<u>Purpose:</u> 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. Practically, this assignment should be considered, as indicated by its number, as *Assignment # 0*! The assignment is there mainly for self-assessment, which is very important. While you must submit the assignment by its due date, <u>the assignment will not be marked and will not carry any load for the grading.</u> So, you must do your best and judge yourself.
<u>However; in case no submission is made, a 4% penalty will be applied.</u>

<u>General Guidelines When Writing Programs:</u>
- 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.

- 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.

- 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: You are allowed to add other attributes to the class if needed.)

- 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)

- It is required to display any Book object (all info of that object) using System.out.println() method. (Hint: **toSting()** method)

Part II)
You are hired by a bookstore to write a software that helps the store owner in keeping track of the books at the store. Write a driver program that will contain, <u>at least</u>, the following methods. **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 <u>option 1</u> is entered:
      i. Prompt the bookstore owner for his/her password. (Make sure you have a constant variable containing the password "249". For consistency, do not use any other string as the password. The bookstore owner has a maximum of 3 attempts to enter the correct password. After the 3rd illegal entry, the main menu in figure 1 is re-displayed again. Additionally after this process is repeated 4 times (i.e. total failed attemopts is 12 by

now), the program must display the following messages: "Program detected suspicous activities and will terminate immediately!", then the program must exits.

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

e. When <u>option 2</u> is entered :
    i. Prompt the bookstore owner for his/her password. Again the bookstore owner has 3 attempts to enter the correct password. However, after the 3rd illegal entry, the main menu in figure 1 is simply re-displayed again (notice the different behaviour in that case from the previous one above).

    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 re-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** (in the main menu shown in Fig. 1) 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** (in the main menu shown in Fig. 1) 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** (in the main menu shown in Fig. 1) is entered, display a closing message and end the program.

Submitting Assignment 0

- For this assignment, you are required to work individually
- <u>Naming convention for zip file</u>: Create one zip file, containing all source files and produced documentations for your assignment using the following naming convention:
- The zip file should be called *a#_StudentName_StudentID*, where # is the number of the assignment and *StudentName/StudentID* is your name and ID number respectively. Use your "official" name only - no abbreviations or nick names; capitalize the usual "last" name. For example, for the first assignment, student 12345678 would submit a zip file named like: *a1_Mike-Simon_123456.zip.*
- You have to submit your assignment using Moodle under **Assignment 0 Submission** folder. **Assignments uploaded to an incorrect DropBox/folder will not be seen and will result in 4% penalty as if no submission was made.**

⇨ **Once again, please notice that no submission will result in a total course penalty of 4%.**

Now, for self-assessment, this is how the assignment *would have been* evaluated if we would to mark it.

Evaluation Criteria for Assignment 0 (10 points)

Part I (Class Book)	4 pts
Default & copy constructors	1 pt
Accessor/mutator method for static attribute	1 pt
equals, toString and static attributes/methods	2 pts
Part II (Driver & other static methods)	**6 pts**
Handling of password	1 pt
Handling of option 1	1.5 pt
Handling of option 2	1 pt
Handling of option 3	1 pt
Handling of option 4	1 pt
Handling of option 5	0.5 pt