CSC240 – SPRING 2013
ASSIGNMENT #2

The Power of OOPS

This assignment involves using the ideas from the program that we discussed in class that used inheritance, polymorphism and abstract classes. In particular, that project used an abstract class, Element. In this assignment, you will create two subclasses of Element. In addition, you will modify the ElementSet class from the program that we discussed to include two new methods.

The Two Subclasses of Element

In this project you will create two subclasses of the abstract class, Element. The Element class (used in the project, Abstract) has the following five methods:

readIn
display
equals
clone
getClassName

The first four of these methods must be implemented in the subclasses. The getClassName method is implemented once and for all in the Element class.

Here is a description of the two subclasses of Element that you will be implementing. The first of these subclasses is the FrozenFood class from assignment #1. The FrozenFood class will just be modified a little bit relative to what you did for assignment #1. In particular, you already have the readIn and display methods. So, you will have to make the following changes:

- FrozenFood must be a subclass of Element
- You must implement the equals method (two FrozenFoods are equal if they have the same name)
- You must implement the clone method.

Here again is a description of the fields in the FrozenFood class:

Field	Type	Description
name	String	Name of the frozen food product (e.g., "VEGGIE BURGER")
manufacturer	String	Name of the company that makes this frozen food product ("FRESH GARDEN")
nutrients	String	A list of the top nutrients that this frozen food provides (e.g., "PROTEIN, SODIUM; VITAMIN A")

The second of the two Element subclasses is the CannedFood class. Here is a description of the fields in the CannedFood class:

Field	Type	Description
name	String	Name of the CannedFood (e.g., "CHILI BEANS")
manufacturer	String	Name of the company that makes this frozen food product ("FANTASTIC BEANS")
ingredients	String	A list of the top ingredients in this canned food product (e.g., "KIDNEY BEANS; PINTO BEANS; GARABANZO BEANS")

For the CannedFood class equals method, two CannedFood objects are equal if they have the same name.

Modifying the ElementSet class

You will be adding two methods to the ElementSet class:

```
public boolean displayAnObject(Element anObject) …
public boolean editAnObject(Element editedObject) …
```

Here are descriptions for these new ElementSet methods:

The displayAnObject method will search through the calling ElementSet for the parameter object, anObject. Now, under most circumstances (and all circumstances for your application), anObject just has the required data to

do a search. In other words, for a FrozenFood or a CannedFood object, anObject's name field is set to a String, but the other fields are just empty Strings. If an equivalent object is found, the display() method is applied to that object polymorphically. If the search is successful, displayAnObject returns true. If the search fails, displayAnObject returns false. The method displayAnObject does not change the value of currentIndex.

The editAnObject method will try to find an object that is equal to the parameter, editedObject. The notion of equality will be the same as that which was used for the displayAnObject method. That is, this method will look for an object that belongs to the same class as the object that editedObject belongs to and is equal to it (as determined by the application of the equals method). If a matching object is found, it is replaced by editedObject and editAnObject returns true. If a matching object is not found, the ElementSet remains unchanged and the editAnObject method returns false.

Creating an Application that Uses these Classes

You will develop an application that uses the FrozenFood, CannedFood and ElementSet classes. The application will present the user with the following menu over and over again, until the user indicates that he/she wants to quit:

Wholly Nutritious Food Data Menu
1 – Add a Frozen Food Product
2 – Add a Canned Food Product
3 – Display names of all Frozen Food Products
4 – Display names of all Canned Food Products
5 – Display data for a Frozen Food Product
6 – Display data for a Canned Food Product
7 – Edit data for a Frozen Food Product
8 – Edit data for a Canned Food Product
9 – Quit

Here are brief descriptions for these menu options:

1 – Add a Frozen Food Product

Prompt the user for the FrozenFood data. If that FrozenFood object is already in the ElementSet (as indicated by the value returned by the add method), give the user appropriate feedback. If the ElementSet is full, give the user appropriate feedback. Otherwise, add the FrozenFood object to the ElementSet and give the user appropriate feedback.

2 – Add a Canned Food Product

Same idea as adding a Frozen Food product, except we are adding a Canned Food object to the ElementSet. In all cases, let the user know what transpired (success or failure).

3 – Display Names of All Frozen Food Products

An important issue here is that ONLY the names of the Frozen Food products will be displayed, not all of the data for each of the Frozen Food objects in the ElementSet. The names of each of the Frozen Food products in the ElementSet will be displayed. However, if there are no Frozen Food objects in the ElementSet, the user is given appropriate feedback.

4 – Display Names of all Canned Food Products

An important issue here is that ONLY the names of the Canned Food products will be displayed, not all of the data for each of the Canned Food objects in the ElementSet. The names of each of the Canned Food products in the ElementSet will be displayed. However, if there are no Canned Food objects in the ElementSet, the user is given appropriate feedback.

5 – Display Data for a Frozen Food Product

Prompt the user for and get the name of a Frozen Food product. Stuff that name into a FrozenFood object and pass that FrozenFood object as the parameter in a call to the displayAnObject method. If an equivalent FrozenFood object is found, the data for that FrozenFood will be displayed and the displayAnObject method will return true. If that FrozenFood object is not found, the displayAnObject method will return false and the calling method in the application will be responsible for giving the user appropriate feedback.

6 – Display Data for a Canned Food Product

Same idea as displaying a Frozen Food product, but this time we will be displaying the data for a CannedFood object using a call to the displayAnObject method. If the search fails, the application will be responsible for giving the user appropriate feedback.

7 – Edit data for a Frozen Food Product

The user is asked to enter the name of the FrozenFood product. The user is then prompted to enter the revised data for the FrozenFood product. Essentially, you can just use the FrozenFood readIn method in order to accomplish this reading in process. You don't need to ask the user which FrozenFood data the user wants to modify. I made this decision in order to keep things simple. So, basically you are creating a FrozenFood product and getting all of the data for that product. If that product is in the ElementSet, it is replaced (using the editAnObject method) with the new data and the user gets appropriate feedback. If that product is not in the ElementSet, the user gets appropriate feedback.

8 – Edit data for a Canned Food Product

Same ideas as editing the data for a FrozenFood product. The user is asked to enter the name of the CannedFood product. The user is then prompted to enter the revised data for the CannedFood product. Essentially, you can just use the FrozenFood readIn method in order to accomplish this reading in process. You don't need to ask the user which CannedFood data the he/she wants to modify. I made this decision in order to keep things simple. So, basically you are creating a CannedFood product and getting all of the data for that product. If that product is in the ElementSet, it is replaced (using the editAnObject method) with the new data and the user gets appropriate feedback. If that product is not in the ElementSet, the user gets appropriate feedback.

9 – Quit

If the user chooses this menu choice, the user must confirm this choice (as in assignment #1). If the user confirms that he/she wants to quit, the program terminates. Otherwise, the user is taken back to the main menu.

grading

Your program will be graded using a grading sheet that will be posted on D2L. Your program should satisfy the items in the grading checklist as much as possible. (Hopefully, 100%.) In addition, your program will be graded on:

- quality of the documentation
- clarity of your code
- the use of appropriate conventions of style
- correct use of classes and objects
- timely submission of your work
- a filled in, submitted project plan form