Observer design pattern

IObserver interface (IListener interface)

- Contains one or more **update methods**
  (one for every relevant change/action in the observable which the observer should be notified of)

Observer classes (Event listener classes)

Observers are **updated when changes/actions happen** and can respond to those events
There can be one or many of these classes, and one or many instances of each

- Each observer class must **implement the IObserver interface**
- Therefore, each observer class must **define all of the update methods**

Observable class (Event source class)

Events of some kind happen here (**data is changed** or **actions are performed**)
The observable **updates all observers** whenever changes/actions happen

- Must contain a **collection of observers**
  - This should be a collection of IObserver **interface references**
  - The collection will often be a list, like **ArrayList**<IObserver>

- Must contain public **addObserver**(IObserver o) and **removeObserver**(IObserver o) methods
  - These should add/remove an observer to/from the collection

- Must **update all observers** when changes/actions happen
  - **For each observer** in the collection of observers:
    - Call the correct **update method** on the observer
  - This will typically be done in a **setter method** or other **action method**