Task 1: Evaluate some metrics (20 points)

Size

1. What is the Total Lines of Code (LOC) in the project?

22539

2. What is the largest single code file in the project and its Total LOC?

Htmleditor.java

3. Inspect CurrentNote.java - what method did the Metrics tool use to determine Total LOC? Describe the method.

Each statement is 1, that's all.

Cohesion

1.

Was not able to find it in the notes, googled and found this: **LCOM2 = 1 - sum(mA)/(m*a)**

Definitions used for LCOM2 and LCOM3	
m	number of procedures (methods) in class
a	number of variables (attributes) in class
mA	number of methods that access a variable (attribute)
sum(mA)	sum of mA over attributes of a class

Implementation details. m is equal to WMC. a contains all variables whether Shared or not. All accesses to a variable are counted.

2.

CalendarDate by far since it doesn't depend on any other resources. It is a stand alone java class.

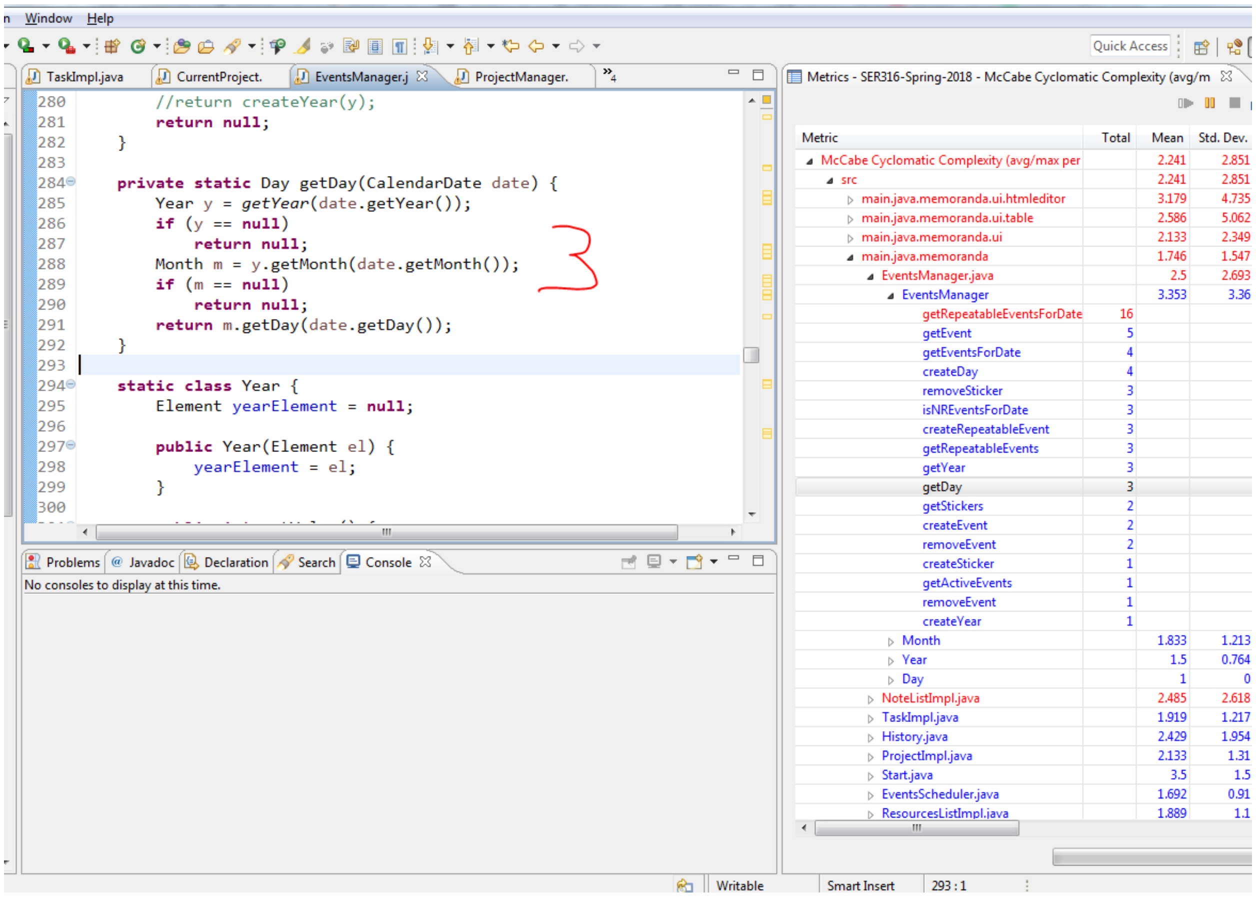
Complexity

1.

