Item

+ Name : String
+ Quantity: Integer
+ Price: Double
+ Weight : Integer

+ calculatePrice(string) : double
  Uses temporary double final_price for calculation
+ toString(void) : String
+printItemArributes(void)

Operations:
-calculatePrice(): returns a double containing the price of the item, including shipping and taxes. This method is called by the driver, where a running tally of all items is held in a double called runningTotal.

-toString(): This overriding is used to create a string that is outputted by printItemArributes().

-printItemAttributes(): Uses system.out.println() to output the toString() method for the Item (parent) class and other classes that inherent from Item.

Pre/Post Conditions
-Variables Name, Quantity, Price, Double are set at construction.
-Instances of this class are held in an ArrayList called shoppingCart in A3Driver.java

Clothing (extends Item)

TAX_RATE: final double

toString(): String
calculatePrice(): double

Operations:
calculatePrice(): Override Item because Clothing is taxed. Returns a double containing the price of the item, including shipping and taxes. This method is called by the driver, where a running tally of all items is held in a double called runningTotal.

-toString(): This overriding is used to create a string that is outputted by printItemArributes(). Adding "Category: Clothing" to output.

Pre/Post Conditions
-Variables Name, Quantity, Price, Double are inherited from Item and set at construction.
-Instances of this class are held in an ArrayList called shoppingCart in A3Driver.java

Electronics (extends Item)

TAX_RATE: final double
state: String
fragility: boolean

toString(): String
calculatePrice(): double

Operations:
calculatePrice(): Override Item because Electronics is taxed depending on the state and some require premium shipping. Returns a double containing the price of the item, including shipping and taxes. This method is called by the driver, where a running tally of all items is held in a double called runningTotal.

-toString(): This overriding is used to create a string that is outputted by printItemArributes(). Adding "Category: Electronics" as well as what state and fragility are set at to output.

Pre/Post Conditions
-Variables Name, Quantity, Price, Double are inherited from Item and set at construction. TAX_RATE is set to 0.1 at initialization. state and fragility are used in calculatePrice() and are set by the constructor.
-Instances of this class are held in an ArrayList called shoppingCart in A3Driver.java

Grocery (extends Item)

perishable: boolean

toString(): String
calculatePrice(): double

Operations:
calculatePrice(): Override Item because Grocery items require premium shipping if they are perishable. Returns a double containing the price of the item, including shipping and taxes. This method is called by the driver, where a running tally of all items is held in a double called runningTotal.

-toString(): This overriding is used to create a string that is outputted by printItemArributes(). Adding "Category: Grocery" as well as what perishable is set at to output.

Pre/Post Conditions
-Variables Name, Quantity, Price, Double are inherited from Item and set at construction. perishable is used in calculatePrice() and is set by the constructor.
-Instances of this class are held in an ArrayList called shoppingCart in A3Driver.java

A3Driver

shoppingCart: ArrayList<Item>
file: File

main(String{}):void
getInput(Scanner, ArrayList<Item>):void
processInsert(String[], ArrayList<Item>):void
processDelete(String[], ArrayList<Item>):void
processUpdate(String[], ArrayList<Item>):void
processSearch(String[], ArrayList<Item>):void
processPrint(String[], ArrayList<Item>):void
insertAlphabetically(String[], ArrayList<Item>):void
isState(string):boolean

Operations:
-main(): Creates an ArrayList<Item> called shopping cart for storing Items (and it's children) in the shopping cart. Scans input from text file specified in args and calls getInput() to parse transactions.

-getInput(): Processes input from file one line (transaction) at a time.

-processInsert(): If an insert command is necessary, this method reads all commands from an insert transaction line and makes sure every field is filled correctly, and then creates an Item of the appropriate type with information provided from the file and adds it to the shopping cart ArrayList

-processDelete(): If delete command is called for, will delete all occurrances of an item from the shopping cart

-processUpdate(): If update command is called for, allows for the first item in the shopping cart with the matching name to have its quantity field updated

-processSearch(): If a search command is called for, searches the shopping cart for all instances of an item with the same name and prints the number of the item to the screen (number of instances, not total quantity)

processPrint(): If a print command is called, then prints statistics on every item in lexicographical order (case insensitive) in the shopping cart (all instance fields for every item), and then prints the total price of everything in the shopping cart

insertAlphabetically(): Inserts an item into the shopping cart arraylist alphabetically by name

isState(): Given a state input string, checks to see if it is one of 50 state abbreviations in the United States

Pre/Post Conditions:
Pre: An empty ArrayList of Items.
Post: An ArrayList of items purchased with associated costs and a final price of items in the cart, including tax and shipping.