6/10/2020

Shelly

DipIt-61 (CT0338949)

Canned Food Inventory System

Brief Explanation

Table of Contents

[1. Functions Implemented 1](#_Toc42718951)

[1.1. Insert a new canned food (insertAnewCannedFood()) 1](#_Toc42718952)

[1.2. List canned food Information (listAnewCannedFood()) 2](#_Toc42718953)

[1.3. Remove a canned food (removeACannedFood()) 4](#_Toc42718954)

[1.4. Exit and Save (saveDataBackToFile) 5](#_Toc42718955)

[2. UML Diagram 7](#_Toc42718956)

[2.1. Bin Class UML Diagram 7](#_Toc42718957)

[2.2. MyDate Class UML Diagram 7](#_Toc42718958)

[2.3. CannedFood Class UML Diagram 7](#_Toc42718959)

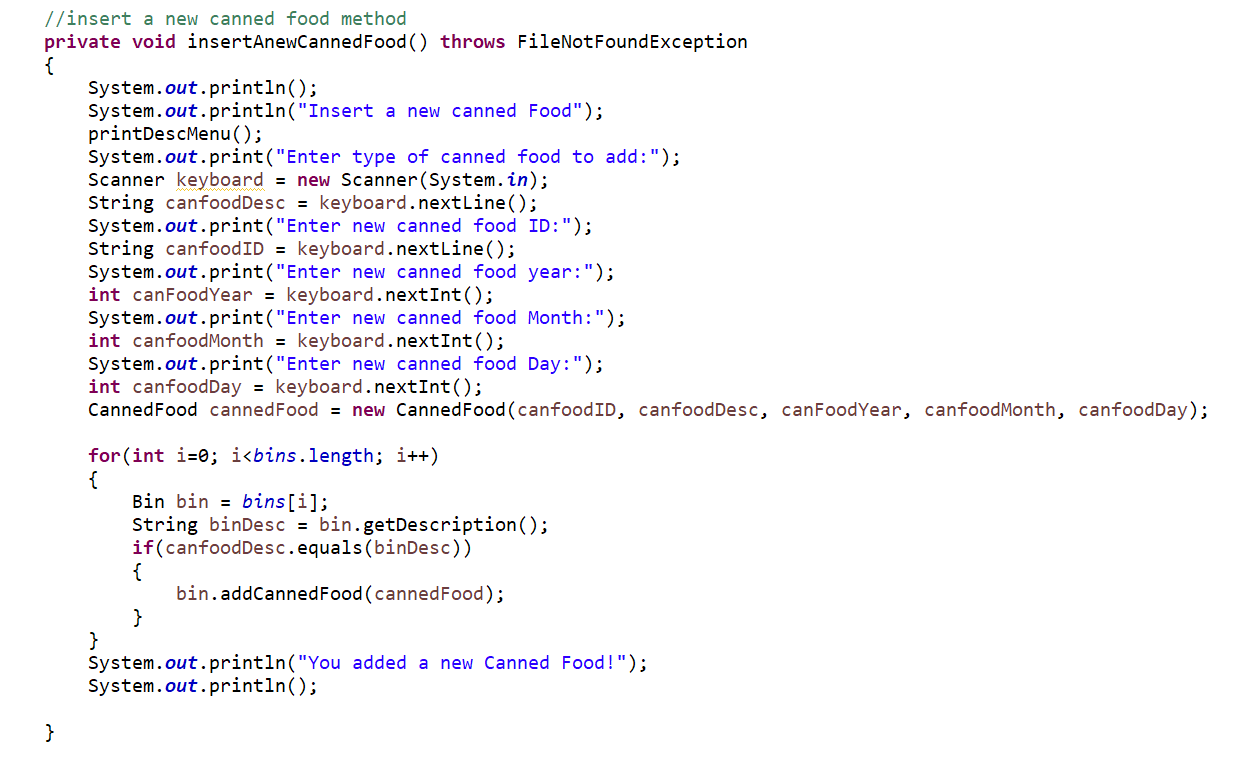
[2.4. CannedFoodWareHouse Class UML Diagram 8](#_Toc42718960)