Mean=2.241, SD=2.851, Maximum of 42 of CC

2. ProjectPackaer.java has over 5.0+

3. In eventsmanager.java I changed CC in getDay function



```java
        //return createYear(y);
        return null;
    }

    private static Day getDay(CalendarDate date) {
        Year y = getYear(date.getYear());
        if (y == null)
            return null;
        Month m = y.getMonth(date.getMonth());
        if (m == null)
            return null;
        return m.getDay(date.getDay());
    }

    static class Year {
        Element yearElement = null;

        public Year(Element el) {
            yearElement = el;
        }
```

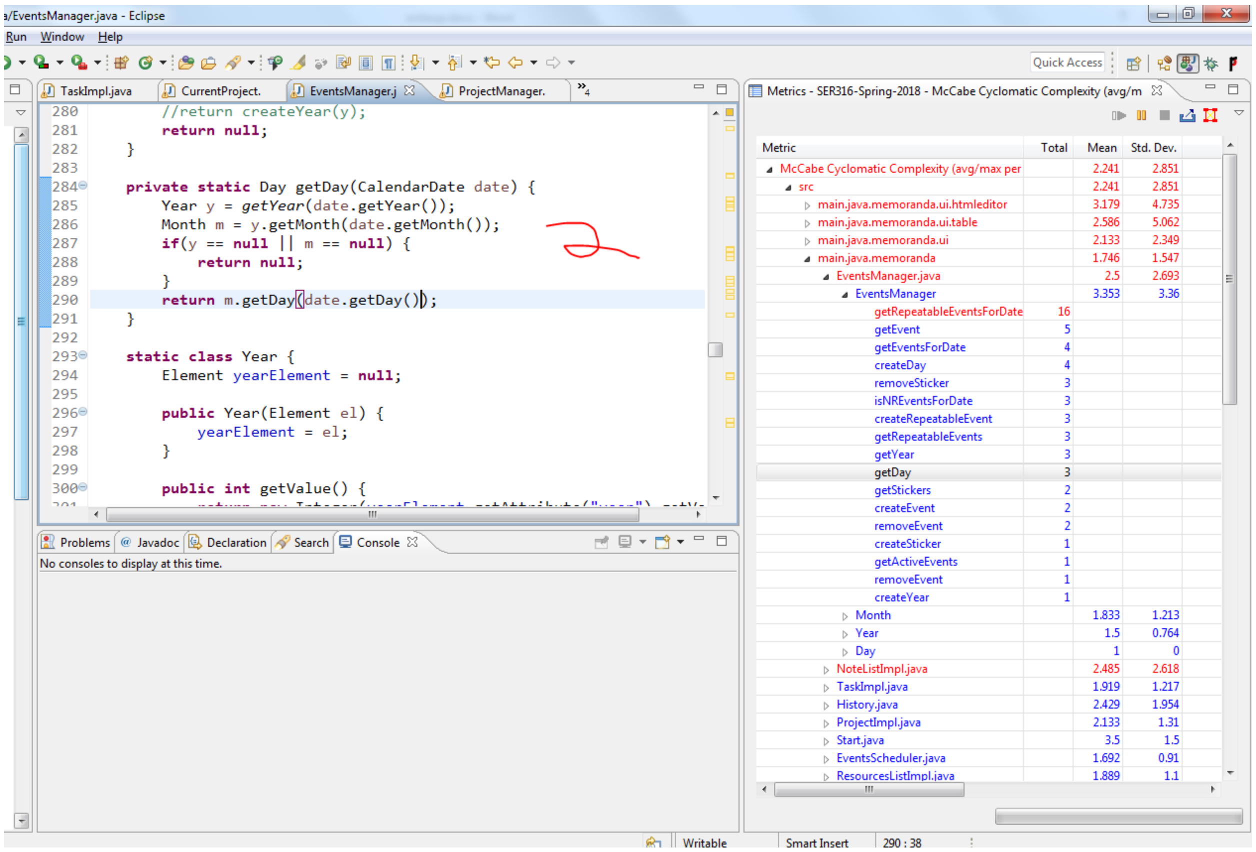
3

No consoles to display at this time.

