Jon Doretti

SE333

Homework 2

Part 1:

For this assignment, I decided to use GnattProject as with DesignateJava the Apache-Ant-1.10.12 was appearing to be less than 50,000 lines. This has to be an error however I decided to pick a different project. One class that DesigniteJava and Understand differed on is the GPLogger class. They differ in calculation of the LOC. This could be because DesigniteJava's counts comments or blank space but Understand does not. I would say DesigniteJava has the more accurate reading as I would assume the higher one is more accurate. This has really perplexed me as I would assume on a metric like Lines of code both softwares would provide the same data. I wonder if this is a bug or code smell is the development of either software. For code smells withing the paintMe method there are long declarations like g.drawString(StringUtils.getTruncatedString(name`getNodeWidth() - getTextPaddingX()`fontMetrics)`x + getTextPaddingX()`y + getTextPaddingY() + fontMetrics.getHeight());. A lot of these calculations are repeated in the paintMe class; to be specific 3 times. The method calls seem like they could be calculated beforehand and added to variables however this would create the LOC to also increase making the method longer. Another possible solution is to create a method where you would pass all these variables and it would return the calculation to the user. This would lessen lines of code and decrease the amount of times this long method call is being put into the code. As it may be necessary creating a method for this would allow the long declaration to only appear once instead of 3 times.

```
doret@MacBook-Pro SE333 HW2 % java -jar DesigniteJava.jar -i /Users/doret/Documents/SE333-HW2/ganttproject-master -o  /Users/doret/Documents/SE333-HW2/output/
Searching classpath folders ...
Could not find any classpath folder.
Parsing the source code ...
Resolving symbols...
Computing metrics...
Detecting code smells...
Exporting analysis results...
wrapping up ...
--Analysis summary--
        Total LOC analyzed: 67794       Number of packages: 92
        Number of classes: 971  Number of methods: 9128
-Total architecture smell instances detected-
        Cyclic dependency: 118  God component: 6
        Ambiguous interface: 0  Feature concentration: 21
        Unstable dependency: 27 Scattered functionality: 0
        Dense structure: 1
-Total design smell instances detected-
        Imperative abstraction: 1       Multifaceted abstraction: 2
        Unnecessary abstraction: 8      Unutilized abstraction: 196
        Feature envy: 33        Deficient encapsulation: 82
        Unexploited encapsulation: 1    Broken modularization: 4
        Cyclically-dependent modularization: 57 Hub-like modularization: 1
        Insufficient modularization: 69 Broken hierarchy: 84
        Cyclic hierarchy: 3     Deep hierarchy: 0
        Missing hierarchy: 1    Multipath hierarchy: 2
        Rebellious hierarchy: 4 Wide hierarchy: 0
-Total implementation smell instances detected-
        Abstract function call from constructor: 3      Complex conditional: 79
        Complex method: 109     Empty catch clause: 19
        Long identifier: 39     Long method: 20
        Long parameter list: 130        Long statement: 849
        Magic number: 1756      Missing default: 45
----
Done.
```

Part 2 – A

Cyclomatic complexity:

CC – E-N+2 | 7-5+2 = 4

CC=4

CC = NoPN + 1 | 3 + 1 = 4

CC = 4

Basis set of test paths:

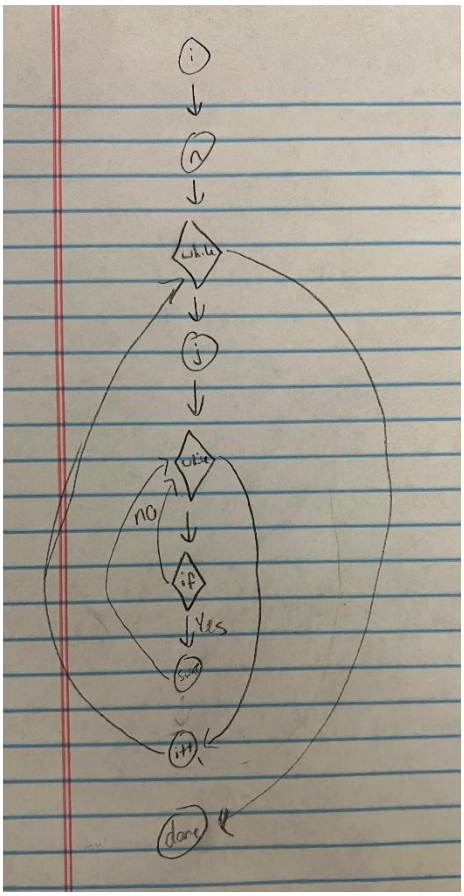
1 -> 2 -> 3 -> 5

1 -> 2- > 4 -> 5

1 -> 2 -> 4 -> 3 -> 5

1 -> 3 -> 5

Part 2 - B



CC = 11 – 9 + 2 = 4

CC = 4