CMPT220 - Program 8
Program Due: Thursday, April 9th, before noon. (saved, submitted and printed to PDF)
Name the project **Prog8YourLastName**
Name the main class **ShoppingDemoYourLastName.java**
Name the Item class **ItemYourLastName.java**
Name the KeyedList class **KeyedListYourLastName.java**

It's time to go shopping! This program lets us exercise the Keyed List Class. You are to complete the implementation of the **Ordered Array implementation** in which the **`add`** method inserts an item into its proper sorted slot in the array.

Items on our list will consist of the name of the item (as the key), the quantity of that item and the item's unit price. Call the item class **ItemYourLastName**. The key field is implemented using the Java String class. All key comparisons should be done in a case insensitive manner using the **`String`** class comparison methods **`equalsIgnoreCase`** and **`compareToIgnoreCase`**.

This example shows how to compare keys. You won't use this exact code – this is just an example.

```
String key1 = list[j].getName();
String key2 = list[j-1].getName();
if(key1.compareToIgnoreCase(key2) < 0)
   key1 precedes key2 in dictionary order
if(key1.compareToIgnoreCase(key2) > 0)
   key1 follows key2 in dictionary order
if(key1.compareToIgnoreCase(key2) == 0)
   key1 and key2 are identical
```

Your program will be menu-driven and will allow the shopper to perform the following functions, each corresponding to a method of the Keyed List Class.

The **`KeyedListYourLastName`** class will implement these methods:

`public KeyedListYourLastName()`: this default constructor creates an empty list. Initialize the capacity of your internal array implementation to hold up to 10 items. You will need an instance variable to keep track of how many items are stored in the array.

`public void clear()`: Resets the list to the empty state. This method should not need to replace any entries with the null value.

You should implement a **private** "helper" method **`findIndex(String keyValue)`** that returns the array index of an item if it is found; else it returns -1. This functionality will be re-used by `add` and `remove` methods.

`public boolean add(ItemYourLastName product)`: Inserts the product into the list **if its key is unique** (case insensitive) and the list is not full. Returns true if successful; otherwise, false. The add method should first use `findIndex` to check for a duplicate. Your `add` method must scan the array to find the sorted position at which to insert the new unique item.

`public boolean remove(String keyValue)`: Deletes the ItemYourLastName with the given key value (the name attribute). Returns true if an ItemYourLastName was found and deleted; else, false. Use helper method `findIndex` to check if the item is present and where it is located.

`public ItemYourLastName retrieve(String keyValue)`: Returns the **ItemYourLastName** having the specified key value; otherwise, return null if not found.

`public boolean isEmpty()`: Returns true if and only if the list is empty.

`public boolean isFull()`: Returns true if and only if the list is full.

`public void print()`: Prints all of the ItemYourLastNames in the list in ascending order by the key field.

Add the following methods to your **`KeyedListYourLastName`** class:

`public int getCount()`: Returns the total number of items to be purchased **by accumulating the sum of the quantity fields** of all ItemYourLastNames in the list.

`public double calcTotal()`: Returns the total cost of all items in the list. For each Item multiply the unit price by the quantity.

The program's menu should appear as shown below. Please **order and number** the menu choices as shown below.

```
1. Add an item to the list
2. Delete an item from the list
3. Print each item in the list
4. Search for a user-specified item in the list
5. Count the total number of items in the list
6. Total the cost of the items in the list
7. Determine whether the list is empty
8. Determine whether the list is full
9. Clear the list
0. Quit
```

Each menu command should display informative prompt messages as well as print a message indicating the result of processing that command. See the sample program run for example prompt and output messages.

We will use an input file to get the initial data for this program. The input file will contain a line that specifies the number of ItemYourLastNames in the file, followed by the Name, Quantity and Price of each Item, **in that order**, each input value on a line by itself. Here is an example of what the input file will look like:

```
2
book
5
15.00
dvd
8
20.00
```

Your program should start by asking for the name of the initial input file, then reading in and adding those ItemYourLastNames to the KeyedListYourLastName. After that, present the menu -- all of the rest of the data will be entered via the keyboard.

Also, be sure to submit your test input file(s) – the easiest way to do this is to copy your input file(s) into the project directory before you .zip it.

Sample Program Run (assume the input file from above)

Please enter the path and name for the data file: **C:\\Users\Public\prog8input.txt**
Thank you.

<prints menu>
Please enter your choice: **1**

Please provide information about the item to add.
Please enter the name: **Cereal**
Please enter the quantity: **4**
Please enter the unit price: **4.00**

Cereal has been added to the cart.

<prints menu>
Please enter your choice: **1**

Please provide information about the item to add.
Please enter the name: **Bananas**
Please enter the quantity: **3**
Please enter the unit price: **0.80**

Bananas has been added to the cart.

<prints menu>
Please enter your choice: **5**

Your cart contains a total of 20 items.

<prints menu>
Please enter your choice: **6**

The total price of all of your items is $253.40.

<prints menu>
Please enter your choice: **4**

Please enter the name of the item to find in your cart: **cereal**

Yes, you have 4 Cereal at $4.00 each.

<prints menu>
Please enter your choice: **2**

Please enter the name of the item to delete from your cart: **pizza**

Sorry, but your cart does not contain the item pizza.