

### Task 1:

#### Size:

1. Lines of Code: 2187
2. Largest Code File: EventsManager.java
3. Method used: lines of code with characters on them

#### Cohesion:

1. Calculating LCOM2:

P: Number of pairs of methods that do not share attributes

Q: Number or pairs that do share attributes

S: Number of pairs

IF  $P-Q < 0$ ,  $LCOM2 = 0$

ELSE  $LCOM2 = (P-Q)/S$

2. Highest Cohesion: Most of the classes have an LCOM of 0, which indicates that they have a high cohesion because they lack methods that stand alone.

The class with the lowest cohesion, and therefor the highest LCOM score, is

TaskListImpl, which means that many of the methods could stand outside the class on their own.

#### Complexity

1. 1.746
2. EventsManager with 2.5
3. I got rid of the if statement at the beginning of `getDay()` and `getDays()` within EventsManager -> Month

#### Package-Level Coupling

1. Afferent couplings are the number of classes outside the package that depend on the package, indicating how much that package is relied upon.

Efferent couplings are the number of classes from other packages which the classes within the packing depends on, indicating package dependencies.

2. main.java.memoranda.util: 57
3. main.java.memoranda.ui: 49

#### Worst Quality

TaskListImpl is the worst class because of both its lack of cohesion (0.679 score) along with its high average cyclomatic complexity (2.273) make it a badly designed class.

## Task 2:

### Before:

Metrics - main.java.memoranda					
Metric	Total	Mean	Std. Dev.	Maximum	Resource causing Maximum
▶ McCabe Cyclomatic Complexity (avg/max per method)		1.74	1.542	16	/SER316-Spring-2018/src/main/java/memoranda/
▶ Number of Parameters (avg/max per method)		0.675	1.004	8	/SER316-Spring-2018/src/main/java/memoranda/
▶ Nested Block Depth (avg/max per method)		0.997	0.945	8	/SER316-Spring-2018/src/main/java/memoranda/
Afferent Coupling	34				
Efferent Coupling	21				
Instability	0.382				
Abstractness	0.275				
Normalized Distance	0.343				
▶ Depth of Inheritance Tree (avg/max per type)		0.854	0.607	2	/SER316-Spring-2018/src/main/java/memoranda/
▶ Weighted methods per Class (avg/max per type)	583	14.22	16.093	71	/SER316-Spring-2018/src/main/java/memoranda/
▶ Number of Children (avg/max per type)	23	0.561	1.624	10	/SER316-Spring-2018/src/main/java/memoranda/
▶ Number of Overridden Methods (avg/max per type)	3	0.073	0.341	2	/SER316-Spring-2018/src/main/java/memoranda/
▶ Lack of Cohesion of Methods (avg/max per type)		0.093	0.211	0.679	/SER316-Spring-2018/src/main/java/memoranda/
▶ Number of Attributes (avg/max per type)	30	0.732	1.037	4	/SER316-Spring-2018/src/main/java/memoranda/
▶ Number of Static Attributes (avg/max per type)	46	1.122	2.549	12	/SER316-Spring-2018/src/main/java/memoranda/
▶ Number of Methods (avg/max per type)	274	6.683	7.687	37	/SER316-Spring-2018/src/main/java/memoranda/
▶ Number of Static Methods (avg/max per type)	61	1.488	3.768	17	/SER316-Spring-2018/src/main/java/memoranda/
▶ Specialization Index (avg/max per type)		0.05	0.308	2	/SER316-Spring-2018/src/main/java/memoranda/
▶ Number of Classes	41				
▶ Number of Interfaces	11				
▶ Total Lines of Code	2183				
▶ Method Lines of Code (avg/max per method)	1255	3.746	5.206	33	/SER316-Spring-2018/src/main/java/memoranda/

### After:

Metrics - main.java.memoranda					
Metric	Total	Mean	Std. Dev.	Maximum	Resource causing Maximum
▶ McCabe Cyclomatic Complexity (avg/max per method)		2	1.718	16	/SER316-Spring-2018/src/main/ja
▶ Number of Parameters (avg/max per method)		0.722	1.024	8	/SER316-Spring-2018/src/main/ja
▶ Nested Block Depth (avg/max per method)		1.347	0.857	8	/SER316-Spring-2018/src/main/ja
Afferent Coupling	30				
Efferent Coupling	19				
Instability	0.388				
Abstractness	0.121				
Normalized Distance	0.491				
▶ Depth of Inheritance Tree (avg/max per type)		1.029	0.514	2	/SER316-Spring-2018/src/main/ja
▶ Weighted methods per Class (avg/max per type)	496	14.588	17.458	71	/SER316-Spring-2018/src/main/ja
▶ Number of Children (avg/max per type)	16	0.471	1.77	10	/SER316-Spring-2018/src/main/ja
▶ Number of Overridden Methods (avg/max per type)	3	0.088	0.373	2	/SER316-Spring-2018/src/main/ja
▶ Lack of Cohesion of Methods (avg/max per type)		0.112	0.227	0.679	/SER316-Spring-2018/src/main/ja
▶ Number of Attributes (avg/max per type)	30	0.882	1.078	4	/SER316-Spring-2018/src/main/ja
▶ Number of Static Attributes (avg/max per type)	29	0.853	1.912	7	/SER316-Spring-2018/src/main/ja
▶ Number of Methods (avg/max per type)	187	5.5	7.504	37	/SER316-Spring-2018/src/main/ja
▶ Number of Static Methods (avg/max per type)	61	1.794	4.071	17	/SER316-Spring-2018/src/main/ja
▶ Specialization Index (avg/max per type)		0.06	0.338	2	/SER316-Spring-2018/src/main/ja
▶ Number of Classes	34				
▶ Number of Interfaces	4				
▶ Total Lines of Code	2079				
▶ Method Lines of Code (avg/max per method)	1255	5.06	5.474	33	/SER316-Spring-2018/src/main/ja

The Afferent and Efferent Coupling both went down, likely due to outside methods calling the interface rather than the implementation.

