Programming Fundamentals II Sec. 601

Lab Assignment #7

Inventory Management

Due date: 10/14/20 at 11:59 pm

Purpose: The lab this week focuses on implementing a custom class with an equals and a toString method as well as using those methods in an application.

Task: Create a project called Inventory_FirstName_LastName or Lab7_FirstName_LastName. This project will consist of two classes: an Item class that you implement as well as the Inventory class provided on Blackboard. Remember to include comments in the Item class as well as the relevant parts of the Inventory class.

1. The Item class includes two fields: a String for the name and a double for the price of an item that will be added to the inventory.
2. The Item class will require one constructor that takes a name and price to initialize the fields. Since the name and price will be provided through TextFields, the parameters can be two Strings (be sure to convert the price to a double before storing this into the field).
3. Write getters and setters for the name and price. Try to be consistent with the behaviors of the setters compared to the constructor.
4. Write an equals method that returns true if two Item objects have the same name.
5. Write a toString method that returns the following String (replacing <name> and <price> with the fields of the object): “The <name> item has a price of $<price>.” Make sure the price is formatted to always round to two decimal places.
6. Study the code provided in the Inventory class. The addButton will add an Item to the ArrayList if that Item is new. This means the name of the Item must be different than any name already in the ArrayList.
7. In the Inventory class, you must implement the lambda expression for the addButton. Start by declaring a boolean to keep track of whether or not an item being added to the inventory (the ArrayList of Items) is actually new. This boolean is initialized to true.
8. Construct a new Item using the text from nameTextField for the name and the text from priceTextField for the price. If the constructor does not already handle the conversion, make sure the text for the price is converted to a double.
9. Write a for-each loop to iterate over the items in the ArrayList. Compare each item already in the ArrayList to the new item. If the name of any item in the ArrayList is the same as the name of the new item, set the boolean to false.

10. After the for-each loop completes, if the item is new, add the item to the ArrayList and display the following in the outputLabel: “Successfully added new item. The <name> item has a price of $<price>.” Keep in mind that the second sentence is the result of calling the object’s toString method. If the item is not new, display the following in the outputLabel: “Failed to add existing item.”

Criteria: Comments summarizing the program are worth 5 points. The fields of the Item class are worth 3 points each (6 points total). The Item constructor is worth 6 points. The getters and setters are worth 3 points each (12 points total). The equals method and toString method are worth 6 points each (12 points total). Properly declaring and initializing the boolean is worth 2 points. Constructing the Item object is worth 12 points. The for-each loop to determine if the item is new is worth 25 points. Adding the item to the ArrayList and displaying the correct text in the output is worth 20 points.