GIT Department of Computer Engineering
CSE 222/505 - Spring 2021
Homework-6 # Report

Can Duyar
171044075

PROBLEM SOLUTIONS APPROACH

The first process was reading the .csv file and then creating the product file from it. I made sure that the path of the "Sample" file to be processed was passed from the outside as a command. RuntimeException is thrown if the .csv file entered does not exist in the system. I used 'Buffer' for read/write operations because the .csv file is quite large. Therefore, my read/write operations become faster. I avoided reading the whole file and keeping it in the memory in general. I also added Queue to make it more efficient. It reads the lines and adds them to the queue. When there are 1,000 lines, it writes from Queue to the file. The purpose of using this is to make it write in bulk, rather than constantly going to the file. Also, since the entire .csv file was read during the creation of the product file, I kept the trader names in a TreeSet at that time, so I avoided spending extra effort. I created user file from this TreeSet. To distinguish between the added users being Traders or Customers, I kept the information at the end of each line.

To facilitate the testing, I created 1 trader and 1 customer. I performed my test operations on these 2 users.

Trader -> id: 50000000 password: 123456 name: testTraderName

Customer -> id: 50000001 password: 123456 name: testCustomerName

I created product and user nets in the FileUtils class.

I divided the read/write operations into UserFile, OrderFile, ProductFile files. The related processes are located in their classes. (For example, User operations occur in the UserFile class). These files are called only from service classes.

I had no problems with add operations. Data can be added to the file later, since the Append feature is turned on. However, the situation was slightly different in update and remove operations.

In remove operations, I wrote all data except the line to be deleted into a temp file. Then I deleted our old file and changed the name of the temp file to the old file name.

In update operations, I wrote all the data except the line to be updated in the temp file, as in the remove operations, I added the current version of the sonar line. So, I did not write the old line in the file, I wrote the new version in the file. Likewise, I deleted the old file and changed the name of the temp file to the old file name.

In Get operations, the object of the appropriate class is created from the incoming line information. These generated objects are added to the ArrayList and returned. I used HashTable to see the orders of each customer. I kept hashTable's key as customer id and values as ArrayList<Order>. In this way, we can see which customer has which orders.

I wrote the Session class to control session information. This class has a login method. This method brings the appropriate User from the user file according to id. If the user is null, it throws an error. If the incoming user's password is not the same as our method parameter password, it throws an error. Otherwise, it returns the user object. I created LoginFailedException as a checked exception. I used Linkedlist to see the trader's products in order. I created their own service classes from the methods they can call for Customer and Trader. (TraderService, CustomerService). I used interface in these classes. Interface can be examined to see which methods are contained. It provides convenience. I kept state information in Order (enum OrderState). This changes if the trader wants to delete or accept the Order.

When the Order was first added, it is in ADDED state. It becomes ACCEPTED when accepted by the OrderTrader. It becomes CANCELED when cancelled by order trader.

Because all areas of Customer and Trader are the same, they extend the User class. The User class is an abstract class. It is asked to show a different menu when users login. The abstract showMenu() method was written for this. Both classes overwrite the showMenu() method.

I have two methods called displayProductsOfTrader and displayProductsByNameOrDescription. The product list given to these methods goes through filter and sort operations on demand. These methods have two additional parameters, SortCriteria and Filter.

The SortCriteria parameter is an enum. It keeps the information which field to sort (NAME, PRICE, DISCOUNT_AMOUNT). I used different sort algorithms for each NAME, PRICE, DISCOUNT_AMOUNT sort. All these sort algorithm classes implement the 'Sort' interface. Polymorphism is applied to call the appropriate algorithm.

*I used the bubble sort algorithm to sort product by price in descending order.

*I used the selection sort algorithm to sort product by name in descending order.

*I used the insertion sort algorithm to sort product by discount amount in descending order.

Filter is a final class and has a private constructor. It creates objects with Factory methods. It calls the method which is wanted to filter according to the category (ofCategory()). It determines which filter to use by switch case, as in sort method.