The Cyclomatic Complexity and LCOM both went up, likely due to the Interfaces not having any code in them.

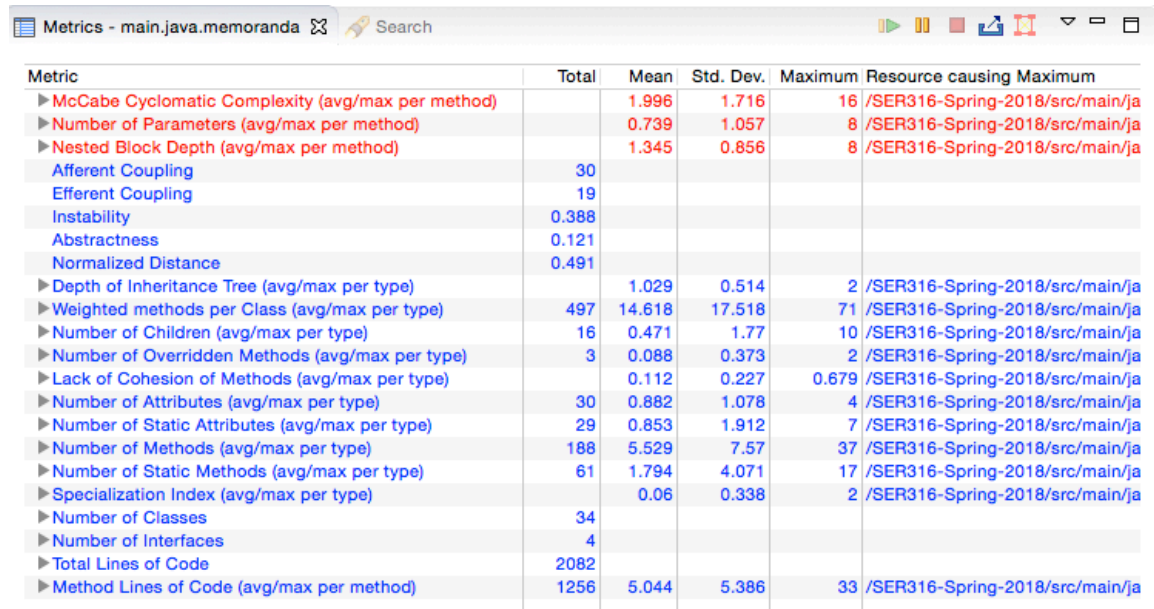
### Task 3:

1: Smell: Too long of a function. The function was over 20 lines long. It was fixed by moving part of the function of building a note into a private method. I moved the first part of adding data to an element into its own private method which the original method calls.

2. Didn't get to it.

3.

After



The screenshot shows a code editor window titled 'Metrics - main.java.memoranda'. Below the editor, a table displays various code metrics. The table has five columns: Metric, Total, Mean, Std. Dev., and Maximum. The 'Maximum' column also includes the resource causing the maximum value. The metrics are listed in descending order of their maximum values.

Metric	Total	Mean	Std. Dev.	Maximum	Resource causing Maximum
▶ McCabe Cyclomatic Complexity (avg/max per method)		1.996	1.716	16	/SER316-Spring-2018/src/main/ja
▶ Number of Parameters (avg/max per method)		0.739	1.057	8	/SER316-Spring-2018/src/main/ja
▶ Nested Block Depth (avg/max per method)		1.345	0.856	8	/SER316-Spring-2018/src/main/ja
Afferent Coupling	30				
Efferent Coupling	19				
Instability	0.388				
Abstractness	0.121				
Normalized Distance	0.491				
▶ Depth of Inheritance Tree (avg/max per type)		1.029	0.514	2	/SER316-Spring-2018/src/main/ja
▶ Weighted methods per Class (avg/max per type)	497	14.618	17.518	71	/SER316-Spring-2018/src/main/ja
▶ Number of Children (avg/max per type)	16	0.471	1.77	10	/SER316-Spring-2018/src/main/ja
▶ Number of Overridden Methods (avg/max per type)	3	0.088	0.373	2	/SER316-Spring-2018/src/main/ja
▶ Lack of Cohesion of Methods (avg/max per type)		0.112	0.227	0.679	/SER316-Spring-2018/src/main/ja
▶ Number of Attributes (avg/max per type)	30	0.882	1.078	4	/SER316-Spring-2018/src/main/ja
▶ Number of Static Attributes (avg/max per type)	29	0.853	1.912	7	/SER316-Spring-2018/src/main/ja
▶ Number of Methods (avg/max per type)	188	5.529	7.57	37	/SER316-Spring-2018/src/main/ja
▶ Number of Static Methods (avg/max per type)	61	1.794	4.071	17	/SER316-Spring-2018/src/main/ja
▶ Specialization Index (avg/max per type)		0.06	0.338	2	/SER316-Spring-2018/src/main/ja
▶ Number of Classes	34				
▶ Number of Interfaces	4				
▶ Total Lines of Code	2082				
▶ Method Lines of Code (avg/max per method)	1256	5.044	5.386	33	/SER316-Spring-2018/src/main/ja

4. It slightly increased the methods per class.

It increased the lines of code.

It slightly increased Number of Methods and Method Lines of Code.

I am not sure any of the changes in there were for the better or the worse. It was mostly a change in the amount of methods per class, so it seems to be a worse change.