Version Information:

- Version: 2
- Last Edited: July 04, 2021
- Changelog:
  JM_AC_CET-CS

Version	Description
3	Added File I/O for data persistence, searching and sorting
2	Added methods that can be added to the classes and sample output
1	Initial Version

The Rules

1. You MUST use the **default** package in your assignment so that we can compile and run your program easily.
2. ALL data members **MUST** be declared as **private** (or protected in the Base class when necessary only).
3. Your program must be written using Object Oriented Principles (as outlined in the course notes and previous courses) and with efficiency in mind, you may lose marks for inefficient code.
4. Never leave compile warnings, address them without suppressing them!
5. A solution that does not compile will receive a grade of 0.

Problem Description

In this assignment, you will modify the solution to Assignment 1. You may start with your solution. You must also submit a test plan for the functionality created in this assignment. You must add to your test plan from assignment 1 to include testing the new functionality and addressing any feedback you have received. All requirements must still behave the same as previously described in Assignment 1 instructions unless overwritten by the requirements outlined here below.

Requirements

1. **Friendly Input:** Ensure that your solution accepts spaces in any string input.
2. **Efficiency:** You will update the *inventory* data structure in two ways:
   - Change the *inventory* declaration in the Inventory class to be an object of ArrayList.
   - Change the insert into the data structure so that the FoodItem objects are in numerical order from smallest to largest by itemCode in the most efficient way possible. Once the inventory structure is sorted, modify your other methods to also make them as efficient as possible. (Note – do NOT use Generic Collection methods yet – we will learn to do the coding ourselves first).
3. **Saving data:** You will add three menu selections to your assignment:
   - The first will read the contents of a file and add them to the Inventory::inventory ArrayList. (i.e. read).
   - And the second will save the current contents of the Inventory::inventory ArrayList to a file (i.e. write).
   - File format of both read and write is one line per piece of data, depending on what kind of resource the file is describing, various numbers of lines are required. Many types of FoodItems can be included in the same file.
   - Each resource will be in the following formats:

Fruit	**Vegetable**	**Preserve**
f item code item name quantity of item item cost item price name of the orchard	v item code item name quantity of item item cost item price farm supplier	p item code item name quantity of item item cost item price size of the jar

   - Upon read, if you find a duplicate itemCode, you can abort reading in the file. Valid FoodItems before that one should remain in the inventory and an appropriate error message should be displayed.

- You don’t know the name of the file so you cannot hard-code it in your program, you must prompt the user for the filename

4. **Search:** You will add the ability to view a FoodItem given the itemCode. Use binary search algorithm to implement the the search operation.
5. **Menu Update:** The new menu should look exactly like this:

```
Please select one of the following:
1: Add Item to Inventory
2: Display Current Inventory
3: Buy Item(s)
4: Sell Item(s)
5: Search for Item
6: Save Inventory to File
7: Read Inventory from File
8: To Exit
> <must be a number between 1 - 8>
```

6. When '5' is chosen by the user, the program will ask the user to enter the itemCode. See table for full details:

When code doesn't exist	`Enter the code for the item: <accept any integer>` `Code not found in inventory...`
When code is found, display details of the item associated to that code	`Enter the code for the item: <accept any integer, example 111>` `Item: 111 Granny Smith Apple 30 price: $0.25 cost: $0.10 orchard supplier: Apple Orchard`

7. When '6' is chosen by the user, the program will ask the user to enter the filename to save to. See table for full details:

```
Enter the filename to save to: <accept any string>
```

8. When '7' is chosen by the user, the program will ask the user to enter the filename to read from. Get the text file from here. See table for full details:

```
Enter the filename to read from: <accept any string>
```

9. When '8' is chosen by the user, the message "Exiting..." is displayed and the program terminates.
10. You can add the following methods to your classes to accomplish the above. If you need any other methods, they must be private helper methods.
    - Inventory:
        - Updated: `public boolean addItem(Scanner scanner, boolean fromFile)`
        - Added: `public void searchForItem(Scanner scanner)`
        - Added: `public void saveToFile(Scanner scanner)`
        - Added: `public void readFromFile(Scanner scanner)`
    - FoodItem:
        - Updated: `public boolean addItem(Scanner scanner, boolean fromFile)`
        - Updated: `public boolean inputCode(Scanner scanner, boolean fromFile)`
        - Added: `public int getItemCode()`
        - Added: `public void outputItem(Formatter writer)`
    - Fruit:
        - Updated: `public boolean addItem(Scanner scanner, boolean fromFile)`
        - Added: `public void outputItem(Formatter writer)`
    - Vegetable:
        - Updated: `public boolean addItem(Scanner scanner, boolean fromFile)`
        - Added: `public void outputItem(Formatter writer)`
    - Preserve:
        - Updated: `public boolean addItem(Scanner scanner, boolean fromFile)`
        - Added: `public void outputItem(Formatter writer)`
11. You must implement one of the comparison interfaces to compare FoodItems, either Comparator<T> or Comparable. If you are implementing the Comparator interface, you will have a new class, if you are