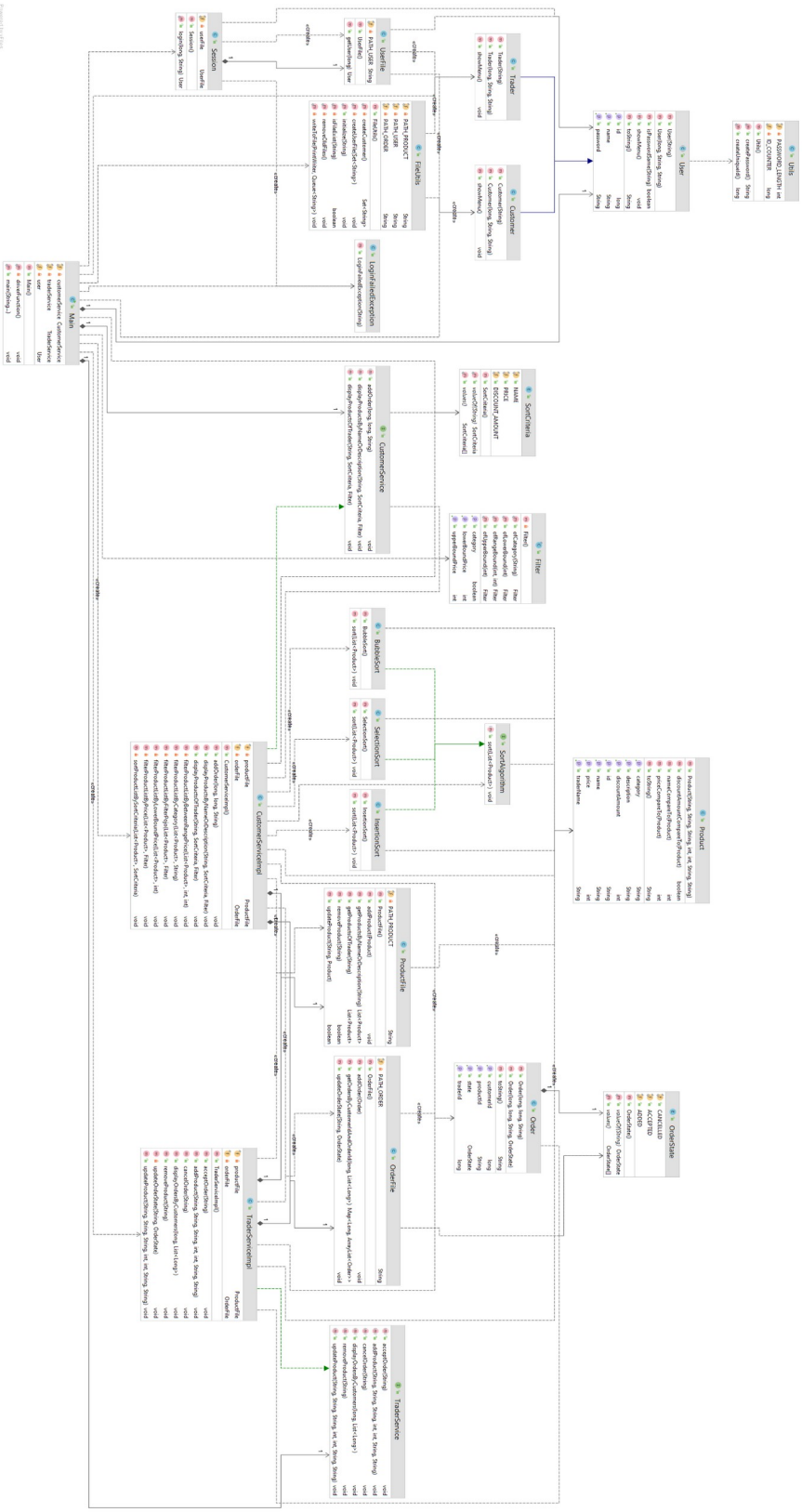
ofUpperBound -> filters by upper limit

ofLowerBound -> filters by lower limit

ofRangeBound -> filters by upper and lower bound

ofCategory ->filters by category

CLASS DIAGRAM



TEST CASES / RUNNING COMMAND AND RESULTS:

