Assignment 6 - Refactoring/Metrics

LATEST SUBMISSION: April 30th (later will be 0 - no late pass or 10% option)

Objectives

- Identify code smells and refactor to solve them.
- Refactor in small steps to improve code quality.

General

In this assignment, we will be using a tool called PMD through Gradle. PMD is a source code analyzer that finds common programming flaws like unused variables, empty catch blocks, unnecessary object creation, and so on. You can install it if you'd like, but it's not required.
You will find an xml file (which is a given ruleset), as well as a new build.gradle file from which you will need all the PMD related things.
During all the tasks below, you need to make sure that the code still works as expected by using your Unit Tests.
Work on the GitHub repo that you used for your assignments 2-4.
You will need a document in which you answer some questions and paste some screenshots.

Task 1 (5 points)

Create a new branch based on your Blackbox branch. Call this branch metrics1. Copy over the PMD related code from the given build.gradle file into your build.gradle file. Also copy the xml file into this branch (same directory as the build.gradle file).
Now, run a *gradle build*. In your report folder you should find a PMD report. Take a look at it. You will see that the thresholds, when PMD should start to report things are set very low. We have such a tiny program and I wanted something to show up. In reality, we would not want the thresholds to be this low.
In your document: Give me a write up in which you explain the metrics and information that PMD gives you and map this to what the XML ruleset file says. Take a screenshot of the report and include it in your document.

Task 2 (5 points)

Create a new branch based on your Dev branch (which should include all your hard work from assign 2-4). Call this branch metrics2. Do the same thing as in Task 1 to include PMD. Now run it.
In your document: Compare the two outputs for me in detail in an evidence based manner using the PMD results. Did all your work make any impact on the reported issues? Why, why not?
Take a screenshot of this report.

Task 3 (20 points)

Now, we want to get into refactoring and also keep track of our metrics while doing this. Keep working on branch metrics2.

Task 3.1: Make code more adaptable (10 points)

Think about how you could alter the code to make it more adaptable to different levelUp oportunities. Do not just think abou the values but make it more adaptable for which of the values actually change. You do not have to make sure it is 100% able to handle all kinda of levelUps but come up with an idea so that the levelUp strategy can easily be changed. I just want to see that you come up with a good idea to improve the extensibility of your code.
In your document: (5 points) Explain in your own words what changes you suggest? (5 points) Implement your suggestions

Task 3.2: Duplicate or similar code (10 points)

You might have some duplicate code in your assignment code (depending on changes you made), remove such duplicates.
Do you find any other duplicate code (depending on your previous changes, you might have more or less to find here).
In your document: Describe what you would need to change (5 points) and make the necessary changes in your code (5 points).

Task 4: Find Code Smells (5 points)

You will find what is considered to be a code smell in the lecture slides. Make sure to read the chapter titled "Code Smells" from Fowler!
Check the PMD rule sets online (search for the PMD documenation). Are there any rules you think would help you find code smells? If so, include them in your rule set into branch metrics2. The rules you choose do not necessarily have to be violated in your code and you do not have to fix them either.
In your document: Explain why you think these rules would help find smells.

Task 5: Final check (5 points)

Based on everything you learned this session and considering the Refactoring you are doing now, are there any parts in this code you would change? Do you still find any code smells that you think should be fixed? You can implement these but do not have to, just naming them is sufficient.
In your document: Explain in detail why refactoring could still improve the code or why it is already good as is. Be specific.

Submission

1. Include your link to your GitHub repo in the PDF and in your comment section when you submit.

2. Document as a pdf named <asurite>_metrics.pdf