implementing Comparable, you will have a new method in one of your classes.

12. Remember, you must write **EFFICIENT** code. The changes above should allow you to improve the efficiency of some of the functionality from Assignment 1, that will be checked, ensure you take full advantage of the changes.

Submission [Via Activities menu/Assignments option]

You must submit to this assignment link in Brightspace by the due date and time. Your submission should follow the Submission Checklist

- All source code – i.e. .java files with headers
  - **NOTE**: we will re-compile and run your program….so all code must be available to us and **MUST NOT** be in a package.
  - Headers should contain name, student number, assignment number, date, purpose of class
- Javadoc:
  - A description of each data member and method should be described using Javadoc notation
  - Generate the Javadoc for your solution and upload a zip of the generated folder with a member visibility of Private
    - In Eclipse, go to Project->Generate Javadoc, select your project and "Create Javadoc for members with Visibility: Private" then click Finish.
- Test plan
  - Can be in word or pdf format
  - Should contain positive and negative testing (sometimes called happy path and unhappy path)
  - You should indicate which tests your program passes and which one fails

Failure to provide any of the above will have an effect on your grade for this assignment. The assignment rubric will be published with the assignment submission link by the end of the week.

Sample Output: green is user input

```
Please select one of the following:
1: Add Item to Inventory
2: Display Current Inventory
3: Buy Item(s)
4: Sell Item(s)
5: Search for Item
6: Save Inventory to File
7: Read Inventory from File
8: To Exit
> 2
Inventory:

Please select one of the following:
1: Add Item to Inventory
2: Display Current Inventory
3: Buy Item(s)
4: Sell Item(s)
5: Search for Item
6: Save Inventory to File
7: Read Inventory from File
8: To Exit
> 5
Enter the code for the item: 123
Code not found in inventory...
Please select one of the following:
1: Add Item to Inventory
2: Display Current Inventory
3: Buy Item(s)
4: Sell Item(s)
5: Search for Item
6: Save Inventory to File
7: Read Inventory from File
8: To Exit
```

```
> 1
Do you wish to add a fruit(f), vegetable(v) or a preserve(p)? f
Enter the code for the item: 111
Enter the name for the item: Granny Smith Apple
Enter the quantity for the item: 234
Enter the cost of the item: .10
Enter the sales price of the item: .25
Enter the name of the orchard supplier: Niagara Orchard
Please select one of the following:
1: Add Item to Inventory
2: Display Current Inventory
3: Buy Item(s)
4: Sell Item(s)
5: Search for Item
6: Save Inventory to File
7: Read Inventory from File
8: To Exit
> 1
Do you wish to add a fruit(f), vegetable(v) or a preserve(p)? v
Enter the code for the item: 222
Enter the name for the item: Hot House Tomato
Enter the quantity for the item: 500
Enter the cost of the item: .15
Enter the sales price of the item: .33
Enter the name of the farm supplier: McDonald's Farm
Please select one of the following:
1: Add Item to Inventory
2: Display Current Inventory
3: Buy Item(s)
4: Sell Item(s)
5: Search for Item
6: Save Inventory to File
7: Read Inventory from File
8: To Exit
> 1
Do you wish to add a fruit(f), vegetable(v) or a preserve(p)? p
Enter the code for the item: 112
Enter the name for the item: Strawberry Jam
Enter the quantity for the item: 35
Enter the cost of the item: 2.24
Enter the sales price of the item: 5
Enter the size of the jar in millilitres: 250
Please select one of the following:
1: Add Item to Inventory
2: Display Current Inventory
3: Buy Item(s)
4: Sell Item(s)
5: Search for Item
6: Save Inventory to File
7: Read Inventory from File