Test Case 1: If filename which is given from terminal is invalid then program throws an exception as “File not found!”

```
(base) can@can-ThinkPad-L13:~/Desktop/ds_hw6/Code$ java Main gtu.csv
Exception in thread "main" java.lang.RuntimeException: File not found!
        at FileUtils.isFileExist(FileUtils.java:77)
        at FileUtils.initialize(FileUtils.java:36)
        at Main.main(Main.java:29)
(base) can@can-ThinkPad-L13:~/Desktop/ds_hw6/Code$
```

Test Case 2: If filename which is given from terminal is valid then program works!

```
(base) can@can-ThinkPad-L13:~/Desktop/ds_hw6/Code$ java Main e-commerce-samples.csv
User not found!!
Wrong password!!


----Signed in---- User id: 50000000

Trader : testTraderName
1: Add/Remove/Edit Products
2: See list of order
3: Accept/Cancel Orders

New product added. ProductId: 'EXPEH2FF9KEPROD1'

New product added. ProductId: 'EXPEH2FF9KEPROD2'

New product added. ProductId: 'EXPEH2FF9KEPROD3'

New product added. ProductId: 'EXPEH2FF9KEPROD4'

New product added. ProductId: 'EXPEH2FF9KEPROD5'

Product updated. ProductId: 'EXPEH2FF9KEPROD4'

Product not found not updated. ProductId: 'INVALIDVALUE'

Product removed. ProductId: 'EXPEH2FF9KEPROD5'

Product not found not removed. ProductId: 'INVALIDVALUE'
```

.

.

.

Test Case 3: If the username cannot be found while trying to login, it shows the message ‘User not found!!’ .

Test Case 4: If the password given while trying to login is incorrect, it shows the message ’Wrong password!!’.

```
User not found!!
Wrong password!!
```

Test Case 5: The message shown after the user logs in with his username and password is as below. The user ID, the role they are logged in and their privileges are shown. In the picture below, the user has successfully logged in as a trader.

```
----Signed in---- User id: 50000000

Trader : testTraderName
1: Add/Remove/Edit Products
2: See list of order
3: Accept/Cancel Orders
```

Test Case 6: 5 products have been added. The message ‘New product added. ProductId: ' is displayed after each product that has been successfully added.

```
New product added. ProductId: 'EXPEH2FF9KEPROD1'

New product added. ProductId: 'EXPEH2FF9KEPROD2'

New product added. ProductId: 'EXPEH2FF9KEPROD3'

New product added. ProductId: 'EXPEH2FF9KEPROD4'

New product added. ProductId: 'EXPEH2FF9KEPROD5'
```

Test Case 7: The products can be updated. After updating the product, "Product updated" message is displayed. The product whose product id is "EXPEH2FF9KEPROD4" has been updated.

```
Product updated. ProductId: 'EXPEH2FF9KEPROD4'
```

Test Case 8: It does not allow updating a product that is not present. In such a case, it shows 'Product not found not updated' message.

```
Product not found not updated. ProductId: 'INVALIDVALUE'
```

Test Case 9: The products can be removed. After removed the product, "Product updated" message is displayed. The product whose product id is "EXPEH2FF9KEPROD5" has been removed.

```
Product removed. ProductId: 'EXPEH2FF9KEPROD5'
```

Test Case 10: It does not allow removing a product that is not present. In such a case, it shows 'Product not found not removed' message.

```
Product not found not removed. ProductId: 'INVALIDVALUE'
```

Test Case 11: In order to perform customer operations, a customer role has been entered. After successfully logging in with the username and password, the user's id, role and authorities are shown.

```
----Signed in---- User id: 50000001

Customer: testCustomerName
1: Search Products/Remove/Edit Products
2: Filter by Category/Prices
3: Display all of the products of a trader
```

Test Case 12: The customer can view all of the trader's products. The results for all products of the trader named "TestTraderName" are shown in the picture below. Information of the products, the name of the trader and how many products the trader has in total are shown.

