CS206 Assignment 1 Style Grading Rubrics

10 points are allocated to fairly mechanical rules on naming/comments/indentation. Another 15 points are allocated to more creative practices, as explained below. Consult the formatting guide for details to check for under each category

Code formatting (**10 points total**)

1. Naming Conventions: **3 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: **5 points**
    a. File header missing or malformatted
    b. Uncommented instance variables
    c. Uncommented methods
    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 (**15 points total**)

1. private Instance variables and getters **2 points**
    a. Any non-private instance variables, including missing modifier
    b. Missing getters, even if not used
2. `public static final` constants instead of integer/double literals - any literal that has reason to be changed later should be a constant **2 points**
    a. Cases noted
        i. Using `"00000"` directly in code
        ii. Using `[0]`, `[1]`, `[2]...` directly in code after calling `split`
3. Constructor must initialize all instance variables **1 point**
    a. Check `Place` constructor
    b. It is acceptable for class `LookupZip` to not define a constructor, as long as they didn't define any instance variables for the class
4. Adhering to given method signatures and designs **3 point**
    a. Check signatures of `parseLine`, `readZipCodes` and `lookupZip`
    b. Each of the methods should also do what they are designed to do - any abuse/overcall/redundant use gets -1:
        i. `parseLine` takes one line of input and creates the corresponding `Place` and returns it
        ii. `readZipCodes` creates `Place[]`, calls `parseLine` in a loop (of exactly # of lines in file times) and populates `Place[]` with return value

of `parseLine` and returns `Place[]`

iii. `lookupZip` is called in a while loop in `main`, once per lookup/user input

5. Only one correctly-sized array of `Place` used and created only once **4 points**
    a. Any additional data structure -2
        i. This includes creating the array in a loop over and over again
    b. Array size should be exact (42,613 - read from first `int` in the file) and not too large -1
    c. Array size should not be hardcoded -1
6. Reasonable designs for `Place`, `LoopupZip` and no additional classes (besides `Main` of course) **3 points**
    a. `Place` has the required instance variables and no additional. Has all the necessary getters (don't take off again, there were points specifically for missing getters - see 1). Preferably has `toString` overridden (see 7)
    b. `LookupZip` doesn't have instance variables
    c. `LookupZip` holds the three methods `parseLine`, `readZipCodes` and `lookupZip`
7. Printouts should be generated by overriding `toString` in `Place`. However, I forgot to specify this in the assignment. So if they didn't, point it out to them but don't take off.
8. There is no need to instantiate `LookupZip`. The static methods can be called directly via "className.methodName". If they didn't, point it out but don't take off.