CSCI 3400
Fall 2023
Lab 03: The Observer Pattern
Assigned date: 09/14/2023
Due date: 09/25/2023(11:59 PM, EST)
Total points: 70 points

Learning Goal: In this lab, we will implement the observer patterns for a specific use case.

Given Source Code:

Package: `csci3400lab03`
Classes:

i) `MyFileReader`
`readFile(String fileName)`→ given a filename, it reads the data and returns the information as a list of strings.

ii) `DemoObservePattern`
`public static void main(String[] args)` → creates necessary objects and executes the instruction sets.

iii) `ObserveAndObservable` → defines the Observer interface and the Observable class (the same interface and class you have seen in the last class).

iv) `Weather` → contains some basic properties and methods to hold the weather information of a particular town/city.

Data File: weatherDatabase.txt

Note:
You are not allowed to alter the datatypes of the already defined classes/interface.
To record the outputs after you complete each task, create and a file named "comment.txt".

Tasks:

A. **[10 points]** Make proper adjustments so that `Weather` becomes Observable (Subject).

B. **[10 points]** Now, create an Observer class (name it as `WeatherObserver`) that will react (meaning, will handle) to any changes to the Weather fields.

C. **[10 points]** Uncomment and complete the following two lines in the `main(…) method` and record the output (meaning, in a separate file called "comment.txt", copy and paste the output of the execution of the main method. Then add one line to summarize the outcome.)

```
//Observer<Weather> weaObs =  ...;
//currentWeather.subscribe(weaObs);
```

Note: Once you are done recording the output, comment them out again.

D. **[10 points]** Now, create another type of Observer class (name it as `FieldObserver`) that will only react (meaning, will handle) to any changes to one particular field of the Weather class.

E. **[10 points]** Uncomment the following two lines in the **`main(…)`** method and record the output (meaning, in the "comment.txt" file, copy and paste the output of the execution of the main method. Then summarize the outcome in your own words.)

```
//FieldObserver tempObserver = new FieldObserver("Temperature-
observer", "temperature"); //assume it only reports the changes
in the temperature
//currentWeather.subscribe(tempObserver);
```

Note: Once you are done recording the output, comment them out again.

F. **[10 points]** Now, create a **subclass** of the **FieldObserver** class (name it as `FieldObserverX`) that will only react (i.e. handle) if a particular field is equal to a threshold/fixed value. For example, if the specified field is precipitation and the threshold is set to "light-rain", then the observer will report only when the weather notifies the precipitation field as "light-rain".

G. **[10 points]** Uncomment and complete the following two lines in the **`main(…)`** method and record the output (meaning, in the "comment.txt" file, copy and paste the output of the execution. Then summarize the outcome in your own words.)

```
//FieldObserverX precipObserverX =  new
FieldObserverX("Precipitation-Observer", "precipitation",
"isolated-thunderstorms");
//currentWeather.subscribe(precipObserverX);
```

Note: Once you are done recording the output, comment them out again.

Submission:

- Add your name after your instructor's name and add citations (any resource you have used while writing code, for example, any website you have looked for).
- Submit the source code and the comment.txt file.
- Please do not forget to include your and your partner's name in each of the files (source code and comment.txt).

Reference:

None for today.