Metrics - SER316-Spring-2018 - McCabe Cyclomatic Complexity (avg/m

Metric	Total	Mean	Std. Dev.
McCabe Cyclomatic Complexity (avg/max per		2.241	2.851
src		2.241	2.851
main.java.memoranda.ui.htmleditor		3.179	4.735
main.java.memoranda.ui.table		2.586	5.062
main.java.memoranda.ui		2.133	2.349
main.java.memoranda		1.746	1.547
EventsManager.java		2.5	2.693
EventsManager		3.353	3.36
getRepeatableEventsForDate	16		
getEvent	5		
getEventsForDate	4		
createDay	4		
removeSticker	3		
isNREventsForDate	3		
createRepeatableEvent	3		
getRepeatableEvents	3		
getYear	3		
getDay	3		
getStickers	2		
createEvent	2		
removeEvent	2		
createSticker	1		
getActiveEvents	1		
removeEvent	1		
createYear	1		
Month		1.833	1.213
Year		1.5	0.764
Day		1	0
NoteListImpl.java		2.485	2.618
TaskImpl.java		1.919	1.217
History.java		2.429	1.954
ProjectImpl.java		2.133	1.31
Start.java		3.5	1.5
EventsScheduler.java		1.692	0.91
ResourcesListImpl.java		1.889	1.1

Writable | Smart Insert | 293 : 1



```java
        //return createYear(y);
        return null;
    }

    private static Day getDay(CalendarDate date) {
        Year y = getYear(date.getYear());
        Month m = y.getMonth(date.getMonth());
        if(y == null || m == null) {
            return null;
        }
        return m.getDay(date.getDay());
    }

    static class Year {
        Element yearElement = null;

        public Year(Element el) {
            yearElement = el;
        }

        public int getValue() {
```

No consoles to display at this time.

Metrics - SER316-Spring-2018 - McCabe Cyclomatic Complexity (avg/m

Metric	Total	Mean	Std. Dev.
McCabe Cyclomatic Complexity (avg/max per		2.241	2.851
src		2.241	2.851
main.java.memoranda.ui.htmleditor		3.179	4.735
main.java.memoranda.ui.table		2.586	5.062
main.java.memoranda.ui		2.133	2.349
main.java.memoranda		1.746	1.547
EventsManager.java		2.5	2.693
EventsManager		3.353	3.36
getRepeatableEventsForDate	16		
getEvent	5		
getEventsForDate	4		
createDay	4		
removeSticker	3		
isNREventsForDate	3		
createRepeatableEvent	3		
getRepeatableEvents	3		
getYear	3		
getDay	3		
getStickers	2		
createEvent	2		
removeEvent	2		
createSticker	1		
getActiveEvents	1		
removeEvent	1		
createYear	1		
Month		1.833	1.213
Year		1.5	0.764
Day		1	0
NoteListImpl.java		2.485	2.618
TaskImpl.java		1.919	1.217
History.java		2.429	1.954
ProjectImpl.java		2.133	1.31
Start.java		3.5	1.5
EventsScheduler.java		1.692	0.91
ResourcesListImpl.java		1.889	1.1

Package Level coupling

1.

Afferent= the number of classes in other packages that depend upon classes within that same package, however

Efferent=number of classes in other packages that the classes in the package depende upon is a idnciator of the packages depeendceies.

In example is 2 classes within a single class that has no function to servre in the current class. But the other 2 call a different class which has nothing to with the current class.

