**Assignment 5**

1. Two stacks of positive integers are needed, both containing integers with values less than or equals to 1000. One stack stores only even numbers, and the other stores only odd numbers. The total number of elements in both stacks is never more than 200 at any given time, but the number of elements in each stack can vary significantly. Implement such integer stacks using *a single array* with a class named DoubleIntegerStack, and provide the following operations:
2. push(int item)
3. popOdd() and popEven() to remove an even number or odd number from one of the stacks
4. getEven() and getOdd() to get stack’s top item of a relevant stack; the stack is unchanged however.
5. getNumEvens(), getNumOdds() to return either the number of even numbers or number of odd numbers.

To test your implementation, write a loop to generate 150 integers randomly, and test the stack operations you coded. (You may read this Webpage for random number generation if needed: <http://www.cs.geneseo.edu/~baldwin/reference/random.html>) *Hint: If you store the numbers with evens and odds mixed, then most operations would require processing speed of O(N), which is not efficient. So how you store numbers does matter.*

1. Do the problem #47 (page 240), or #44 (page 233) if you use 2nd Edition of the text. Specifically, you may write a (static) method that simulates the auction process as described:

*public static void doAuction()*

In this method, you obtain user input from console: the name of person who is making a bid and the bidding amount (integer is fine). You then process the bidding according to the rules explained in the text. When information needs to be saved, you save it in stacks. As input proceeds the current status of the auction should be displayed (new bid, result, high bidder, high hid, and max bid). When user chooses to quit, the method displays the bid history (as shown in the book). You can use the example provided in the text to test the method.

***Note:*** It’s still expected that you use a table to describe all the tests you used. However use of JUnit is optional.