Task 1:

Size:

1. Total Lines of Code: 2187
2. Largest file – EventsManager.java – LOC – 329
3. The metircs tool calculated 28 lines of code for the CurrentNote.java file by counting the lines that have actual active code on it; in other words, the total lines in the file minus the blank lines and commented lines.

Cohesion:

1. LCOM2 is the Lack of Cohesion that is calculated within methods of a given class. The way it is calculated is by summing the total amount of intersections of method argument types with the list of all argument types of all methods with the given class.
2. The class with the highest Cohesion is TaskImpl.java and this is probably because it has a very high amount of methods that take in many different argument types, therefore making the cohesion that the class accomplished much higher than that of other classes.

Complexity:

1. The mean of the cyclomatic complexity in the main package is 1.746.
2. EventsManager.java – 2.5
3. I reduced the cyclomatic complexity in the EventsManager class by modifying the Create day function and separating out one of it's if checks into a separate function; see comment "TASK 1 – Complexity" to find it in the code"

Package-Level Coupling:

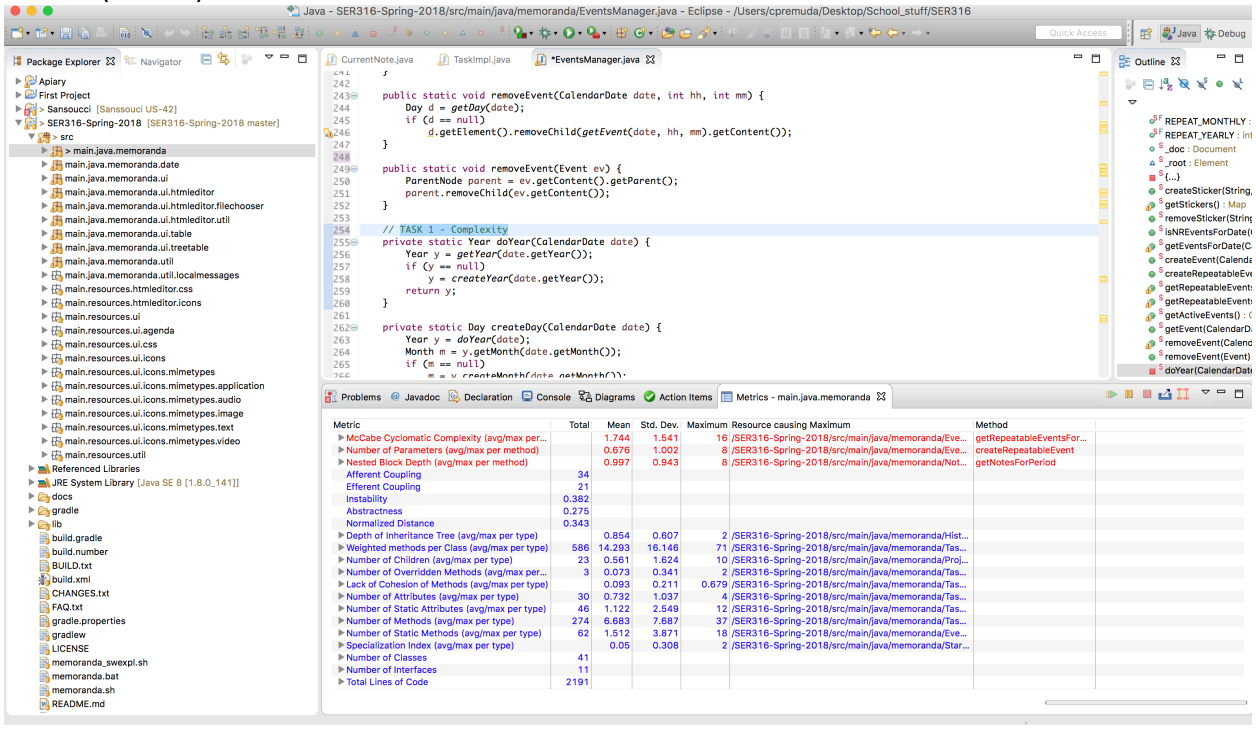
1. Afferent refers to other classes outside of the package that depend on classes from the package to show how complex the package is as a dependency. Efferent refers to how many classes from outside packages that are used in the package (and therefore are dependencies of the package) to show complexity.
2. Worst afferent – main.java.memoranda.util – 57
3. Worst efferent - main.java.memoranda.ui – 49

Worst Quality:

I chose the NoteListImpl.java as the class with the worst quality. Although many other classes are in the same ballpark as this one as far as metrics (like History.java, and EventsManager.java), NoteLIstImpl.java has a very high cyclomatic complexity, and also a very high rating for nested block depth, much higher than other classes. Additionally, this class had a higher ranking on lack of cohesion than other classes (most other classes didn't have this element of lack of cohesion at all; in other words, most other classes got a 0 in the category). All these factors combined with the other factors (such as high LOC count, static methods, etc.) is the reason I think this class has the worst ranking in quality.