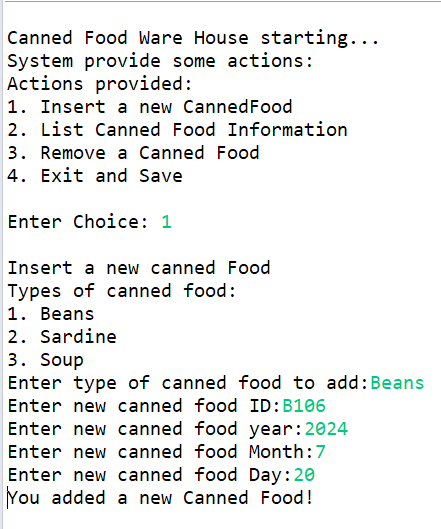
[3. Short Reflection 8](#_Toc42718961)

## Functions Implemented

According to the requirements of Canned Food Ware House Class there are several functions that have been implemented.

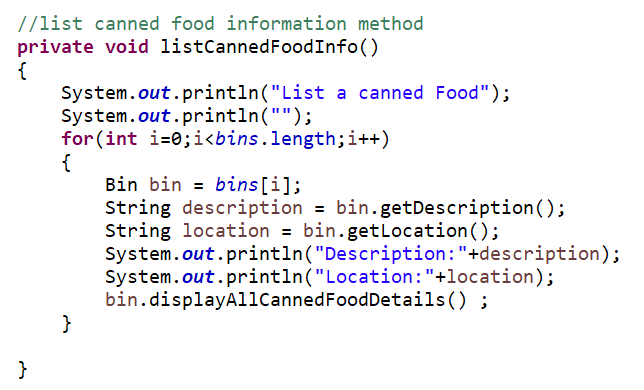
### Insert a new canned food (insertAnewCannedFood())

 This method allows the user to insert a new canned food into the bin. Firstly, this program will ask the user to enter what type of canned food user want to insert, and the user must type in the description they want to select. Secondly, the program will ask the user to enter an ID and Expiry Date of new canned food. Lastly, the canned food will be stored inside the bin chosen. Here is the code:

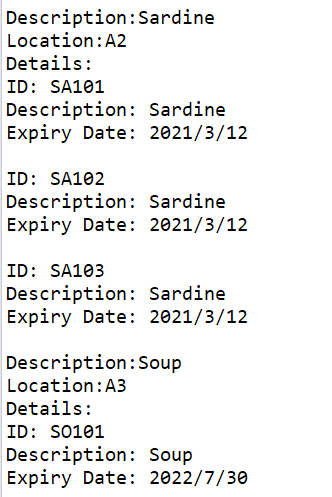
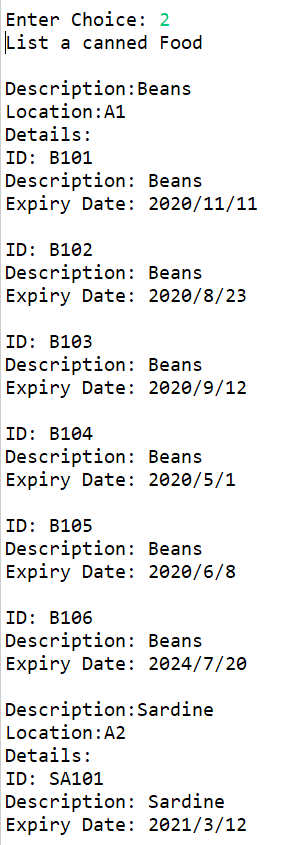
Here is the output printed in the console:

