tp

FridgeFriend is an app for managing food in the fridge that is optimised for use via a Command Line Interface (CLI) and is targeted at new homeowners who cook. If you can type fast, `FridgeFriend` can track your cold or frozen groceries faster and easier than any other apps. It is written in Java, and has more than 6000 lines of code.

View the Project on GitHub at https://github.com/

User Guide

Introduction

`FridgeFriend` is an app for managing food in the fridge that is optimised for use via a Command Line Interface (CLI) and is targeted at new homeowners who cook. If you can type fast, `FridgeFriend` can track your cold or frozen groceries faster and easier than any other apps. It is written in Java, and has more than 6000 lines of code.

Contents

Quick Start

1. Ensure that you have Java 11 or above installed.
2. Download the latest version of `FridgeFriend` from here.
3. Copy the file to the folder you want to use as the home folder for your `FridgeFriend`.
4. Open your Command Line Terminal in the folder where the `jar` file is located, and run `FridgeFriend` with `java -jar [CS2113-T10-1][FridgeFriend].jar`.
5. Type the command in the command box and press Enter to execute it. For example, typing `list` and pressing Enter will show a list of all current food. Some example commands you can try:

   - `help`: Displays a help message
   - `list`: Lists all food.
   - `bye`: Exits the app.

6. Refer to the **Features** below for details of each command.

⚠️ Do ***NOT*** modify any of text files in the `/data` folder in order to prevent errors and loss of data.

Features

ℹ️ Words in UPPER_CASE are the parameters to be supplied by the user. For example, `add FOOD_NAME` should be executed as `add chicken` where `FOOD_NAME` is the parameter.

ℹ️ Extraneous parameters for commands that do not take in parameters will be ignored. For example, `help me` will execute as `help`.

Get help message: `help`

Prints a list of available commands and formats.

Format: `help`

Example of usage:

```
>> help
These are the list of available commands:
        help
        add foodName /cat categoryName /exp dd-mm-yyyy /loc storageLocation /qty foodQuantity
        list
        list categoryName
        list storageLocation
        remove foodName /qty removeQuantity
        search searchString
        expiring
        runninglow
        setlimit foodCategory /qty newLimit
        history
        history clear
        clear
        bye

This is the list of food categories:
        [VEGETABLE, FRUIT, MEAT, SEAFOOD, EGG, DAIRY, BEVERAGE, COOKED_DISH, READY_TO_EAT,
FROZEN, OTHER]

This is the list of storage locations:
        [FREEZER, UPPER_SHELF, MIDDLE_SHELF, LOWER_SHELF, DRAWERS, FRIDGE_DOOR, OTHER]
```

Adding a food item: `add`

Adds a food item into the fridge.

If a particular `FOOD_NAME` is in the fridge, the other fields have to be same in order to add the quantity. Otherwise, a unique `FOOD_NAME` has to be used to add the food into the FridgeFriend.

Format: `add FOOD_NAME /cat FOOD_CATEGORY /exp EXPIRY_DATE /loc LOCATION_IN_THE_FRIDGE /qty FOOD_QUANTITY`

- The `FOOD_NAME` is the name of a food but not an empty description.
- The `FOOD_CATEGORY` must be one of the available food categories.
- The `EXPIRY_DATE` must be in the format `dd-mm-yyyy`.
- The `LOCATION_IN_THE_FRIDGE` must be one of the available storage locations.
- The `FOOD_QUANTITY` must be a positive integer.

💡 **Tips:**

- If you want to add more to the same batch of food (same category, same location and same expiry date), you should specify exactly the same `FOOD_NAME`,`FOOD_CATEGORY`,`EXPIRY_DATE`, `LOCATION_IN_THE_FRIDGE` and the new quantity in `QUANTITY` field.
- The food names should not repeat unless it is the same batch as described above. Otherwise, you will be prompted to retry the `add` command.

ℹ️ Additional info:

- Available Food Categories: `MEAT`, `SEAFOOD`, `EGG`, `DAIRY`, `VEGETABLE`, `FRUIT`, `BEVERAGE`, `COOKED_DISH`, `READY_TO_EAT`, `FROZEN`, `OTHER`
- Available Fridge Location: `FREEZER`, `UPPER_SHELF`, `MIDDLE_SHELF`, `LOWER_SHELF`, `DRAWERS`, `FRIDGE_DOOR`, `OTHER`
- The list of available `FOOD_CATEGORY` and `LOCATION_IN_THE_FRIDGE` can be found using the help command.

