**HW1 - Object Oriented Programing - Objects and Classes - Creating a Book Store Program**

Book class (Total 11 points)

* 2pts Create a Book class with the following fields: pageLength (int), genre (int)
* 2pts Create a Book constructor that takes two ints (one for pageLength and genre) that sets the fields to these values.
* 2pts Practice data encapsulation with all fields (look it up if you don't know what it is) in the Book class. Make all fields readable and writable (use getters and setters).
* 1pt In the Book class, create the following static int constants for pageLength: SHORT, MEDIUM, LONG  (0, 1, 2 respectively) and for genre: NONFICTION, FICTION, FANTASY, and SCIFI  (0, 1, 2, 3 respectively). Do not use getters and setters for these; make them public. Whenever you create a new Book, use these to enter the information --- do not enter integers directly in to the Book constructor!
* 2pts In the Book class, create a calcCost method that returns the cost of the book according to the following table:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | NONFICTION | FICTION | FANTASY | SCIFI |
| Short | 9.90 | 10.25 | 15.25 | 12.50 |
| Medium | 20.10 | 22.00 | 20.00 | 15.00 |
| Long | 32.00 | 30.50 | 25.50 | 22.50 |

* 2pts In the Book class, create a toString method that returns a formatted string representing the book in the following format: "Short Book, Fiction, $10.25." The money and type formats must be exactly as above (e.g., $9.90, not 9.9).

Order Class (Total 6 points)

* 2pts Create an Order class with three fields: id (int), books (Book ArrayList), and time (Date). Initialize time with the current time in the constructor, and enter a unique integer in the id field. Encapsulate the first two fields with both read and write access, and the time with read access only.
* 2pts In the Order class, implement a toString method that: print out all the books with their prices in order (each on their own line), the total cost, and the time the order was created (in human readable format).
* 2pts Implement a calculateOrderTotal method in the Order class that computes the total cost of all the books in its Book ArrayList.

Main class/Bookstore (Total 11 points)

* 1pt Create a Main class with a main method.

2pts Create a command line program in the Main class that presents the following menu to a user. This will be continually presented to the user until the user enters 0 and quits (use a while loop).

Order Book (1)

Exit (0)

* 2pts The first time a user enters 1, create a new Order object. Each time the user selects (1) present a book type menu:

NonFiction (1)

Fiction (2)

Fantasy (3)

SciFi (4)

Previous menu (0)

* 2pts If the user selects 0, return them to the previous menu. If they select anything other than 0, keep track of the choice and present them with this menu:

Short (1)

Medium (2)

Long (3)

Previous menu (0)

* 2pts If the user selects 0, return them to the previous menu. If they select anything other than 0, create a new Book with the entered information,*using the constants*, and store it in the Order's ArrayList of books. Print out the string representation of the book implemented earlier followed by "added to order."
* 2pts When the user presses 0 to exit the program at the top level menu, if books have been ordered, use the toString method in Order to print out all the books in the order (each on their own line), the total cost, and the time the order was created (in human readable format).