### List canned food Information (listAnewCannedFood())

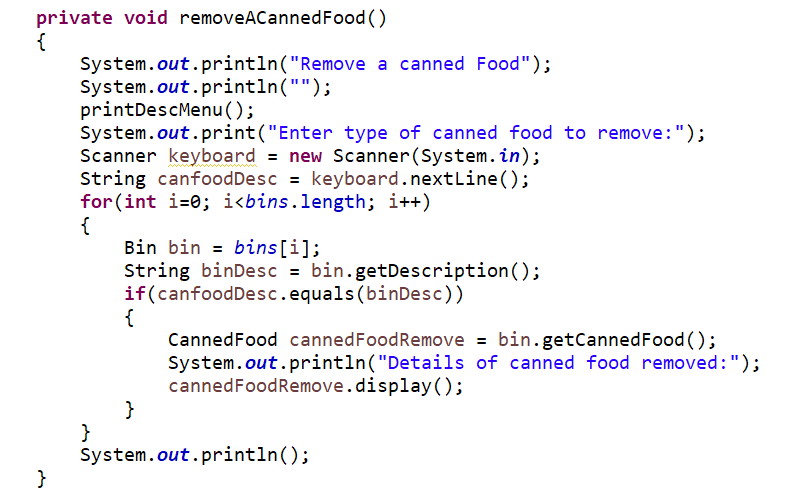
This method helps the user to find out the stock level of canned food inside the bin by listed all the canned food information. This function will display each bin (description and location) followed by each canned food details (ID, description, expiry date) that stored inside each bin. Here is the code:



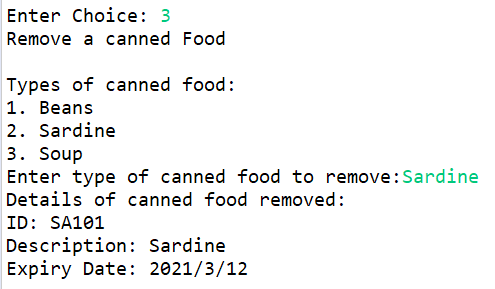
Below is the output display:



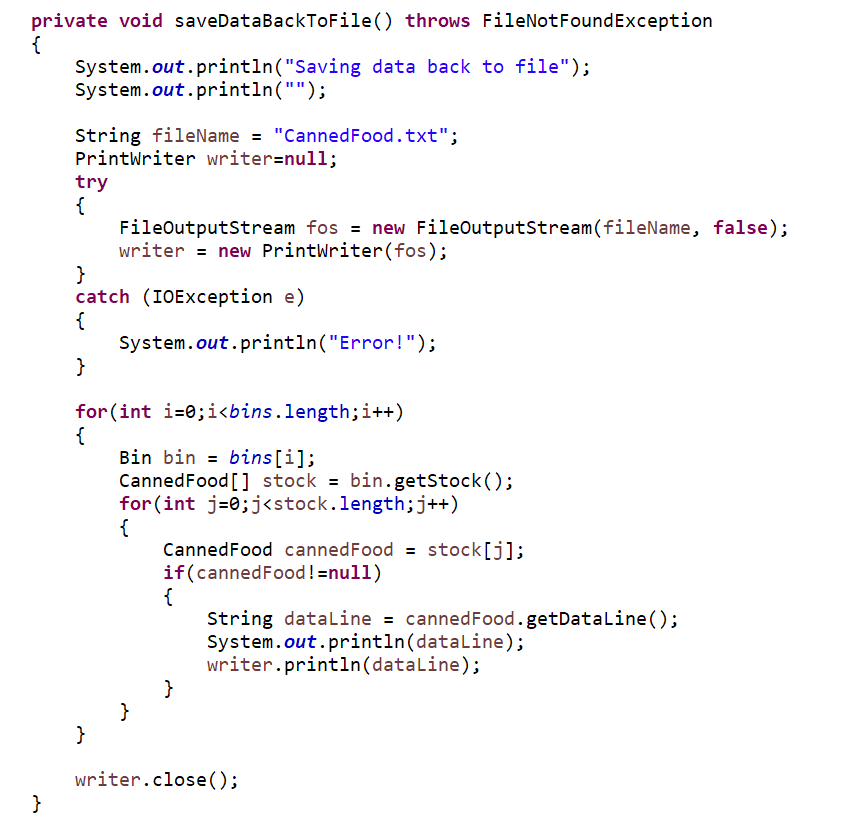
### Remove a canned food (removeACannedFood())

 This function allows the user to remove canned food from bin. First, the user needs to enter what type of canned food to be removed. After that, it will automatically remove a canned food out from the selected bin. Lastly, this method also will display the canned food detail that been removed.

