Academic Honesty Rules for This Assignment

This assignment is individual work. You may:

- Use the textbook and class videos, including code in the textbook and videos.
- Use the documentation at docs.oracle.com/en/java/ as a resource. Do not borrow code from the documentation.
- Ask me questions.

Discussing this assignment with any other person (in person or on the Internet), searching for answers on the Internet, or using sources other than the ones listed above is a violation of the class academic honesty policy. Please ask if you have any questions about this.

Specifications

Problem Description

You are going to write a program that reads U.S. commercial fishing data from a file. It will store the data in an array of `CommercialFish` objects (the class `CommercialFish` is provided for you with this assignment). You will then provide the user with a menu of options of operations to take on the data, such as modifying data, summing data, finding the species that resulted in the highest dollar value, etc.

Input: Your program will read a file called `Commercial Fish 2014.txt` (provided in the class website).

You can see from the text file that there is data for ten species of fish in the file, ordered as name, weight, average weight over five years, dollar value. Here are the top eight lines of the file:

```
Alewife
1735
1645
488
Anchovies
23410
10371
1680
```

You can assume a file of data for ten fish organized in this way. You do not need to validate the data in the file. You may assume complete records of the correct data types. You may not assume that this data or these species are in the file.

(Data is from the following source:
https://www.st.nmfs.noaa.gov/Assets/commercial/fus/fus15/documents/02_Commercial2015.pdf)

Additional input will be user responses to prompts. You may assume that the user responds with the correct data type at each prompt.

Output: Your program will present a menu of options to the user and will obtain the user's choice. For each option, there are additional responses from the system. See the interface specifications for details.

Interface: Your program will present the following menu and prompt to the user:

```
Commercial Fisheries Data Menu
1. List all species data
2. Change weight
3. Show average weight difference
4. Show largest dollar value
5. Show total dollar value
6.  Quit
Your choice:
```

If the user chooses to list all species data, the program will display a summary of each species in the file. Each summary will be formatted as these two examples (note that the provided class will format this for you):

```
Name:                          Alewife
Pounds (1000's):               1,735
Five year average:             1,645
Cost (1000's): $               488


Name:                          Anchovies
Pounds (1000's):               23,410
Five year average:             10,371
Cost (1000's): $               1,680
```

If the user chooses to change weight or dollar value, they will be presented with a menu of species from the file and a prompt. The menu items should be numbered starting with one. (Remember that the species shown here are not necessarily the species in the file):

```
Select a species:
1. Alewife
2. Anchovies
3. Atka mackerel
4. Bluefish
5. Blue runner
6. Bonito
7. Butterfish
8. Catfish
9. Atlantic Cod
10. Pacific Cod
Which species? (1-10):
```

If the user has chosen to change weight, the current weight should be shown, followed by a prompt for the new value, followed by a summary for the species that was changed, as in this examples:

```
Current weight: 1735
Please enter the new weight: 1800
Name:                          Alewife
Pounds (1000's):               1,800
Five year average:             1,645
Cost (1000's): $               488
```

If the user chooses option 3, the menu of species and prompt should be shown, followed by the calculated result for that species, as in this example:

```
Difference between five-year average and 2014 weight: 595
```

If the user chooses options 4-5, the result should be calculated from the data in the array (originally from the file) and displayed as in these examples:

```
The species with the largest dollar value is:
Name:                          Pacific Cod
Pounds (1000's):               717,548
Five year average:             664,353
Cost (1000's): $               153,724

The sum of the dollar values of all species is: $201,172,000.00
```

If the user enters an invalid option at any prompt, the prompt should be displayed again.