Task 2:

1. (before)



Java - SER316-Spring-2018/src/main/java/memoranda/EventsManager.java - Eclipse - /Users/cpremuda/Desktop/School_stuff/SER316

```
Package Explorer
▶ Apiary
▶ First Project
▶ > Sansoucci [Sanssouci US-42]
▼ > SER316-Spring-2018 [SER316-Spring-2018 master]
  ▼ > src
    ▶ > main.java.memoranda
    ▶ main.java.memoranda.date
    ▶ main.java.memoranda.ui
    ▶ main.java.memoranda.ui.htmleditor
    ▶ main.java.memoranda.ui.htmleditor.filechooser
    ▶ main.java.memoranda.ui.htmleditor.util
    ▶ main.java.memoranda.ui.table
    ▶ main.java.memoranda.ui.treetable
    ▶ main.java.memoranda.util
    ▶ main.java.memoranda.util.localmessages
    ▶ main.resources.htmleditor.css
    ▶ main.resources.htmleditor.icons
    ▶ main.resources.ui
    ▶ main.resources.ui.agenda
    ▶ main.resources.ui.css
    ▶ main.resources.ui.icons
    ▶ main.resources.ui.icons.mimetypes
    ▶ main.resources.ui.icons.mimetypes.application
    ▶ main.resources.ui.icons.mimetypes.audio
    ▶ main.resources.ui.icons.mimetypes.image
    ▶ main.resources.ui.icons.mimetypes.text
    ▶ main.resources.ui.icons.mimetypes.video
    ▶ main.resources.util
  ▶ Referenced Libraries
  ▶ JRE System Library [Java SE 8 [1.8.0_141]]
  ▶ docs
  ▶ gradle
  ▶ lib
    build.gradle
    build.number
    BUILD.txt
    build.xml
    CHANGES.txt
    FAQ.txt
    gradle.properties
    gradlew
    LICENSE
    memoranda_swexpl.sh
    memoranda.bat
    memoranda.sh
    README.md
```

CurrentNote.java | TaskImpl.java | *EventsManager.java

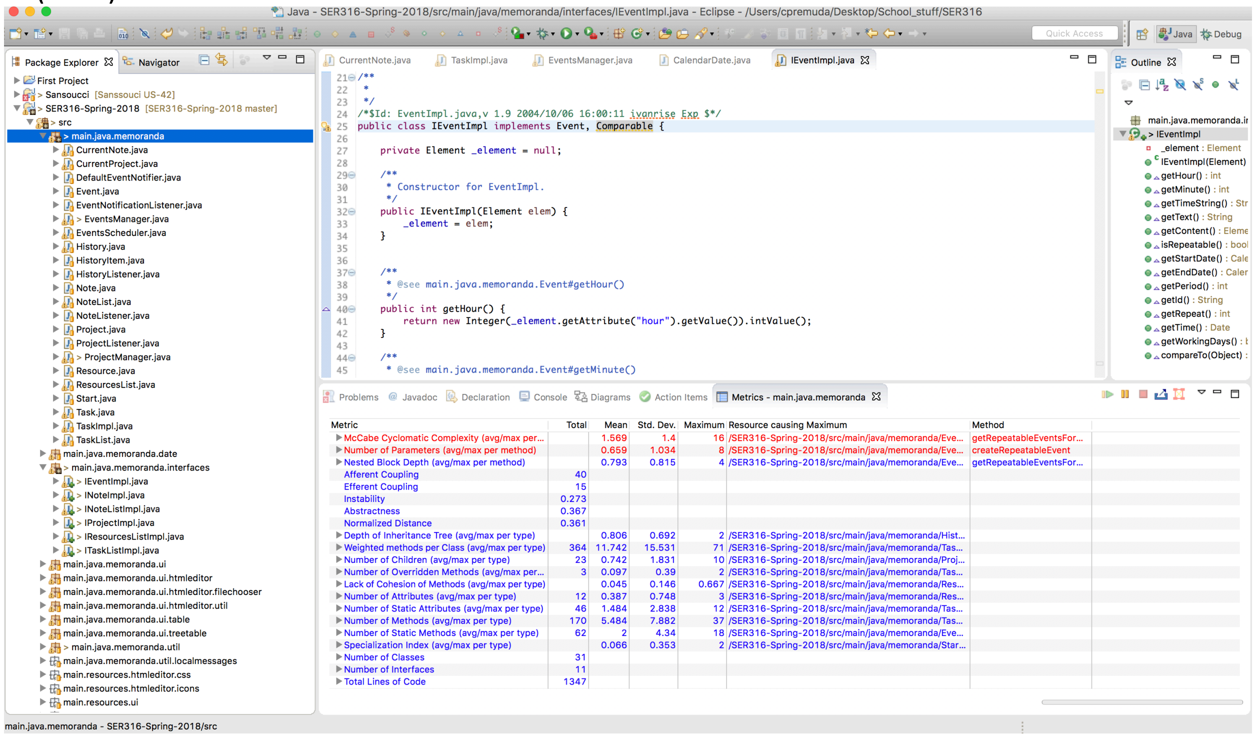
```java
    public static void removeEvent(CalendarDate date, int hh, int mm) {
        Day d = getDay(date);
        if (d == null)
            d.getElement().removeChild(getEvent(date, hh, mm).getContent());
    }

    public static void removeEvent(Event ev) {
        ParentNode parent = ev.getContent().getParent();
        parent.removeChild(ev.getContent());
    }

    // TASK 1 - Complexity
    private static Year doYear(CalendarDate date) {
        Year y = getYear(date.getYear());
        if (y == null)
            y = createYear(date.getYear());
        return y;
    }

    private static Day createDay(CalendarDate date) {
        Year y = doYear(date);
        Month m = y.getMonth(date.getMonth());
        if (m == null)
```

```
Outline
REPEAT_MONTHLY :
REPEAT_YEARLY : int
_doc : Document
_root : Element
{...}
createSticker(String,
getStickers() : Map
removeSticker(String
isNREventsForDate(
getEventsForDate(C
createEvent(Calenda
createRepeatableEve
getRepeatableEvent
getRepeatableEvent
getActiveEvents() : C
getEvent(CalendarD
removeEvent(Calend
removeEvent(Event)
doYear(CalendarDate
```

