Question 3:
Explain how the composite and template method patterns are used in the Junit framework.

Both the composite pattern and template method pattern are made use of within testing for the Junit framework. The composite method pattern lets us abstract classes into a tree structure where a component class specifies an interface for compositing children and leaves. This lets the client call any object based off of a class in the tree through the interface specified by the component class. This is very useful for testing in JUnit because we want to have our test manager (Client) be able to call different tests and test suites (composites) we write through a uniform interface. Within these compositing classes and leaves we can use the template method pattern to further enhance our testing logic. The template method pattern allows us to redefine methods in superclasses in our subclasses that will be used in algorithms within the superclass. In general this is useful because you can redefine parts of an algorithm within a superclass method through inheritance rather than with more complex methods like lambdas and functional interfaces. In JUnit testing this is very useful because it lets us write our general testing logic within a Testcase superclass or have various further levels of child test case classes that implement their own general testing logic, before finally writing specific test case classes as our leaf nodes that will only need to run very basic operations since the most of the logic for testing is handled in the superclasses testing methods calling our subclass implementation.