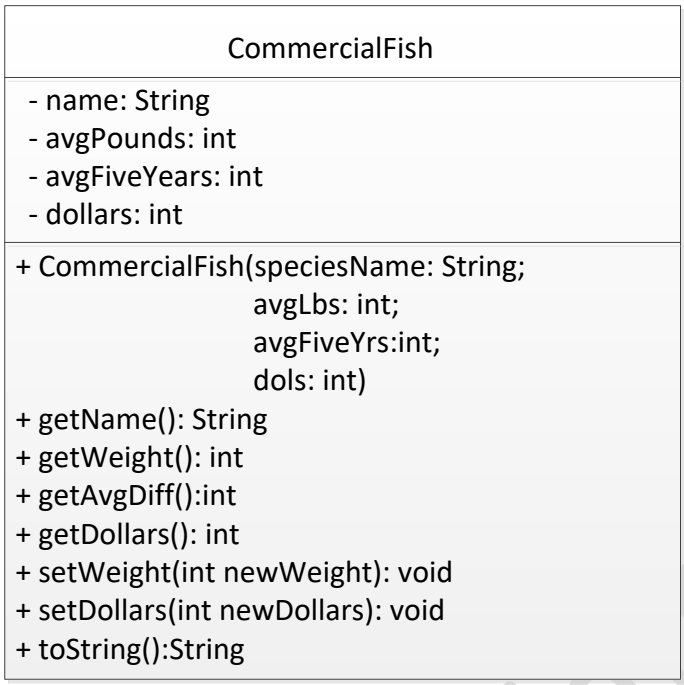
If the user chooses to quit the program, the program should end (with no output).

Design Specifications:

Use only techniques that are covered in this course. Use console input and output. Do not, under any circumstances, use `System.exit`. Design your methods to have one `return` statement. Use `break` only in `switch` statements. Do not submit code that uses these constructs inappropriately.

`CommercialFish.java`

This is a class diagram of the `CommercialFish` class. You must use this class in your program to store the data for each species read from the file. This class is provided for you as a `.class` file in the class website. You should not implement it yourself. When I test your program, I will test it with the `.class` file that I provided. (Reminder: a `.class` file is a compiled `.java` file. You won't be able to view it with a text editor.) This part of this assignment is often tricky for students. Please start early, watch the videos on adding a class file to your project, and plan to spend some time getting this to work.



`CommercialFishClient.java`

This is the program that you will write. Follow these specifications:

`readFile method`

This program will start by invoking a method that will read the contents of a file called `Commercial Fish 2014.txt` (provided in the class website) into an array of ten `CommercialFish` objects. The method will return the array of objects. (Hint: Read four lines of data from the file, instantiate a `CommercialFish` object using the values you read from the file, add the object to your array – repeat until you reach the end of the file.)

`displayMenu method`

The program will then invoke a method that will offer the user the menu of choices, shown above in the interface section. The user should be prompted to select an option until they choose a valid option. The method will return the user's choice.

`changeWeight, showDiff, showAll, showLargeDollar, showTotalDollar methods`

The program will then invoke the method that handles the chosen menu option. Thus, there will be one method for each menu choice (except quit). The program must pass the array of `CommercialFish` objects as a parameter to each of the methods. The methods will display results directly to the console.

`pickSpecies method`

Because menu options 2 and 3 operate on one `CommercialFish` object, you should have a method called `pickSpecies` which takes the array of `CommercialFish` as a parameter, allows the user to select a species, and returns the selected object.

Hints

Begin by making sure you can use the `CommercialFish` class. Watch the videos in the class website on adding a class file to your Eclipse project, or, if you are running Java programs from the console or

using Notepad++, you should be able to simply drop the class file in the same folder as your program. You can test that it is working by hard-coding some `CommercialFish` objects. Try all of the methods. They match the class diagram. Make sure you understand how they work and how to use them.

Next, create an array of `CommercialFish` objects. Write the algorithm to print all of the objects in the array first, so that you have some output you can look at. You can use the `CommercialFish toString` method to display the information shown for each species.