Problems | Javadoc | Declaration | Console | Diagrams | Action Items | Metrics - main.java.memoranda

Metric	Total	Mean	Std. Dev.	Maximum	Resource causing Maximum	Method
▶ McCabe Cyclomatic Complexity (avg/max per...		1.744	1.541	16	/SER316-Spring-2018/src/main/java/memoranda/Eve...	getRepeatableEventsFor...
▶ Number of Parameters (avg/max per method)		0.676	1.002	8	/SER316-Spring-2018/src/main/java/memoranda/Eve...	createRepeatableEvent
▶ Nested Block Depth (avg/max per method)		0.997	0.943	8	/SER316-Spring-2018/src/main/java/memoranda/Not...	getNotesForPeriod
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/Hist...	
▶ Weighted methods per Class (avg/max per type)	586	14.293	16.146	71	/SER316-Spring-2018/src/main/java/memoranda/Tas...	
▶ Number of Children (avg/max per type)	23	0.561	1.624	10	/SER316-Spring-2018/src/main/java/memoranda/Proj...	
▶ Number of Overridden Methods (avg/max per...	3	0.073	0.341	2	/SER316-Spring-2018/src/main/java/memoranda/Tas...	
▶ Lack of Cohesion of Methods (avg/max per type)		0.093	0.211	0.679	/SER316-Spring-2018/src/main/java/memoranda/Tas...	
▶ Number of Attributes (avg/max per type)	30	0.732	1.037	4	/SER316-Spring-2018/src/main/java/memoranda/Tas...	
▶ Number of Static Attributes (avg/max per type)	46	1.122	2.549	12	/SER316-Spring-2018/src/main/java/memoranda/Tas...	
▶ Number of Methods (avg/max per type)	274	6.683	7.687	37	/SER316-Spring-2018/src/main/java/memoranda/Tas...	
▶ Number of Static Methods (avg/max per type)	62	1.512	3.871	18	/SER316-Spring-2018/src/main/java/memoranda/Eve...	
▶ Specialization Index (avg/max per type)		0.05	0.308	2	/SER316-Spring-2018/src/main/java/memoranda/Star...	
▶ Number of Classes	41					
▶ Number of Interfaces	11					
▶ Total Lines of Code	2191					

7. (after)



