CS106/206 Assignment 4 Style Grading Rubrics

General

7 points are allocated to fairly mechanical rules on naming/comments/indentation - these should be easy to check off. Another 43 points are allocated to more creative practices, as explained below. Consult the formatting guide for details to check for under each category

Print student programs from Emacs, via “postscript print buffer” menu option.

Code formatting (**7 points total**)

1. Naming Conventions: **2 points**
    a. if any of the rules are violated
2. Whitespace: **1 point**
    a. inconsistent spacing (excessively) - - if just one place, point it out but don’t take off
3. Comments: **3 points**
    a. File header missing or malformatted
    b. Uncommented instance variables - no comment is okay if well-named
    c. Uncommented methods (getters and setters can have no comments, when appropriately named)
    d. Method comments that do not conform to javadoc style
    e. Uncommented complex blocks of code
    f. Unhelpful comments
4. Indentation: **1 point**
    a. inconsistent indentation (excessively) - if just one single line, point it out but don’t take off

Design principles (**43 points total**)

Assignment 4 (Stacks and Queues)

1. private Instance variables and getters **1 point**
    a. Any non-private instance variables, including missing modifier
2. Use `public static final` constants instead of integer/double/String literals - any literal that has reason to be changed later should be a constant **1 point**
    a. Default array capacity for part 2 should be a constant
3. Constructor must initialize all instance variables **1 point**
4. Part 1 **10 points** (if design is completely messed up, i.e. instead of using two ArrayStacks used many, or something else, take it all off. If efficiency is worse than should be, take at least half)
    a. implementation **5 points**
        i. Only two ArrayStack used and no other instance variables
        ii. toString overridden
        iii. enqueue is O(1)

      iv. dequeue is O(n)
   b. discussion **5 points**
      i. README analysis correct with justifications

5. Part 2 **10 points** (if design is completely messed up, i.e. not using an array circularly, or using many arrays instead of one, take it all off. If efficiency is worse than should be, take at least half)
   a. Implementation
      i. One circular array is used correctly with %
      ii. All methods should be O(1)
      iii. toString overridden
6. Part 3 **10 points** (If efficiency is not O(1) take it all off)
   a. Implementation **5 points**
      i. push and pop are O(1)
      ii. minElement is O(1)
      iii. Not too much additional storage is used to achieve O(1) for minElement, i.e. only one additional stack or queue or list
      iv. toString overridden
   b. discussion **5 points**
      i. README analysis correct with justifications
7. Main **10 points**
   a. Having reasonable tests in main that covers the functionality of all parts