**Exercise 12: Francisco Núñez Gómez**

**Please research from any resource and briefly explain the origin of Software design patterns.**

Patterns originated as an architectural concept by Christopher Alexander (1977/79). In 1987, Kent Beck and Ward Cunningham began experimenting with the idea of applying patterns to programming and presented their results at the OOPSLA conference that year. In the following years, Beck, Cunningham and others followed up on this work.

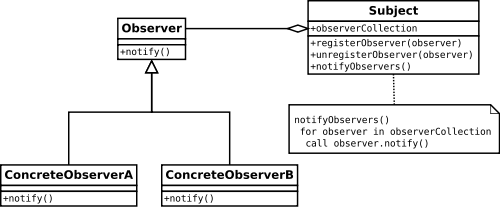
**Please explain in your own words why we should use Design Patterns in our projects?**

To have a standard on how to work, if we don’t have standards and patterns everyone would work as they think is the better way and no one could understand each other’s code.

**Please complete the below questions for each of the below list of patterns.**

**Observer Pattern-------------------------------------------------------------------**

Description: The observer pattern is a software design pattern in which an object, called the subject, maintains a list of its dependents, called observers, and notifies them automatically of any state changes, usually by calling one of their methods.



Use: It is mainly used to implement distributed event handling systems. The Observer pattern is also a key part in the familiar model–view–controller (MVC) architectural pattern. The observer pattern is implemented in numerous programming libraries and systems, including almost all GUI toolkits.

Example



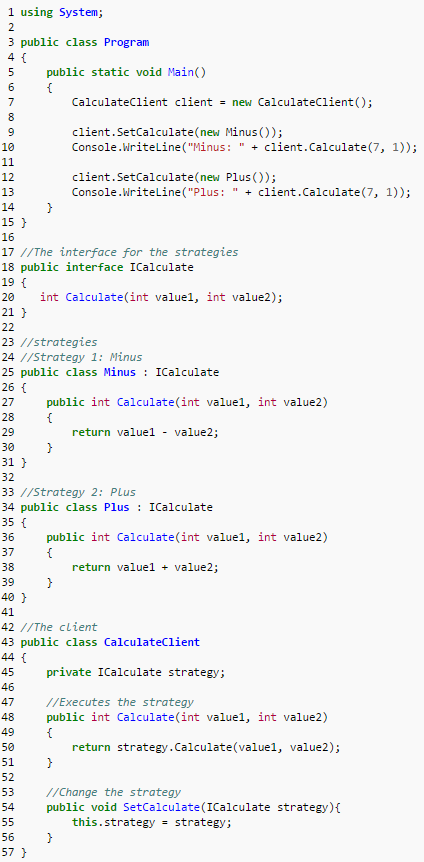
**Strategy Pattern---------------------------------------------------------------**

Description: is a software design pattern that enables an algorithm's behavior to be selected at runtime. The strategy pattern defines a family of algorithms, encapsulates each algorithm, and makes the algorithms interchangeable within that family.



Use: The validation strategies, encapsulated separately from the validating object, may be used by other validating objects in different areas of the system (or even different systems) without code duplication.

Example



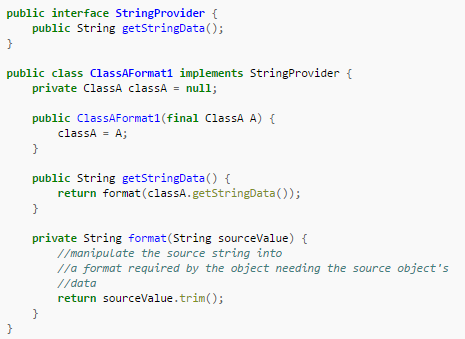
**Adapter Pattern--------------------------------------------------------------**

Description: allows the interface of an existing class to be used as another interface.



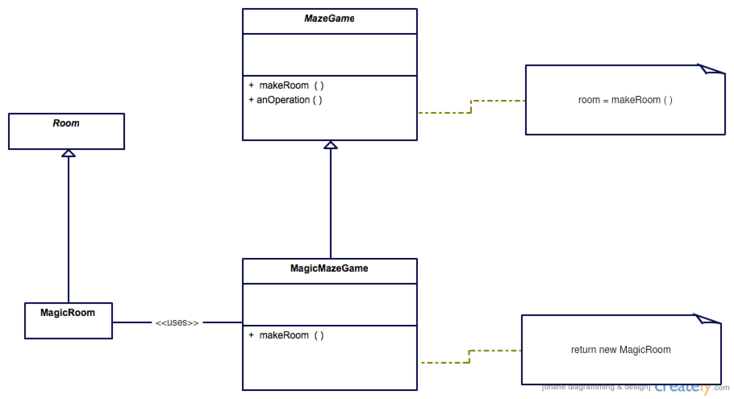
Use: It is often used to make existing classes work with others without modifying their source code.

Example



**Factory Method Pattern----------------------------------------------------**

Description: creational pattern that uses factory methods to deal with the problem of creating objects without having to specify the exact class of the object that will be created.



Use: This is done by creating objects by calling a factory method—either specified in an interface and implemented by child classes, or implemented in a base class and optionally overridden by derived classes—rather than by calling a constructor.

Example