Next, put the program structure in. Write the method to read the data from a file. Even if you're still hard-coding the data in at this point, invoke the method and return the array. Write the method to show the menu, get a valid choice from the user, and return the choice. Put the functionality to display all species into a method. Pass the array in to that method.

Once you have that structure in place, start fleshing it out by writing all of the other methods and testing them, one at a time.

It is most important that you create an array of objects, so get that working first.

Clarifications: Include in your submission everything you've clarified with the professor.

Assumptions: Include in your submission anything you assumed but didn't clarify. Include a justification.

Testing

Test case 1

Purpose

Test all menu options with correct input, testing fish at the beginning and end of the array.

Input

A file of ten records of commercial fishing data such as:

```
Alewife
1735
1645
488
Anchovies
23410
10371
1680
Atka mackerel
69503
96543
22494
Bluefish
5182
5538
3106
Blue runner
301
306
268
```

```
Bonito
152
140
182
Butterfish
7292
3319
4754
Catfish
10000
9405
5118
Atlantic Cod
5170
11196
9358
Pacific Cod
717548
664353
153724
```

User input for each menu item, for selecting fish at the beginning and end of the array, and for new values.

Expected output

```
Commercial Fisheries Data Menu
1. List all species data
2. Change weight
3. Show average weight difference
4. Show largest dollar value
5. Show total dollar value
6.  Quit
Your choice: 1
Name:                       Alewife
Pounds (1000's):            1,735
Five year average:          1,645
Cost (1000's): $            488


Name:                       Anchovies
Pounds (1000's):            23,410
Five year average:          10,371
Cost (1000's): $            1,680


Name:                       Atka mackerel
Pounds (1000's):            69,503
Five year average:          96,543
Cost (1000's): $            22,494


Name:                       Bluefish
Pounds (1000's):            5,182
Five year average:          5,538
Cost (1000's): $            3,106
```

```
Name:                          Blue runner
Pounds (1000's):               301
Five year average:             306
Cost (1000's): $               268


Name:                          Bonito
Pounds (1000's):               152
Five year average:             140
Cost (1000's): $               182


Name:                          Butterfish
Pounds (1000's):               7,292
Five year average:             3,319
Cost (1000's): $               4,754


Name:                          Catfish
Pounds (1000's):               10,000
Five year average:             9,405
Cost (1000's): $               5,118


Name:                          Atlantic Cod
Pounds (1000's):               5,170
Five year average:             11,196
Cost (1000's): $               9,358


Name:                          Pacific Cod
Pounds (1000's):               717,548
Five year average:             664,353
Cost (1000's): $               153,724


Commercial Fisheries Data Menu
1. List all species data
2. Change weight
3. Show average weight difference
4. Show largest dollar value
5. Show total dollar value
6.  Quit
Your choice: 2

Select a species:
1. Alewife
2. Anchovies
3. Atka mackerel
4. Bluefish
5. Blue runner
6. Bonito
7. Butterfish
8. Catfish
9. Atlantic Cod
10. Pacific Cod
Which species? (1-10): 1
```

```
Current weight: 1735
Please enter the new weight: 1700
Name:                          Alewife
Pounds (1000's):               1,700
Five year average:             1,645
Cost (1000's): $               488


Commercial Fisheries Data Menu
1. List all species data
2. Change weight
3. Show average weight difference
4. Show largest dollar value
5. Show total dollar value
6.  Quit
Your choice: 3

Select a species:
1. Alewife
2. Anchovies
3. Atka mackerel
4. Bluefish
5. Blue runner
6. Bonito
7. Butterfish
8. Catfish
9. Atlantic Cod
10. Pacific Cod
Which species? (1-10): 5

Difference between five-year average and 2014 weight: -5

Commercial Fisheries Data Menu
1. List all species data
2. Change weight
3. Show average weight difference
4. Show largest dollar value
5. Show total dollar value
6.  Quit
Your choice: 4
The species with the largest dollar value is:
Name:                          Pacific Cod
Pounds (1000's):               717,548
Five year average:             664,353
Cost (1000's): $               153,700

Commercial Fisheries Data Menu
1. List all species data
2. Change weight
3. Show average weight difference
4. Show largest dollar value
5. Show total dollar value
6.  Quit
Your choice: 5
The sum of the dollar values of all species is: $201,148,000.00

Commercial Fisheries Data Menu
1. List all species data
```

```
2. Change weight
3. Show average weight difference
4. Show largest dollar value
5. Show total dollar value
6.  Quit
Your choice: 6
```