The list of products is listed in descending order of name.

```
sorted by name
---Display Products of trader--
EXPEH2FF9KEPROD4        updated_Headphones_Name Electronic>>Headphones  600    400      Headphones_Description  testTraderName
EXPEH2FF9KEPROD3        Television_Name Electronic>>Television  2500    1500   Television_Description   testTraderName
EXPEH2FF9KEPROD1        Notebook_Name   Electronic>>Computer>>Notebook  10000  6000     Notebook_Description    testTraderName
EXPEH2FF9KEPROD2        Camera_Name     Electronic>>Camera      4000    3000   Camera_Description       testTraderName
Found '4' products. Trader Name: testTraderName
```

Test Case 13: In the picture below, customer gets all products according to the trader's name and sees the price in decreasing order.

```
sorted by price
---Display Products of trader--
EXPEH2FF9KEPROD1        Notebook_Name   Electronic>>Computer>>Notebook  10000  6000     Notebook_Description    testTraderName
EXPEH2FF9KEPROD2        Camera_Name     Electronic>>Camera      4000    3000   Camera_Description       testTraderName
EXPEH2FF9KEPROD3        Television_Name Electronic>>Television  2500    1500   Television_Description   testTraderName
EXPEH2FF9KEPROD4        updated_Headphones_Name Electronic>>Headphones  600    400      Headphones_Description  testTraderName
Found '4' products. Trader Name: testTraderName
```

Test Case 14: In the picture below, customer gets all products according to the trader's name and sees the discount amount in decreasing order.

```
sorted by discount amount
---Display Products of trader--
EXPEH2FF9KEPROD1        Notebook_Name   Electronic>>Computer>>Notebook  10000  6000     Notebook_Description    testTraderName
EXPEH2FF9KEPROD2        Camera_Name     Electronic>>Camera      4000    3000   Camera_Description       testTraderName
EXPEH2FF9KEPROD3        Television_Name Electronic>>Television  2500    1500   Television_Description   testTraderName
EXPEH2FF9KEPROD4        updated_Headphones_Name Electronic>>Headphones  600    400      Headphones_Description  testTraderName
Found '4' products. Trader Name: testTraderName
```

Test Case 15: All products of the trader are brought in according to the trader’s name and filtering can be done.

The picture above is filtered by category. Electronic categories are shown.

```
---Filter Operations--
sorted by name
---Display Products of trader--
EXPEH2FF9KEPROD4        updated_Headphones_Name Electronic>>Headphones  600    400      Headphones_Description  testTraderName
EXPEH2FF9KEPROD3        Television_Name Electronic>>Television  2500    1500   Television_Description   testTraderName
EXPEH2FF9KEPROD1        Notebook_Name   Electronic>>Computer>>Notebook  10000  6000     Notebook_Description    testTraderName
EXPEH2FF9KEPROD2        Camera_Name     Electronic>>Camera      4000    3000   Camera_Description       testTraderName
Found '4' products. Trader Name: testTraderName
```

Test Case 16: All of the trader's products are fetching by the trader name and filtered by the lower bound. Filtered by the lower bound of the trader’s products, and products that are equal to or higher than the lower bound are shown. Here, the lower bound has been entered as 2500.

```
sorted by name
---Display Products of trader--
EXPEH2FF9KEPROD3        Television_Name Electronic>>Television  2500    1500   Television_Description   testTraderName
EXPEH2FF9KEPROD1        Notebook_Name   Electronic>>Computer>>Notebook  10000  6000     Notebook_Description    testTraderName
EXPEH2FF9KEPROD2        Camera_Name     Electronic>>Camera      4000    3000   Camera_Description       testTraderName
Found '3' products. Trader Name: testTraderName
```

