CS 1302 – HW 4

*1-Many, ArrayList*

Contents

[1 Overview 1](#_Toc167792962)

[2 Steps to Complete 1](#_Toc167792963)

[2.1 Modify Item Related Classes & Test Classes 1](#_Toc167792964)

[2.2 Modify Warehouse Class 2](#_Toc167792965)

[2.3 Modify *WarehouseTest* Class 2](#_Toc167792966)

[3 Grading Criteria 3](#_Toc167792967)

[4 Submission Requirements 3](#_Toc167792968)

# Overview

1. You will modify the *Item* class from HW 3 to implement *equals.*
2. You will modify the *Warehouse* class from HW 3 so that it uses an *ArrayList* instead of an array. In addition, you will change the implementation of several methods, and add 2 more.
3. Ensure that the provided *HW04CompileTest.java* compiles. This ensures that you have properly implemented the signature of the methods which allows my grading program to operate on your code. This is found in hw04\_code.zip on the HW Page.
4. Type this Academic Honesty statement followed by your full name, as a comment at the top of the *Warehouse* class:

“This homework represents my own work. I understand that I may receive help, but I did not copy any portion of this assignment from anywhere. I understand that a violation of this will result in a Report of Academic Dishonesty.—YOUR FULL NAME HERE”

# Steps to Complete

## Modify Item Related Classes & Test Classes

1. Setup – do the following:
2. Create a Java Project in Eclipse with the name: *hw4\_lastName*.
3. Using File Explorer, select the *prob1* folder **from HW 3** and drag into the *src* node in the Package Explorer in Eclipse.
4. Modify the *Item* class so that two items are considered equal if they have the same *name*. Hint: override the *equals* method.
5. Add 2 test methods to *ItemTest* to test *equals*: once when the method returns *true,* and one when it returns *false.*
6. Add 2 test methods to *RefrigeratedItemTest* to test *equals*: once when the method returns *true,* and one when it returns *false.* Remember, *equals* will be inherited in the *RefrigeratedItem* class.

## Modify Warehouse Class

You need to modify the *Warehouse* class as described below. You should read through this entire section before coding.

1. Change the array implementation of *items* to an *ArrayList*. Hint: look at every method, most/all methods will have to be changed to support this.
2. Change the implementation of these methods:
3. Change the *addItem(item:Item)* method so that it only adds an item if it doesn’t already exist (*i.e.* if there is not already an item that is equal to it). It should also return *true* if the add is successful and *false* otherwise.
4. Change the *getItem(name:String):Item* method to use an approach that relies on *equals*. In other words, eliminate the loop approach and instead use the approach from the notes where we used a dummy *Item,* and the *contains* and *indexOf* methods.
5. Change the *getRefrigeratedItems()* so that it returns: *ArrayList<RefrigeratedItem>*. You will no longer need the helper method, so you can delete it.
6. Change the *removeItem(name:String):Item* methodso that it uses an approach that relies on *equals,* similar to the *getItem(name:String)* method. In other words, eliminate the loop approach and instead use the approach from the notes where we used a dummy *Item*.
7. Add these new methods:
8. *getItemsWithName(partialName:String):ArrayList<Item>*. This method returns an *ArrayList* of items whose *name* contains *partialName* at the beginning. For example, if these items have names: *Dog Food, Pretzels, Dog Collar, Basketball, Doghouse* and *partialName = “Dog”,* then the items with these names will be returned: *Dog Food, Dog Collar, Doghouse.*
9. *hasItem(name:String):boolean*. This method returns *true* if there is an item whose *name* is the same as the input name,and *false* otherwise.

## Modify *WarehouseTest* Class

You need to modify the *WarehouseTest* class as described below. You should read through this entire section before coding.

1. Modify the *WarehouseTest* class so that the existing test methods work with the changes in *Warehouse* to use an *ArrayList*. Hint: whatever doesn’t compile needs to change. I believe there is only one method that needs to change: *getRefrigeratedItems*.
2. Add a test method, *testAddItem\_Duplicate* for *addItem* to ensure that it will not add a duplicate *Item*.
3. Add a test method, *testGetItemsWithName* for *getItemsWithName* where the method returns 2 or more items that partially match the argument.
4. Add two test methods for *hasItem,*
5. *testHasItem\_Find* that finds the *Item* requested.
6. *testHasItem\_NoFind* that does not find the *Item* requested.

# Grading Criteria

|  |  |
| --- | --- |
| **Grading Criteria** | **Points** |
| Automated Tests | 76 |
| ItemTest (3+2) | 6 |
| RerigeragedItemTest (4+2) | 6 |
| WarehouseTest (16+4) | 12 |
| **Total** | **100** |

# Submission Requirements

Checklist:

|  |  |  |
| --- | --- | --- |
|  | **Complete?** | **Requirement** |
| 1. |  | *HW04CompileTest* compiles. |
| 2. |  | The Academic Honesty statement followed by your full name, appears as a comment at the top of the *Warehouse* class. |
| 3. |  | All Java files (including test classes) are in the *prob1* package. |
| 4. |  | Your *prob1* folder is zipped into a file name: *hw4\_yourLastName.zip.** See Lab 2, Stage 9 for exact instructions.
* Do not zip your workspace folder
* Do not zip your *src* folder.
* Do not zip just the java files
* Do zip just your *prob1* folder
 |
| 5. |  | Submit in the *hw4* dropbox on Blazeview by the deadline. |