8: To Exit
> 2
Inventory:
Item: 111 Granny Smith Apple 234 price: $0.25 cost: $0.10 orchard supplier: Niagara Orchard
Item: 112 Strawberry Jam 35 price: $5.00 cost: $2.24 size: 250mL
Item: 222 Hot House Tomato 500 price: $0.33 cost: $0.15 farm supplier: McDonald's Farm

Please select one of the following:
1: Add Item to Inventory
2: Display Current Inventory
3: Buy Item(s)
4: Sell Item(s)
5: Search for Item
6: Save Inventory to File
7: Read Inventory from File
8: To Exit
> 6
Enter the filename to save to: MyInventory.txt
Please select one of the following:
1: Add Item to Inventory
```

```
2: Display Current Inventory
3: Buy Item(s)
4: Sell Item(s)
5: Search for Item
6: Save Inventory to File
7: Read Inventory from File
8: To Exit
> 7
Enter the filename to read from: MyInventory.txt
Item code already exists
Error Encountered while reading the file, aborting...
Please select one of the following:
1: Add Item to Inventory
2: Display Current Inventory
3: Buy Item(s)
4: Sell Item(s)
5: Search for Item
6: Save Inventory to File
7: Read Inventory from File
8: To Exit
> 2
Inventory:
Item: 111 Granny Smith Apple 234 price: $0.25 cost: $0.10 orchard supplier: Niagara Orchard
Item: 112 Strawberry Jam 35 price: $5.00 cost: $2.24 size: 250mL
Item: 222 Hot House Tomato 500 price: $0.33 cost: $0.15 farm supplier: McDonald's Farm

Please select one of the following:
1: Add Item to Inventory
2: Display Current Inventory
3: Buy Item(s)
4: Sell Item(s)
5: Search for Item
6: Save Inventory to File
7: Read Inventory from File
8: To Exit
> 5
Enter the code for the item: 333
Code not found in inventory...
Please select one of the following:
1: Add Item to Inventory
2: Display Current Inventory
3: Buy Item(s)
4: Sell Item(s)
5: Search for Item
6: Save Inventory to File
7: Read Inventory from File
8: To Exit
> 5
Enter the code for the item: 112
Item: 112 Strawberry Jam 35 price: $5.00 cost: $2.24 size: 250mL
Please select one of the following:
1: Add Item to Inventory
2: Display Current Inventory
3: Buy Item(s)
4: Sell Item(s)
5: Search for Item
6: Save Inventory to File
7: Read Inventory from File
8: To Exit
> 7
Enter the filename to read from: NoFile.txt
File Not Found, ignoring...
Please select one of the following:
1: Add Item to Inventory
2: Display Current Inventory
3: Buy Item(s)
4: Sell Item(s)
5: Search for Item
6: Save Inventory to File
7: Read Inventory from File
```

```
8: To Exit
> 7
Enter the filename to read from: MyInventory2.txt
Please select one of the following:
1: Add Item to Inventory
2: Display Current Inventory
3: Buy Item(s)
4: Sell Item(s)
5: Search for Item
6: Save Inventory to File
7: Read Inventory from File
8: To Exit
> 2
Inventory:
Item: 100 Barlett Pear 100 price: $0.25 cost: $0.10 orchard supplier: Niagara Orchard
Item: 111 Granny Smith Apple 234 price: $0.25 cost: $0.10 orchard supplier: Niagara Orchard
Item: 112 Strawberry Jam 35 price: $5.00 cost: $2.24 size: 250mL
Item: 212 Raspberry Jam 30 price: $5.50 cost: $2.28 size: 250mL
Item: 222 Hot House Tomato 500 price: $0.33 cost: $0.15 farm supplier: McDonald's Farm
Item: 311 McIntosh Apple 200 price: $0.25 cost: $0.10 orchard supplier: Niagara Orchard
Item: 322 Vine Ripe Tomato 100 price: $0.35 cost: $0.10 farm supplier: McDonald's Farm

Please select one of the following:
1: Add Item to Inventory
2: Display Current Inventory
3: Buy Item(s)
4: Sell Item(s)
5: Search for Item
6: Save Inventory to File
7: Read Inventory from File
8: To Exit
> 5
Enter the code for the item: 100
Item: 100 Barlett Pear 100 price: $0.25 cost: $0.10 orchard supplier: Niagara Orchard
Please select one of the following:
1: Add Item to Inventory
2: Display Current Inventory
3: Buy Item(s)
4: Sell Item(s)
5: Search for Item
6: Save Inventory to File
7: Read Inventory from File
8: To Exit
> 8
Exiting...
```