Test Case 17: All of the trader's products are fetching by the trader name and filtered by the upper bound. Filtered by the upper bound of the trader's products, and products that are equal to or lower than the upper bound are shown. Here, the upper bound has been entered as 2500.

```
sorted by name
---Display Products of trader--
EXPEH2FF9KEPROD4        updated_Headphones_Name Electronic>>Headphones  600    400      Headphones_Description  testTraderName
EXPEH2FF9KEPROD3        Television_Name Electronic>>Television  2500    1500   Television_Description   testTraderName
Found '2' products. Trader Name: testTraderName
```

Test Case 18: All of the trader's products are fetching by the trader name and filtered by the range bound. Filtered by the range bound of the trader's products. Here, the upper bound has been entered as 400 and lower bound has been entered 500.

- In all filtering processes, products are listed in descending order of product name

```
sorted by name
---Display Products of trader--
EXPEH2FF9KEPROD4        updated_Headphones_Name Electronic>>Headphones  600    400      Headphones_Description  testTraderName
EXPEH2FF9KEPROD3        Television_Name Electronic>>Television  2500    1500   Television_Description   testTraderName
EXPEH2FF9KEPROD2        Camera_Name     Electronic>>Camera      4000    3000   Camera_Description       testTraderName
Found '3' products. Trader Name: testTraderName
```

Test Case 19: The customer can search for a product based on a word in the product name or description. The search word in the picture below is '_Name'. Products with the word _Name in the product name or description are shown. By default, products are listed in descending order of product name.

```
sorted by name
---Display Products by name or description---
EXPEH2FF9KEPROD4        updated_Headphones_Name Electronic>>Headphones  600    400      Headphones_Description  testTraderName
EXPEH2FF9KEPROD3        Television_Name Electronic>>Television  2500    1500   Television_Description   testTraderName
EXPEH2FF9KEPROD1        Notebook_Name   Electronic>>Computer>>Notebook  10000  6000     Notebook_Description    testTraderName
EXPEH2FF9KEPROD2        Camera_Name     Electronic>>Camera      4000    3000   Camera_Description       testTraderName
Found '4' products. Searched Word: _Name
```

Test Case 20: Products with the word _Name in the product name or description are shown. The products shown are shown in descending order according to price.

```
sorted by price
---Display Products by name or description---
EXPEH2FF9KEPROD1        Notebook_Name   Electronic>>Computer>>Notebook  10000  6000     Notebook_Description    testTraderName
EXPEH2FF9KEPROD2        Camera_Name     Electronic>>Camera      4000    3000   Camera_Description       testTraderName
EXPEH2FF9KEPROD3        Television_Name Electronic>>Television  2500    1500   Television_Description   testTraderName
EXPEH2FF9KEPROD4        updated_Headphones_Name Electronic>>Headphones  600    400      Headphones_Description  testTraderName
Found '4' products. Searched Word: _Name
```

Test Case 21: Products with the word _Name in the product name or description are shown. The products shown are shown in descending order according to discount amount.

```
sorted by discount amount
---Display Products by name or description---
EXPEH2FF9KEPROD1        Notebook_Name    Electronic>>Computer>>Notebook  10000  6000     Notebook_Description    testTraderName
EXPEH2FF9KEPROD2        Camera_Name      Electronic>>Camera      4000    3000   Camera_Description       testTraderName
EXPEH2FF9KEPROD3        Television_Name Electronic>>Television  2500    1500   Television_Description   testTraderName
EXPEH2FF9KEPROD4        updated_Headphones_Name Electronic>>Headphones  600    400      Headphones_Description  testTraderName
Found '4' products. Searched Word: _Name
```

Test Case 22:A filter has been applied to products with _Name in the product name or description. The filter applied to these products is on a category basis. The electronic category is shown.

```
---Filter Operations--
sorted by name
---Display Products by name or description---
EXPEH2FF9KEPROD4        updated_Headphones_Name Electronic>>Headphones  600    400      Headphones_Description  testTraderName
EXPEH2FF9KEPROD3        Television_Name Electronic>>Television  2500    1500   Television_Description   testTraderName
EXPEH2FF9KEPROD1        Notebook_Name    Electronic>>Computer>>Notebook  10000  6000     Notebook_Description    testTraderName
EXPEH2FF9KEPROD2        Camera_Name      Electronic>>Camera      4000    3000   Camera_Description       testTraderName
Found '4' products. Searched Word: _Name
```

