Explanation for Question #2

For question #2, we're asked to test 1 million operations to see which one is faster. Based on our testings, we found out that an Array Stack is faster than a LinkedList Stack. For this testing, we simply have 3 java files which are ArrayStack.java, LinkedListStack.java and StackPerformanceTest.java.

The StackPerformanceTest.java is the main class that performs and test each stack. We tested our operations in nanoseconds to give the best accurate results. Two variables we're using to calculate the nanoseconds are called "startingTimer" and "endingTimer". startingTimer will start timing the operation all the way until we initiate endingTimer. From there we will subtract (endingTime - startingTimer) to determine how long it took to fulfill the operations.

To test the speed of an Array Stack, we first initiated the "startingTimer" before the operations started. After initiating the timer we have for loops that will first push the operations into the array stack and then pop all the operations out of the array stack. Towards the end of the operation we ended the timer by initiating "endingTimer" which grabs the initial time. And finally we subtracted (endingTime - startingTimer) and got our results.

For LinkedList Stack we did the same process as the Array Stack List.

In our final testings, it shows that an Array Stack is faster than a LinkedList Stack from adding 1 million operations into the stack.

A few key notes about the remaining two classes:

ArrayStack.java Class

For the Array Stack, we will be using generics in the class to help push and pop out operations. This class also has methods to push, pop and determine if the stack is empty or not.

LinkedListStack.java Class

For the LinkedList stack we created a Node class along with other methods to help push, pop, peek, and check if the LinkedList stack is empty or not.