MP5: Code Smell and Static Analysis

1 Introduction

Writing good code is difficult, but fixing someone else's code can be even harder...

Suppose now you are contributing to the Jsoup project, and you are asked to revise/refactor the `src/main/java/org/jsoup/parser/TokenQueue.java` file so it can pass some new code review rules. Note that if you have skipped MP1, please follow MP1 first to clone the Jsoup version we use for this class via `git clone https://github.com/uiuc-cs427-f23/jsoup`; if you already have done MP1, you can work on the same Jsoup copy you had for this MP.

The new code review has the following strict rules (checking Google coding conventions from Google Java Style):

- Line length should not exceed 100 characters.
- Method length should not exceed 35 lines (`MethodLengthCheck`).
- Inappropriate import statements should not appear (`RedundantImportCheck`, `UnusedImportsCheck`, `AvoidStarImportCheck`).
- Magic numbers should not appear (`MagicNumberCheck`).
- One line should have only one statement (`OneStatementPerLineCheck`).
- Constant names should have the correct naming format (`ConstantNameCheck`).
- One method should have only one return statement (`ReturnCountCheck`).
- There should not be overly complicated boolean expressions.
- All the conditional statements, try/catch statements, ... should be wrapped by braces, even if their body contains a single statement (`NeedBracesCheck`).
- There should be no empty block (`EmptyBlockCheck`, `EmptyCatchBlockCheck`).
- You should avoid multiple variable declarations in one statement (`MultipleVariableDeclarationsCheck`).

Please refactor the Java file without changing the original semantics of the program. **Please ensure that all Jsoup tests still pass after the changes**. You should do the refactoring based on the results of the CheckStyle report.

2 Instructions

- The grader checks about the following smells that are explained above: `RedundantImportCheck`, `MagicNumberCheck`, `OperatorWrapCheck`, `MultipleStringLiteralsCheck`, `UnusedImportsCheck`, `ConstantNameCheck`, `AvoidStarImportCheck`, `ReturnCountCheck`, `EmptyBlockCheck`, `MultipleVariableDeclarationsCheck`, `EmptyCatchBlockCheck`, `OneStatementPerLineCheck`, `MethodLengthCheck`, `NeedBracesCheck`
- You should use `mvn clean test checkstyle:checkstyle -Dcheckstyle.includes="**\\TokenQueue.java"` to generate a style report for the Java file (you can find the report under the `target` folder). Note that the checkstyle plugin has already been set up in the Jsoup project released for MP1. `mvn clean test` was included in the command to ensure that all Jsoup tests still pass after your changes.
- The style report is stored in the `target/site` directory in an html file.

3 Deliverables

You are expected to upload a zip file including only the `src/main/java/org/jsoup/parser/TokenQueue.java` file you completed (no folders please). The name of the zip file should be your NetID. Please make sure that the modified file now does not have any warnings and can pass the checkstyle check (as well as passing all tests via `mvn clean test`).

Warning: you may lose all your points if you violate the following rule:

- Please **DO NOT fork any assignment repo** from our GitHub organization, `uiuc-cs427-f23`, or share your solution online.

4 Grading rubric

The autograder will run checkstyle on your submitted file and your final score for this MP will be calculated as the remaining warnings. In total, this MP is also 5pt. You will lose 1pt for each remaining warning until you receive 0pt. Note that if your submitted file fails any test, you will directly receive 0pt.