Nam Hoang

Software Development I

Prof. Arias

May 4 2018

Final Project: Inventory Program

Abstract:

This is a program that allows an user to create and manipulate an inventory of clothing items. The program fetches data from a text document to create an array list of items. From here, the user can shape the inventory to their liking. This includes adding items, deleting items, and modifying items. after all modifcations to the inventory is done and the user chooses to exit the program, updates to the inventory is saved back into the text document. The program allows the user to keep a structured inventory for better management and demonstrates the implementation of core Java programming concepts learned in Software Development I.

Introduction:

Regardless of their interest and participation in fashion, everybody has a collection of clothing. Many like to keep it simple with their wardrobes contaitning just a few essential clothing items. But for many others, clothing and fahsion is a great way to express themselves and satisfy their aesthetic eye. With the interest in fashion comes the understanding that

diversity is a great way to elevate one's style. So they buy and owns multiple pieces. But having many differences pieces can lead to management problems and that is where this program can be of use. With it, one can create an inventory from scratch and manage it will relative ease.

User manual:

The user interacts with the program through entering inputs based on a series of prompts. The best analogy to describe this interaction would be how one interacts with the automatic answering system when they call the customer service line for their telephone service provider. The user selects the numbers corresponding to the action that they want to execute which invokes the corresponding method. The user then follows the prompts to comeplete the action.

Detailed System Description:

The user interacts with the system through the main method of the Inventory class. The user invoke the methods of the inventory class for different functions of the program. First, when initiated, the program calls the programStart method which locates the .txt data file. Using a scanner, the content of the

file is read and the information is used to create an ArrayList that represents the inventory.

From there, the user can choose to add new items to the inventory (or Item objects to the ArrayList) and delete or modify an existing item in the inventory. When invoked, the addItem method will display a series of prompts through which the user can input the information (item type, brand, name, size, buy price,...) of the item being added to the inventory. The constructor method in the class of the item will create the item object and it will be added to the ArrayList.

Once added, an item can be deleted or modified through the deleteItem and modItem method, respectively. The delete Item method, when invoked, asks the user to input the item ID and iterate the ArrayLsit to find and delete the item object that matches that Id number. The modItem method asks the user specify the item and the aspect of the item that they want to modify, enter the new value, and proceeds to modify the item.

Lastly, the user can exit the program after they have finished shaping their inventory through the programExit method. The method takes the updated ArrayList, and using an insertion sort algorithm, sort the items in the ArrayList by ID number. It

then takes print the updated and sorted Array to the inventory.txt file.

Requirements:

Usage-wise, the program performs only a few simple actions. Takes data from a .txt File and create an ArrayList, let the user add, delete, and modify the objects in that ArratList, and print the ArrayList back to the .txt file to save the changes. But code-wise is where the complexity of the program lies. The program implements the core Java programming concepts of Software Development I such as loops and if statements, creating and iterating array and ArrayList, exception handling, creating, reading, and printing files, and sorting algorithms.

UML Diagrams

Inventory Class
- inventory: ArrayList<Item> + running: boolean
+ addItem(): void + deleteItem(): void + modItem(): void + programExit(ArrayList<Item> inventory): void + programStart(ArrayList<Item> inventory): void + main(): void + printInvent(): void + isNumeric(String str): boolean + insertionSort(ArrayList<Item> inventory): void

Item Abstract Class
- brand: String - itemName: String - itemId: int
+ getBrand(): String + setBrand(String brand): void + setItemtype(String): + getItemName(): string + setItemName(String itemName): void + getItemId(): int + setItemId(int itemId): void + isId(int i): boolean + toString(): String + toInvent(): String

Top Class extends Item Class
- size: String
+ Top() + Top(String brand, String itemName, String size) + getSize(): String + setSize(String size): void @Overriding Item class + toString(): String + toInvent(): String

Bottom Class extends Item Class
- waist: int - inseam: int
+ Bottom() + Bottom(String brand, String itemName, int waist, int inseam) + getWaist(): int + setWaist(double waist): void + getInseam(): int + setInseam(double inseam): void + getSize(): String @Overriding Item class + toString(): String + toInvent(): String

Shoes Class extends Item Class
- size: double
+ Shoes() + Shoes(String brand, String itemName, double size) + getSize(): double + setSize(double size): void @Overriding Item class + toString(): String + toInvent(): String