2

Util package

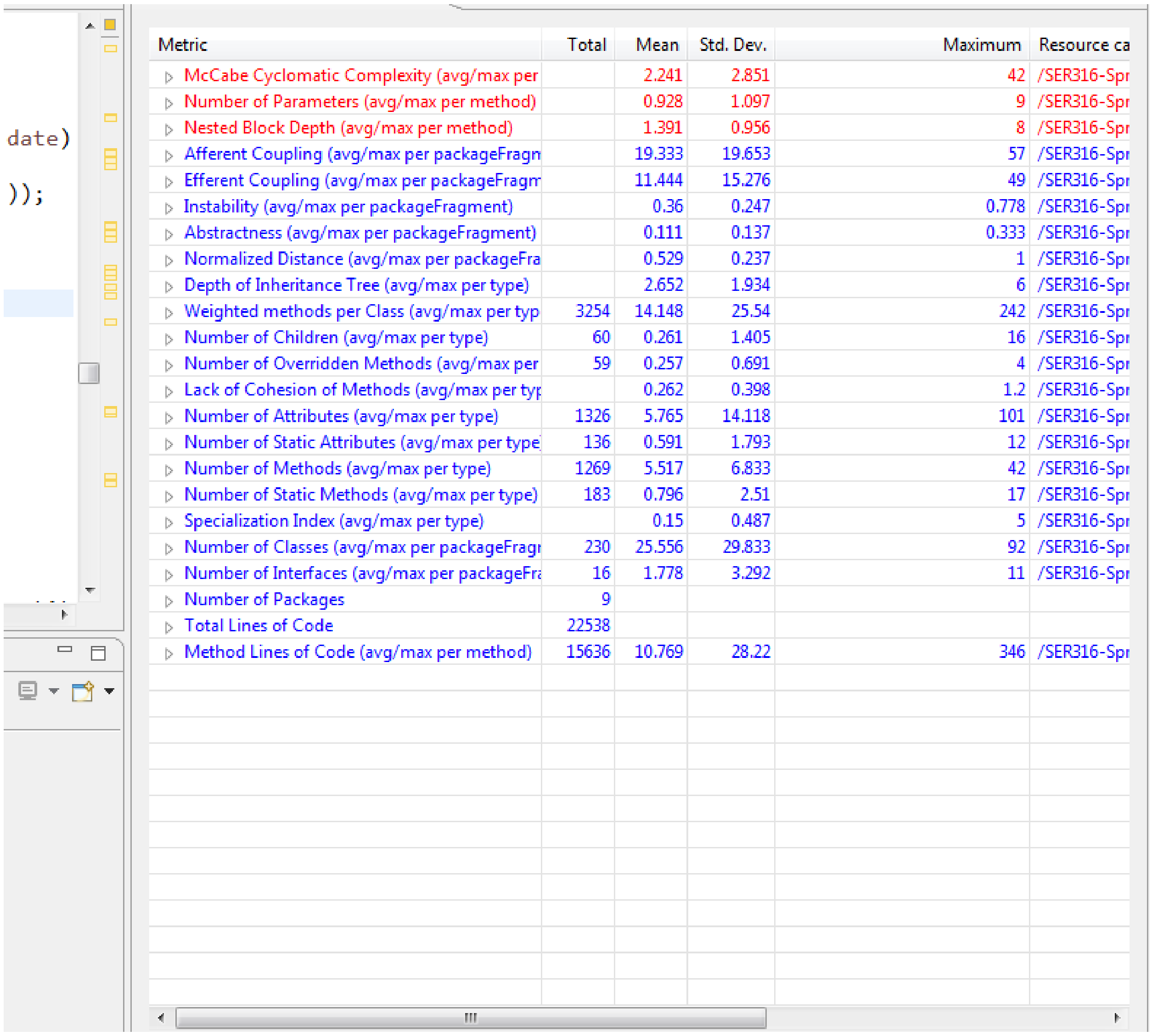
3

Ui package

Worst quality

I think the htmleditor class. It has the highest LOC, it has tons of attributes, it has over 80 + methods, it just doesn’t seem to fit quite in with the project, perhaps it can be refactored to not include all those uncessary functioanilities.

