Sumedh Shah
Qing Wang
EE 422C - Program #3

Analysis

The input to our program will be a text file with a series of transactions that will be given to us line-by-line. The text file is like its own shopping cart and its per customer. There are 5 transactions that a shopper would want to make (insert, delete, update, search, or print). They will work with 3 different items (groceries, electronics, and clothing). Each of these items has its own classes and depending on the item, has different functions from their superclass, Item. Groceries must account perishability and thus, premium shipping. Electronics must account fragility and thus, premium shipping, and they also must account the shipping state, and thus, sales tax depending on the state. Groceries have no sales tax. Shoppers can only make transactions on only 1 shopping cart. On top of these simple transactions, you must determine for the delete and update functions whether those items exist. The search function must not search through an empty array. Each of the transactions has a different number of words/numbers that you must input, and for insert, there are 3 variations. Assuming each of the transactions is formatted correctly, you must read an array of commands different for each of the 5 transactions differently. Before all the transactions, you must check for the length and type of each command in each transaction.

ADT Level Description of Classes (Assignment 3 Shopping Cart)

- ShoppingCartDriver.java
  - Input processed as one String per line, and it is checked by our program and spit out as a String[] (String array), between 1 and 8 Strings long containing each field in the input command (operation, category, name, quantity, price, weight, fragility, state). This **String[]** is named “**command**.”
  - In the main method, there is an **ArrayList<Item> shoppingCart** that stores every item as we insert them in one by one
- Item.java
  - An **Item** object has a String name, float price, String premium (“F”, “NF”, etc.), int weight, and int quantity as instance variables
  - Has methods printItemAttributes that prints using the instance variables, a calculatePrice that calculates a final price based on tax and shipping (returned as a float)
  - Includes a constructor that places parameters into instance variables (initializes them)
- Electronics.java/Clothing.java/Grocery.java
  - Updated different types of the constructor (with 1-2 more fields), printAllAttributes (with 1-2 more fields) and calculatePrice (calculated with special cases for each)
- Functions.java
  - insertItem returns an **ArrayList<Item>** that is the updated shoppingCart after putting in a new Item (and initializing it as one of Grocery/Electronics/Clothing) and putting it in the ArrayList
  - deleteItem returns an **ArrayList<Item>** that removes any items with a certain name
  - searchItem outputs how many different items have a certain name in the **ArrayList<Item>**
  - updateItem returns the **ArrayList<Item>** with up to one quantity of one item changed
  - printAll uses the **ArrayList<Item>** to go through and invoke printItemAttributes in all of them

High-Level Algorithm - ShoppingCartDriver

```
Create array list of Item objects
Check input file to make sure it exists and then proceed
Read in file
Create Functions object to run insert, delete, search, update, and print
Loop through each line in file
    Split commands in line by space and store in array
    Check errors in the command
        Check length of command and ensure that each element is correct type
    If no errors in the command
        Read in first element of command
        If first element = insert
            function object takes in arraylist and command and performs insert
        Else If first element =delete
            function object takes in arraylist and command[1] and performs delete
        Else If first element = search, function object performs search
            function object takes in arraylist and command[1] and performs search
        Else If first element = update, function object performs update
            function object takes in arraylist and command[1], [2] and performs update
        Else If first element = print, function object performs print
            function object takes in arraylist and prints all contents
    Else if there are errors
        go to next line in file and repeat from “Loop….”
```