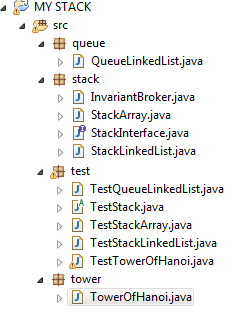
**Question 1**: The aim of this exercise is firstly, to write 2 different implementations of a Stack (as a linked list and as an array, for example). And secondly, to write tests for these 2 implementations.

Remark: A stack is a LIFO (Last In First Out) kind of storage. The stack should have a fixed size, determined at instantiation. Most part of the tests you’ll write should be the same for the 2 implementations. You can use interfaces, abstract classes, ...

**Answer**: The project MY STACK is structured as given described below:



1. **StackInterface:** Interface which consists all the common unimplemented methods for two different stacks and queue.
2. **StackArray:** Array implementation of the stack with all common unimplemented methods of interface StackInterface.
3. **StackLinkedList:** LinkedList implementation of the stack implementing the common unimplemented methods.
4. **VariantBroken:** Userdefined exception class which is customized to print the exception message provided as parameter.
5. Test package contains: Junit classes to test the corresponding implementation of stack implementation, Queue and tower of Hanoi.
6. TestStack: Abstract class contains all test cases available for both type of stack implementation and Queue implementation.
7. TestStackArray: Test class which extends all the method of TestStack and use some setup method to declare specific initialization for Array implementation of Stack.
8. TestStackLinkedList: Test class which extends all the method of TestStack and use setup method to declare specific initialization required for LinkedList implementation of Stack.

**Question 2**: Implement a Queue (FIFO-like storage) from 2 stacks. ¬ Can you use any of the 2 different implementations you developed? ¬ Write the tests. ¬ Can you use the same test suite as the one defined above? Why ?

**Answer**: Queue is implemented with the help of LinkedList implementation of stack and the corresponding class Is QueueLinkedList which implements the stackInterface.

QueueLinkedList has been tested with extending same TestStack abstract class and corresponding Test class is TestQueueLinkedList which is only for the neccessary initialization for QueueLinkedList.

As the implementation is performed with same linkedlist implementation of stack so, the same test class has been used with some different values for specific test methods like test\_pull and test\_head methods.

**Question 3:** Implement the “Tower of Hanoi” game using the store of integers you want.

* Write a test to verify that it works.
* Using the refactoring feature of Eclipse, replace the implementation of the store of integers you

Used first by the second one. Does it still work?

* Can you still use the test written first? Recursive algorithm:

**Answer:** Implementation of the “Tower of Hanoi” game is performed with the help of recursive algorithm and the corresponding class is **TowerOfHanoi**. The TowerOfHanoi is implemented using TestTowerOfHanoi on the basis of final result string.

It should work the stack mechanism as the logic of the game is same LIFO but the implementation wise, it didn’t work well to get the values on final tower and test against the final result of the game.

If the implementation is working fine, then the test written should also work fine. But as the implementation couldn’t performed with the stack. The separate tests are used to test the tower of Hanoi in Junit. As mentioned, the tests are included in TestTowerOfHanoi.