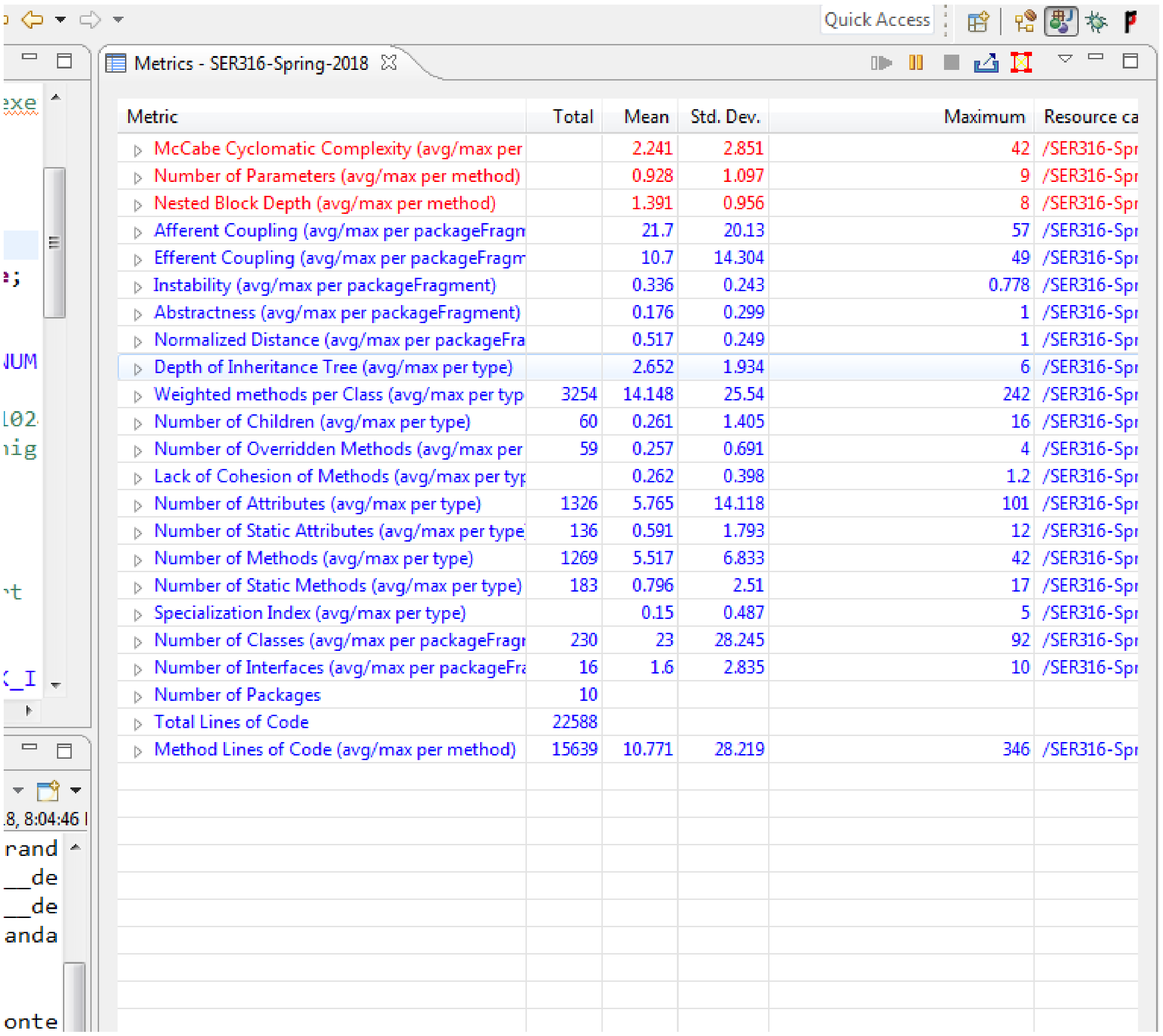
Task 2: Eclipse refactoring (25 points)



```
date)
));
```

Metric	Total	Mean	Std. Dev.	Maximum	Resource ca
McCabe Cyclomatic Complexity (avg/max per		2.241	2.851	42	/SER316-Spr
Number of Parameters (avg/max per method)		0.928	1.097	9	/SER316-Spr
Nested Block Depth (avg/max per method)		1.391	0.956	8	/SER316-Spr
Afferent Coupling (avg/max per packageFragn		19.333	19.653	57	/SER316-Spr
Efferent Coupling (avg/max per packageFragm		11.444	15.276	49	/SER316-Spr
Instability (avg/max per packageFragment)		0.36	0.247	0.778	/SER316-Spr
Abstractness (avg/max per packageFragment)		0.111	0.137	0.333	/SER316-Spr
Normalized Distance (avg/max per packageFra		0.529	0.237	1	/SER316-Spr
Depth of Inheritance Tree (avg/max per type)		2.652	1.934	6	/SER316-Spr
Weighted methods per Class (avg/max per typ	3254	14.148	25.54	242	/SER316-Spr
Number of Children (avg/max per type)	60	0.261	1.405	16	/SER316-Spr
Number of Overridden Methods (avg/max per	59	0.257	0.691	4	/SER316-Spr
Lack of Cohesion of Methods (avg/max per typ		0.262	0.398	1.2	/SER316-Spr
Number of Attributes (avg/max per type)	1326	5.765	14.118	101	/SER316-Spr
Number of Static Attributes (avg/max per type	136	0.591	1.793	12	/SER316-Spr
Number of Methods (avg/max per type)	1269	5.517	6.833	42	/SER316-Spr
Number of Static Methods (avg/max per type)	183	0.796	2.51	17	/SER316-Spr
Specialization Index (avg/max per type)		0.15	0.487	5	/SER316-Spr
Number of Classes (avg/max per packageFragı	230	25.556	29.833	92	/SER316-Spr
Number of Interfaces (avg/max per packageFra	16	1.778	3.292	11	/SER316-Spr
Number of Packages	9				
Total Lines of Code	22538				
Method Lines of Code (avg/max per method)	15636	10.769	28.22	346	/SER316-Spr