Example of usage:

```
>> add chicken /cat meat /exp 30-06-2021 /loc lower_shelf /qty 800
Great! I have added chicken into your fridge.
Details: Food name: chicken, category: MEAT, expiry: 30-06-2021, stored in: LOWER_SHELF,
quantity: 800

>> add chicken /cat meat /exp 07-10-2021 /loc freezer /qty 200
Sorry my friend, you have added this food before.
It is in a different category or location or have a different expiry date.
Please specify another foodname.

>> add chicken /cat meat /exp 30-06-2021 /loc lower_shelf /qty 200
Great! I have added chicken into your fridge.
Details: Food name: chicken, category: MEAT, expiry: 30-06-2021, stored in: LOWER_SHELF,
quantity: 1000

>> add pork /cat meat /exp 04-07-2021 /loc freezer /qty 500
Great! I have added pork into your fridge.
```

```
Great! I have added pork into your fridge.
Details: Food name: pork, category: MEAT, expiry: 04-07-2021, stored in: FREEZER, quantity: 500

>> add chicken wings /cat ready_to_eat /exp 17-04-2021 /loc upper_shelf /qty 500
Great! I have added chicken wings into your fridge.
Details: Food name: chicken wings, category: READY_TO_EAT, expiry: 17-04-2021, stored in:
UPPER_SHELF, quantity: 500

>> add lettuce /cat vegetable /exp 08-04-2021 /loc lower_shelf /qty 3
Great! I have added lettuce into your fridge.
Details: Food name: lettuce, category: VEGETABLE, expiry: 08-04-2021, stored in: LOWER_SHELF,
quantity: 3
```

Display the list of all foods: `list`

Displays the full list of food items stored in the fridge.

Format: `list`

- There should not be any descriptions after keyword `list`.

Example of usage:

```
>> list
Here are the items in your fridge:
        1. Food name: chicken, category: MEAT, expiry: 30-06-2021, stored in: LOWER_SHELF,
quantity: 1000
        2. Food name: pork, category: MEAT, expiry: 04-07-2021, stored in: FREEZER, quantity:
500
        3. Food name: chicken wings, category: READY_TO_EAT, expiry: 17-04-2021, stored in:
UPPER_SHELF, quantity: 500
        4. Food name: lettuce, category: VEGETABLE, expiry: 08-04-2021, stored in: LOWER_SHELF,
quantity: 3
```

Display the list of foods by category: `list <CATEGORY>`

Displays the list of food items under specified category stored in the fridge.

Format: `list CATEGORY_NAME`

- The `CATEGORY_NAME` can only be recognised if it is in the pre-defined categories: `VEGETABLE, FRUIT, MEAT, SEAFOOD, EGG, DAIRY, BEVERAGE, COOKED_DISH, READY_TO_EAT, FROZEN, OTHER`
- The `CATEGORY_NAME` is case-insensitive.
- If invalid input detected, `FridgeFriend` would give an error message.

Example of usage:

```
>> list meat
>> list meat
These are the MEAT in your fridge:
        1. Food name: chicken, category: MEAT, expiry: 30-06-2021, stored in: LOWER_SHELF,
quantity: 1000
        2. Food name: pork, category: MEAT, expiry: 04-07-2021, stored in: FREEZER, quantity:
500

>> list dairy
These are the DAIRY in your fridge:
```

Display the list of foods by storage location: `list <LOCATION>`

Displays the list of food items under specified category stored in the fridge.

Format: `list STORAGE_LOCATION_NAME`

- The `STORAGE_LOCATION_NAME` can only be recognised if it is in the pre-defined categories: `FREEZER, UPPER_SHELF, MIDDLE_SHELF, LOWER_SHELF, DRAWERS, FRIDGE_DOOR, OTHER`
- The `STORAGE_LOCATION_NAME` is case-insensitive.
- If invalid input detected, `FridgeFriend` would give an error message.

Example of usage:

```
>> list freezer
These are the food stored in FREEZER:
        1. Food name: pork, category: MEAT, expiry: 04-07-2021, stored in: FREEZER, quantity:
500

>> list drawers
These are the food stored in DRAWERS:
```

Remove a food item: `remove`

Removes a food item by quantity. Additionally, if a removal results in a food category falling below the specified limit (as per in `setlimit`), this command will also give a warning that the category is `runninglow` on food.

Format: `remove FOODNAME /qty QUANITTY_TO_REMOVE`

- The `FOODNAME` must be an existing food name (same with the first parameter during `add`) in the fridge.
  - If the `FOODNAME` is not found, `FridgeFriend` will give an error message.

- The `QUANTITY_TO_REMOVE` must be lower than or equal to the current quantity of the food.
  - If the `QUANTITY_TO_REMOVE` is larger than current quantity of the food, `FridgeFriend` will give an error message.