Test Case 23: Filtered products with _Name in product name or description. The filter applied to these products is relative to the lower bound. Products filtered according to the lower bound of products and equal to or higher than the lower bound are shown. Here, the lower bound has been entered as 2500.

```
sorted by name
---Display Products by name or description---
EXPEH2FF9KEPROD3        Television_Name Electronic>>Television  2500    1500   Television_Description   testTraderName
EXPEH2FF9KEPROD1        Notebook_Name    Electronic>>Computer>>Notebook  10000  6000     Notebook_Description    testTraderName
EXPEH2FF9KEPROD2        Camera_Name      Electronic>>Camera      4000    3000   Camera_Description       testTraderName
Found '3' products. Searched Word: _Name
```

Test Case 24: Filtered products with _Name in product name or description. The filter applied to these products is relative to the upper bound. Products filtered according to the upper bound of products and equal to or lower than the upper bound are shown. Here, the upper bound has been entered as 2500.

```
sorted by name
---Display Products by name or description---
EXPEH2FF9KEPROD4        updated_Headphones_Name Electronic>>Headphones  600    400      Headphones_Description  testTraderName
EXPEH2FF9KEPROD3        Television_Name Electronic>>Television  2500    1500   Television_Description   testTraderName
Found '2' products. Searched Word: _Name
```

Test Case 25:Filtered products with _Name in product name or description. The filter applied to these products is relative to the range bound. Here, the upper bound has been entered as 400 and lower bound has been entered 500.

```
sorted by name
---Display Products by name or description---
EXPEH2FF9KEPROD4        updated_Headphones_Name Electronic>>Headphones  600    400      Headphones_Description  testTraderName
EXPEH2FF9KEPROD3        Television_Name Electronic>>Television  2500    1500   Television_Description   testTraderName
EXPEH2FF9KEPROD2        Camera_Name     Electronic>>Camera      4000    3000   Camera_Description       testTraderName
Found '3' products. Searched Word: _Name
```

Test Case 26: Customer can add orders. In the picture above, the customer has added two products.

```
New order added. ProductId: 'EXPEH2FF9KEPROD1' CustomerId: '50000001' ProductId: 'EXPEH2FF9KEPROD1'
New order added. ProductId: 'EXPEH2FF9KEPROD4' CustomerId: '50000001' ProductId: 'EXPEH2FF9KEPROD4'
```

Test Case 27:Re-entry has been made with the trader role. With this role, customers' orders will be displayed and transactions will be made on the orders.

```
----Signed in---- User id: 50000000

Trader : testTraderName
1: Add/Remove/Edit Products
2: See list of order
3: Accept/Cancel Orders
```

Test Case 28:The trader can view the customer's orders. Here, the orders of the customer. This customer does not have an order.

```
---Display Orders By Customers---
Found '0' orders. TraderId: 50000000
```

Test Case 29: Here, the orders of the customer whose customer id is 50000001 are shown. This customer has two orders.

```
---Display Orders By Customers---
50000000        50000001        EXPEH2FF9KEPROD1        ADDED
50000000        50000001        EXPEH2FF9KEPROD4        ADDED
Found '2' orders. TraderId: 50000000
```

Test Case 30: The trader can accept the customer's order. If it accepts successfully, "Order accepted" message is displayed.

```
Order accepted. ProductId: 'EXPEH2FF9KEPROD1'
```

Test Case 31: The trader can cancel the customer's order. If canceled successfully, "Order canceled" message is displayed.

```
Order cancelled. ProductId: 'EXPEH2FF9KEPROD4'
```

Test Case 32:The trader can view the customer's orders. Orders have been processed. Here, it indicates that a product has been accepted and a product has been canceled.