Your program running

Paste your program executing in the testing document here. Show a complete execution.

Conclusions

Describe whether your program's output matches expectations.

Test case 2

Purpose

Test invalid menu input for both menus.

Input

A file of ten records of commercial fishing data such as:

```
Alewife
1735
1645
488
Anchovies
23410
10371
1680
Atka mackerel
69503
96543
22494
Bluefish
5182
5538
3106
Blue runner
301
306
268
Bonito
152
140
182
Butterfish
7292
3319
4754
Catfish
10000
9405
5118
Atlantic Cod
5170
```

```
11196
9358
Pacific Cod
717548
664353
153724
```

User input for each menu item.

Expected output

```
Commercial Fisheries Data Menu
1. List all species data
2. Change weight
3. Show average weight difference
4. Show largest dollar value
5. Show total dollar value
6.  Quit
Your choice: 7
Your choice: 0
Your choice: 3

Select a species:
1. Alewife
2. Anchovies
3. Atka mackerel
4. Bluefish
5. Blue runner
6. Bonito
7. Butterfish
8. Catfish
9. Atlantic Cod
10. Pacific Cod
Which species? (1-10): 0
Which species? (1-10): 11
Which species? (1-10): 1

Difference between five-year average and 2014 weight: 90

Commercial Fisheries Data Menu
1. List all species data
2. Change weight
3. Show average weight difference
4. Show largest dollar value
5. Show total dollar value
6.  Quit
Your choice: 6
```

Your program running

Paste your program executing in the testing document here. Show a complete execution.

Conclusions

Describe whether your program's output matches expectations.

Turn in

1. Your java class (the .java file only). Make sure that it meets the requirements of the rubric.

2. The text files you used to test your program and the `CommercialFish` class file.

3. A word-processed document containing the following:

    Section 1: Clarifications and assumptions

    If you clarified specifications or made assumptions that were not clarified with the professor prior to submission, include them here with justifications.

    Section 2: Testing

    Copy / paste your program running on all test cases and include any additional test cases you used to thoroughly test your program.

Zip the java file, text file(s), class file, and the word-processed document into a folder named JavaProgram3_your initials, for example, JavaProgram3_msb.

Upload the zipped folder to Blackboard.

Rubric

An exceptional-quality assignment will meet the following standards:

- Meeting functional and design specifications, 70 points
  The Java program works and meets all of the specifications, with no additional unspecified functionality. The programmer has used programming techniques from the first six lessons only. If the program misses specifications or does not function correctly, errors are acknowledged with a thorough and reflective analysis in the testing section (points will be removed for missed specifications).
- Communicating with identifiers and white space, 10 points
  The program makes appropriate use of variables. Variables and constants are named according to convention and are named for understandability and purpose. White space, both vertical and horizontal, is correctly used for readability and meets programming conventions.
- Communicating through documentation, 10 points
  The Java program contains comments including the programmer's name and date. There are block comments (as many as necessary) for each distinct block of code which accurately describe what the block is accomplishing by relating the code to the problem being solved. Javadoc is included and meets the javadoc standards.
- Assumptions and Testing, 10 points
  Testing is thorough. If there are errors, they are described in the testing section. If there are questions, they are answered thoughtfully in the testing section. All assumptions are made explicit. The assignment will be reduced by a minimum of 10 points if the programmer claims a non-working program meets specifications.