static methods)	**6 pts**
Handling of password	1 pt
Handling of option 1	1 pt
Handling of option 2	1 pt
Handling of option 3	1 pt
Handling of option 4	1 pt
Handling of option 5	1 pt