7.


Quick Access

Metrics - SER316-Spring-2018

Metric	Total	Mean	Std. Dev.	Maximum	Resource ca
McCabe Cyclomatic Complexity (avg/max per		2.241	2.851	42	/SER316-Spr
Number of Parameters (avg/max per method)		0.928	1.097	9	/SER316-Spr
Nested Block Depth (avg/max per method)		1.391	0.956	8	/SER316-Spr
Afferent Coupling (avg/max per packageFragn		21.7	20.13	57	/SER316-Spr
Efferent Coupling (avg/max per packageFragm		10.7	14.304	49	/SER316-Spr
Instability (avg/max per packageFragment)		0.336	0.243	0.778	/SER316-Spr
Abstractness (avg/max per packageFragment)		0.176	0.299	1	/SER316-Spr
Normalized Distance (avg/max per packageFra		0.517	0.249	1	/SER316-Spr
Depth of Inheritance Tree (avg/max per type)		2.652	1.934	6	/SER316-Spr
Weighted methods per Class (avg/max per typ	3254	14.148	25.54	242	/SER316-Spr
Number of Children (avg/max per type)	60	0.261	1.405	16	/SER316-Spr
Number of Overridden Methods (avg/max per	59	0.257	0.691	4	/SER316-Spr
Lack of Cohesion of Methods (avg/max per typ		0.262	0.398	1.2	/SER316-Spr
Number of Attributes (avg/max per type)	1326	5.765	14.118	101	/SER316-Spr
Number of Static Attributes (avg/max per type	136	0.591	1.793	12	/SER316-Spr
Number of Methods (avg/max per type)	1269	5.517	6.833	42	/SER316-Spr
Number of Static Methods (avg/max per type)	183	0.796	2.51	17	/SER316-Spr
Specialization Index (avg/max per type)		0.15	0.487	5	/SER316-Spr
Number of Classes (avg/max per packageFragi	230	23	28.245	92	/SER316-Spr
Number of Interfaces (avg/max per packageFra	16	1.6	2.835	10	/SER316-Spr
Number of Packages	10				
Total Lines of Code	22588				
Method Lines of Code (avg/max per method)	15639	10.771	28.219	346	/SER316-Spr