    - If the `QUANTITY_TO_REMOVE` is equal to current quantity of the food, it means food is depleted and thus the whole food item will be deleted by `FridgeFriend` (it will not appear on list either).
    - If the `QUANTITY_TO_REMOVE` is lower than current quantity of the food, it means there is still some food left of this item. The quantity of food will be updated.

Example of usage:

```
>> list
Here are the items in your fridge:
	1. Food name: chicken, category: MEAT, expiry: 30-06-2021, stored in: LOWER_SHELF,
quantity: 1000
	2. Food name: pork, category: MEAT, expiry: 04-07-2021, stored in: FREEZER, quantity:
500
	3. Food name: chicken wings, category: READY_TO_EAT, expiry: 17-04-2021, stored in:
UPPER_SHELF, quantity: 500
	4. Food name: lettuce, category: VEGETABLE, expiry: 08-04-2021, stored in: LOWER_SHELF,
quantity: 3

>> remove chicken /qty 200
Noted! I've removed 200 of the food chicken from your fridge.
New quantity: 800

>> remove pork /qty 700
Not enough in fridge to remove!

>> remove pork /qty 500
Noted! I've removed pork from your fridge.
Now you have 3 food in the fridge.

>> remove chicken /qty 500
Noted! I've removed 500 of the food chicken from your fridge.
New quantity: 300
WARNING! You are running low on MEAT
Total MEAT quantity: 300
```

Search: `search`

Checks if a certain food item is inside the fridge. If the item is found, `FridgeFriend` will inform the user of the food items that matches the search query.

Format: `search FOOD_NAME`

- The `FOOD_NAME` can be a partial or full name of a food item but not an empty description.
- If there are multiple food items which names contain the same partial name, the search function will display all of those food items.

Example of usage:

```
>> search chicken
These are the chicken in your fridge:
        1. Food name: chicken, category: MEAT, expiry: 30-06-2021, stored in: LOWER_SHELF,
quantity: 300
        2. Food name: chicken wings, category: READY_TO_EAT, expiry: 17-04-2021, stored in:
UPPER_SHELF, quantity: 500

>> search oyster
You do not have oyster in your fridge.
```

List expiring foods: `expiring`

Displays a list of food items that are expiring within a week. An additional message will be displayed, if there are food items that have already expired.

Format: `expiring`

Example of usage:

```
>> expiring
These are the food expiring in the next week:
1. Food name: chicken wings, category: READY_TO_EAT, expiry: 17-04-2021, stored in: UPPER_SHELF,
    quantity: 500

These are the food that has already expired, please consider removing them:
1. Food name: lettuce, category: VEGETABLE, expiry: 08-04-2021, stored in: LOWER_SHELF,
quantity: 3

>> remove lettuce /qty 3
Noted! I've removed lettuce from your fridge.
Now you have 2 food in the fridge.
WARNING! You are running low on VEGETABLE
Total VEGETABLE quantity: 0

>> expiring
These are the food expiring in the following week:


1. Food name: chicken wings, category: READY_TO_EAT, expiry: 17-04-2021, stored in: UPPER_SHELF,
quantity: 500
```

List categories with food running low: `runninglow`

Displays a list of food categories which total quantity is below a specified minimum limit This command also displays the existing quantities along with its limits for each category.

💡 These limits can be modified with the `setlimit` command.

If the user tries to run the `runninglow` command with all the food categories limit set to 0, `FridgeFriend` would prompt the user to set at least one of the food category to be more than 0.

Format: `runninglow`

Example of usage:

```
>> runninglow
You are running low on food in these categories:
1. VEGETABLE quantity: 0 out of 500
2. FRUIT quantity: 0 out of 500
3. SEAFOOD quantity: 0 out of 500
4. EGG quantity: 0 out of 500
5. DAIRY quantity: 0 out of 500
6. BEVERAGE quantity: 0 out of 500
7. COOKED_DISH quantity: 0 out of 500
8. FROZEN quantity: 0 out of 500
9. OTHER quantity: 0 out of 500
```

Modify the minimum quantity limits: `setlimit`

Changes the minimum quantity limit for a specific food category. To remove a limit for a category, set its limit to 0.