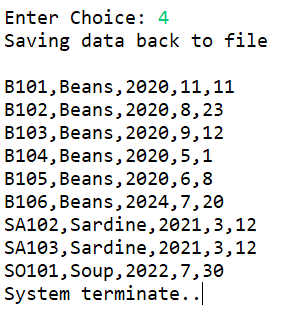
Here is the output:

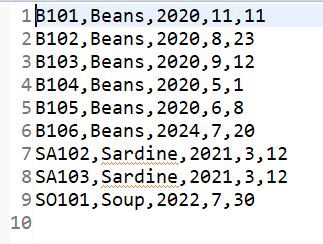


### Exit and Save (saveDataBackToFile)

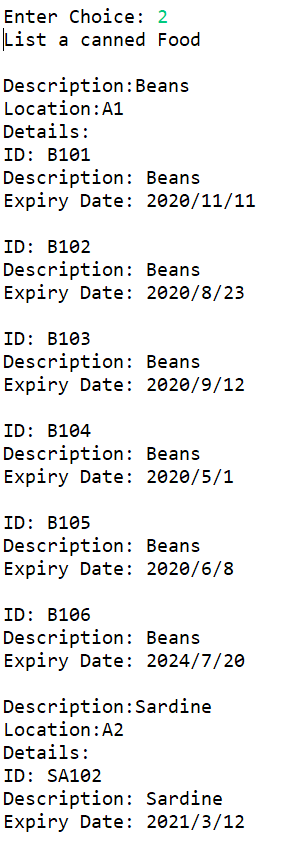
 This is the last function which will help the user to show the updated data based on what the user has modified. For example, in this case, user inserts (B106, Beans, 2024, 7, 20) and remove a can of Sardine.

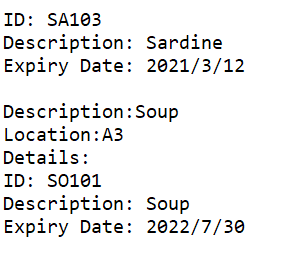
Here is the output:



After that, the data that has been updated will automatically overwrite in the CannedFood.txt

When the user opens the file again, the user will operate with the updated data. To prove this, I run my program again to list the new canned food information.





The output is exactly the same as the updated CannedFood.txt.

## UML Diagram

### Bin Class UML Diagram

|  |
| --- |
| Bin |
| - location: String  - description: String  - stock: CannedFood[] |
| +Bin(String location, String description)  +getDescription(): String  +getLocation(): String  +displayAllCannedFoodDetails(): void  +getStock(): CannedFood[]  +addCannedFood(CannedFood cannedFood): boolean +getCannedFood(): CannedFood |

### MyDate Class UML Diagram

|  |
| --- |
| MyDate |
| - year: int  - month: int  - day: int |
| +MyDate(int year, int month, int day)  +display(): void  +getDateData(): String  +isBefore(): boolean |

### CannedFood Class UML Diagram

### 

|  |
| --- |
| CannedFood |
| - ID: String  - description: String  - expiryDate: MyDate |
| +CannedFood(String ID, String description, int year, int month, int day)  +display(): void  +getDataLine(): String  +isExpired(): boolean |

### CannedFoodWareHouse Class UML Diagram

|  |
| --- |
| CannedFoodWareHouse |
| - bins: Bin[] |
| +CannedFoodWareHouse()  - loadFileToBinArray: void +printMenu(): void  +getChoice(): int  +printDescMenu(): void  - insertACannedFood(): void  - listCannedFoodInfo(): void  - removeACannedFood(): void  - saveDataBackToFile(): void  - display(): void |

## Short Reflection

After finish doing this assignment, I realise that giving a meaningful variable name is very important especially for medium to a large project. Give a meaningful variable name can help me to reduce the duration of finding the code. Because I don’t get confused while calling the other method. Other than that understanding the problem and break it into smaller parts make me easier to find the solution. As a beginner, Object-oriented topic is a bit challenging for me but through this assignment, I have improved my knowledge about Java Programming. All in all, I have learned a lot about java programming since this term and would love to explore more into this field.