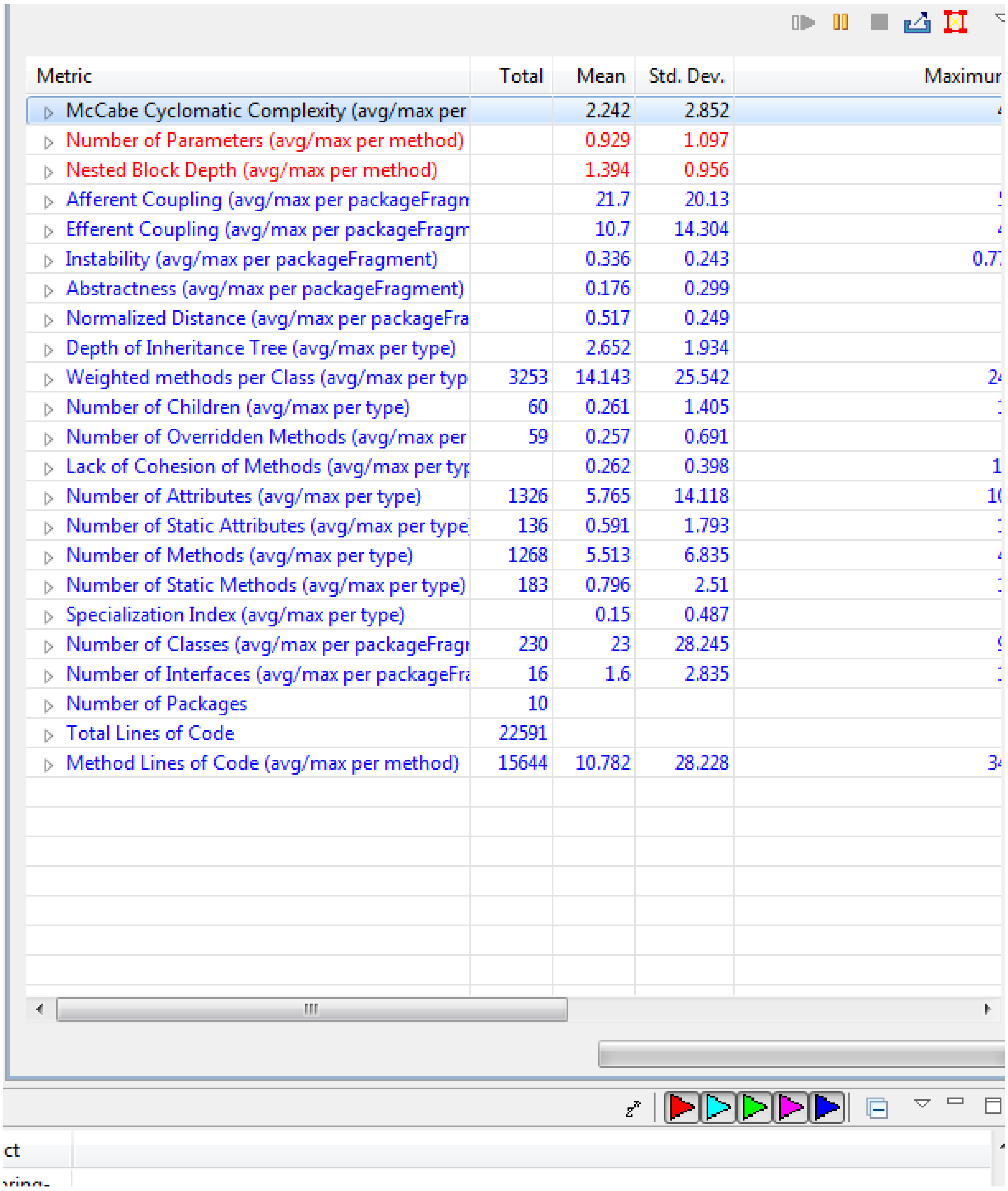
Yeah the affrent coupling as well as the efferent coupling went way down ☺

Task 3: Find code smells and refactor

1. Removed the constructor in DefaultEventNotifer since it was not needed.

2. Change code so that it includes proper if {} blocks and change short variable name from d to day

3.



Metric	Total	Mean	Std. Dev.	Maximur
McCabe Cyclomatic Complexity (avg/max per		2.242	2.852	4
Number of Parameters (avg/max per method)		0.929	1.097	
Nested Block Depth (avg/max per method)		1.394	0.956	
Afferent Coupling (avg/max per packageFragm		21.7	20.13	5
Efferent Coupling (avg/max per packageFragm		10.7	14.304	4
Instability (avg/max per packageFragment)		0.336	0.243	0.7
Abstractness (avg/max per packageFragment)		0.176	0.299	
Normalized Distance (avg/max per packageFra		0.517	0.249	
Depth of Inheritance Tree (avg/max per type)		2.652	1.934	
Weighted methods per Class (avg/max per typ	3253	14.143	25.542	24
Number of Children (avg/max per type)	60	0.261	1.405	1
Number of Overridden Methods (avg/max per	59	0.257	0.691	
Lack of Cohesion of Methods (avg/max per typ		0.262	0.398	1
Number of Attributes (avg/max per type)	1326	5.765	14.118	10
Number of Static Attributes (avg/max per type	136	0.591	1.793	1
Number of Methods (avg/max per type)	1268	5.513	6.835	4
Number of Static Methods (avg/max per type)	183	0.796	2.51	1
Specialization Index (avg/max per type)		0.15	0.487	
Number of Classes (avg/max per packageFragr	230	23	28.245	9
Number of Interfaces (avg/max per packageFra	16	1.6	2.835	1
Number of Packages	10			
Total Lines of Code	22591			
Method Lines of Code (avg/max per method)	15644	10.782	28.228	34

ct