Format: `setlimit FOOD_CATEGORY /qty QUANTITY`

Example of usage:

```
>> runninglow
You are running low on food in these categories:
1. VEGETABLE quantity: 0 out of 500
2. FRUIT quantity: 0 out of 500
3. SEAFOOD quantity: 0 out of 500
4. EGG quantity: 0 out of 500
5. DAIRY quantity: 0 out of 500
6. BEVERAGE quantity: 0 out of 500
7. COOKED_DISH quantity: 0 out of 500
8. FROZEN quantity: 0 out of 500
9. OTHER quantity: 0 out of 500

>> setlimit ready_to_eat /qty 600
Okie dokie! The new minimum quantity for category 'READY_TO_EAT' is 600

>> setlimit vegetable /qty 0
Okie dokie! The new minimum quantity for category 'VEGETABLE' is 0

>> runninglow
You are running low on food in these categories:
1. FRUIT quantity: 0 out of 500
2. SEAFOOD quantity: 0 out of 500
3. EGG quantity: 0 out of 500
```

```
4. DAIRY quantity: 0 out of 500
5. BEVERAGE quantity: 0 out of 500
6. COOKED_DISH quantity: 0 out of 500
7. READY_TO_EAT quantity: 500 out of 600
8. FROZEN quantity: 0 out of 500
9. OTHER quantity: 0 out of 500
```

List history of items added: `history`

Displays a history of food items that have been added to `FridgeFriend`. `FridgeFriend` keeps track of all food items added in automatically. Unlike using the add command, the history command records each addition separately. Thus, the user can use this command to keep track of all occurrences where food has been added to `FridgeFriend`.

ℹ️ The data is saved to disk in a text file, with default location as `data/historyData.txt`.

⚠️ Do NOT modify the data file as this might lead to corruption and loss of the data.

⚠️ In the event that the data in the text file is corrupted or in an unreadable format, the `history` command may fail to output the contents of the file. Users may have to manually inspect the file to delete the invalid content, or wipe the contents of the file with the `history clear` command, to resume normal function. The execution of the `FridgeFriend` program, however, will not be interrupted.

Format: `history`

Example of usage:

```
>> history
This is the full history of items you've added in the fridge:
        1. Food name: chicken, category: MEAT, expiry: 30-06-2021, stored in: LOWER_SHELF,
quantity: 800
        2. Food name: chicken, category: MEAT, expiry: 30-06-2021, stored in: LOWER_SHELF,
quantity: 1000
        3. Food name: pork, category: MEAT, expiry: 04-07-2021, stored in: FREEZER, quantity:
500
        4. Food name: chicken wings, category: READY_TO_EAT, expiry: 09-04-2021, stored in:
UPPER_SHELF, quantity: 500
        5. Food name: lettuce, category: VEGETABLE, expiry: 08-04-2021, stored in: LOWER_SHELF,
quantity: 3
```

Clear list history of items added: `history clear`

Wipes the data from the history text file.

⚠️ This command will ***permanently*** erase the entire history log and this action ***cannot*** be reversed.

Format: `history clear`

Example of usage:

```
>> history clear
History successfully cleared!

>> history
This is the full history of items you've added in the fridge:
```

Empty fridge: `clear`

Removes all food items in the fridge.

⚠️ This command will ***permanently*** empty the fridge and this action ***cannot*** be reversed.

Format `clear`

Example of usage:

```
>> list
Here are the items in your fridge:
        1. Food name: chicken, category: MEAT, expiry: 30-06-2021, stored in: LOWER_SHELF,
quantity: 300
        2. Food name: chicken wings, category: READY_TO_EAT, expiry: 09-04-2021, stored in:
UPPER_SHELF, quantity: 500

>> clear
Fridge has been cleared!

>> list
Here are the items in your fridge:
```

Exit the application: `bye`

Exits the application.

Format: `bye`

Example of usage:

```
>> bye
Bye! Hope to see you again soon!
```

FAQ

Index	Question	Answer

Index	Question	Answer
1	How do I transfer my data to another computer?	Copy the `.jar` file along with `data` folder to the target computer and place them together into an empty folder. As long as the target computer satisfies the application prerequisites, it can execute with the saved data as before.
2	What if I forget the correct format of a command?	You will get a tip if you use any of the command keywords incorrectly. Plus, you are always welcome to use `help` command.
3	How do I report a bug in the appplication?	You can create a new issue here on the project repository. Do state the type of bugs as well as give a clear description of the bug supported with appropriate screenshots.

Command Summary

- Get help message `help`
- Add food `add FOOD_NAME /cat FOOD_CATEGORY /exp EXPIRY_DATE /loc LOCATION_IN_THE_FRIDGE /qty FOOD_QUANTITY`
- List food `list`
- List food by category `list CATEGORY_NAME`
- List food by storage location `list STORAGE_LOCATION_NAME`
- Remove food `remove FOODNAME /qty QUANITTY_TO_REMOVE`
- Search for food `search FOOD_NAME`
- List expiring foods `expiring`
- List categories with food running low: `runninglow`
- Modify the minimum quantity limits: `setlimit FOOD_CATEGORY /qty QUANTITY`
- List history of items added: `history`
- Clear list history of items added: `history clear`
- Empty fridge: `clear`
- Exit application `bye`