AP Computer Science - Class Construction

**Part I. *LaundryOrder***

Create a class called LaundryOrder. Each LaundryOrder will need to be able to store the following information:

* The name of the customer
* The number of items to be cleaned
* Whether the order is ready
* Amount due by the customer

The following operations should be handled by the LaundryOrder object:

* Create a laundryOrder object given a customer name, number of items and amount. Assume it is not ready.
* Access the customer name, number of items, order status and amount due
* Change the value of the name, items, status and amount.

**Part II. AlterationOrder**

The laundry also does alterations on items brought in for cleaning and an AlterationOrder class will be used to keep track of alteration orders. An AlterationOrder is a LaundryOrder with the following additional properties:

* An AlterationOrder includes a String that specifies what is to be done (alteration instructions)
* Alterations are performed on only one item, therefore the number of items is always one.
* The alteration instructions can be accessed but not modified.

**Part III. Laundry**

To keep track of all current orders, the laundry uses the Laundry class.

* The Laundry class should contain all LaundryOrders in an ArrayList called allOrders.
* The Laundry should also implement the following methods:

a. public ArrayList<LaundryOrder> unfinishedOrders()

This method returns an ArrayList of all LaundryOrders that are not ready.

b. public double averageBill()

This method will calculate the average amount due by the customers for all LaundryOrders in the ArrayList.

c. public ArrayList<LaundryOrder> combineOrders()

This method combine any LaundryOrders for the same customer name and rebuild the *allOrders* ArrayList appropriately. In other words, if Bill Dixon had 2 separate orders, dropped off at different times, he would have 2 elements in the allOrders ArrayList. The combineOrders method would add up the items and amount due of both of Bill's orders, as well as changing the order ready status to true only if both of the orders are ready. This will affect the size of the allOrders ArrayList, since two orders are combined into one.

**Part IV. LaundryRunner**

Build a runner to test all methods. Make sure you have at least 2 orders by people with the same name to make sure the combineOrders method works properly. Have your runner show the size of the allOrders ArrayList before and after combine methods.

When you've completed, print out your 4 class files and include a copy of the runner output.