Java - SER316-Spring-2018/src/main/java/memoranda/interfaces/IEventImpl.java - Eclipse - /Users/cpremuda/Desktop/School_stuff/SER316

```
Package Explorer | Navigator
▶ First Project
▶ > Sansoucci [Sanssouci US-42]
▼ > SER316-Spring-2018 [SER316-Spring-2018 master]
  ▼ > src
    ▼ > main.java.memoranda
      ▶ CurrentNote.java
      ▶ CurrentProject.java
      ▶ DefaultEventNotifier.java
      ▶ Event.java
      ▶ EventNotificationListener.java
      ▶ > EventsManager.java
      ▶ EventsScheduler.java
      ▶ History.java
      ▶ HistoryItem.java
      ▶ HistoryListener.java
      ▶ Note.java
      ▶ NoteList.java
      ▶ NoteListener.java
      ▶ Project.java
      ▶ ProjectListener.java
      ▶ > ProjectManager.java
      ▶ Resource.java
      ▶ ResourcesList.java
      ▶ Start.java
      ▶ Task.java
      ▶ TaskImpl.java
      ▶ TaskList.java
    ▶ main.java.memoranda.date
    ▼ > main.java.memoranda.interfaces
      ▶ > IEventImpl.java
      ▶ > INoteImpl.java
      ▶ > INoteListImpl.java
      ▶ > IProjectImpl.java
      ▶ > IResourcesListImpl.java
      ▶ > ITaskListImpl.java
    ▶ main.java.memoranda.ui
    ▶ main.java.memoranda.ui.htmleditor
    ▶ main.java.memoranda.ui.htmleditor.filechooser
    ▶ main.java.memoranda.ui.htmleditor.util
    ▶ main.java.memoranda.ui.table
    ▶ main.java.memoranda.ui.treetable
    ▶ > main.java.memoranda.util
    ▶ main.java.memoranda.util.localmessages
    ▶ main.resources.htmleditor.css
    ▶ main.resources.htmleditor.icons
    ▶ main.resources.ui
```

CurrentNote.java | TaskImpl.java | EventsManager.java | CalendarDate.java | IEventImpl.java

```java
/**
 *
 */
/*$Id: EventImpl.java,v 1.9 2004/10/06 16:00:11 ivanrise Exp $*/
public class IEventImpl implements Event, Comparable {

    private Element _element = null;

    /**
     * Constructor for EventImpl.
     */
    public IEventImpl(Element elem) {
        _element = elem;
    }


    /**
     * @see main.java.memoranda.Event#getHour()
     */
    public int getHour() {
        return new Integer(_element.getAttribute("hour").getValue()).intValue();
    }

    /**
     * @see main.java.memoranda.Event#getMinute()
```

```
Outline
main.java.memoranda.i
▼ > IEventImpl
  _element : Element
  IEventImpl(Element)
  getHour() : int
  getMinute() : int
  getTimeString() : Str
  getText() : String
  getContent() : Eleme
  isRepeatable() : bool
  getStartDate() : Cale
  getEndDate() : Caler
  getPeriod() : int
  getId() : String
  getRepeat() : int
  getTime() : Date
  getWorkingDays() : 
  compareTo(Object) :
```

Problems | Javadoc | Declaration | Console | Diagrams | Action Items | Metrics - main.java.memoranda

Metric	Total	Mean	Std. Dev.	Maximum	Resource causing Maximum	Method
▶ McCabe Cyclomatic Complexity (avg/max per...		1.569	1.4	16	/SER316-Spring-2018/src/main/java/memoranda/Eve...	getRepeatableEventsFor...
▶ Number of Parameters (avg/max per method)		0.659	1.034	8	/SER316-Spring-2018/src/main/java/memoranda/Eve...	createRepeatableEvent
▶ Nested Block Depth (avg/max per method)		0.793	0.815	4	/SER316-Spring-2018/src/main/java/memoranda/Eve...	getRepeatableEventsFor...
Afferent Coupling	40					
Efferent Coupling	15					
Instability	0.273					
Abstractness	0.367					
Normalized Distance	0.361					
▶ Depth of Inheritance Tree (avg/max per type)		0.806	0.692	2	/SER316-Spring-2018/src/main/java/memoranda/Hist...	
▶ Weighted methods per Class (avg/max per type)	364	11.742	15.531	71	/SER316-Spring-2018/src/main/java/memoranda/Tas...	
▶ Number of Children (avg/max per type)	23	0.742	1.831	10	/SER316-Spring-2018/src/main/java/memoranda/Proj...	
▶ Number of Overridden Methods (avg/max per...	3	0.097	0.39	2	/SER316-Spring-2018/src/main/java/memoranda/Tas...	
▶ Lack of Cohesion of Methods (avg/max per type)		0.045	0.146	0.667	/SER316-Spring-2018/src/main/java/memoranda/Res...	
▶ Number of Attributes (avg/max per type)	12	0.387	0.748	3	/SER316-Spring-2018/src/main/java/memoranda/Res...	
▶ Number of Static Attributes (avg/max per type)	46	1.484	2.838	12	/SER316-Spring-2018/src/main/java/memoranda/Tas...	
▶ Number of Methods (avg/max per type)	170	5.484	7.882	37	/SER316-Spring-2018/src/main/java/memoranda/Tas...	
▶ Number of Static Methods (avg/max per type)	62	2	4.34	18	/SER316-Spring-2018/src/main/java/memoranda/Eve...	
▶ Specialization Index (avg/max per type)		0.066	0.353	2	/SER316-Spring-2018/src/main/java/memoranda/Star...	
▶ Number of Classes	31					
▶ Number of Interfaces	11					
▶ Total Lines of Code	1347					

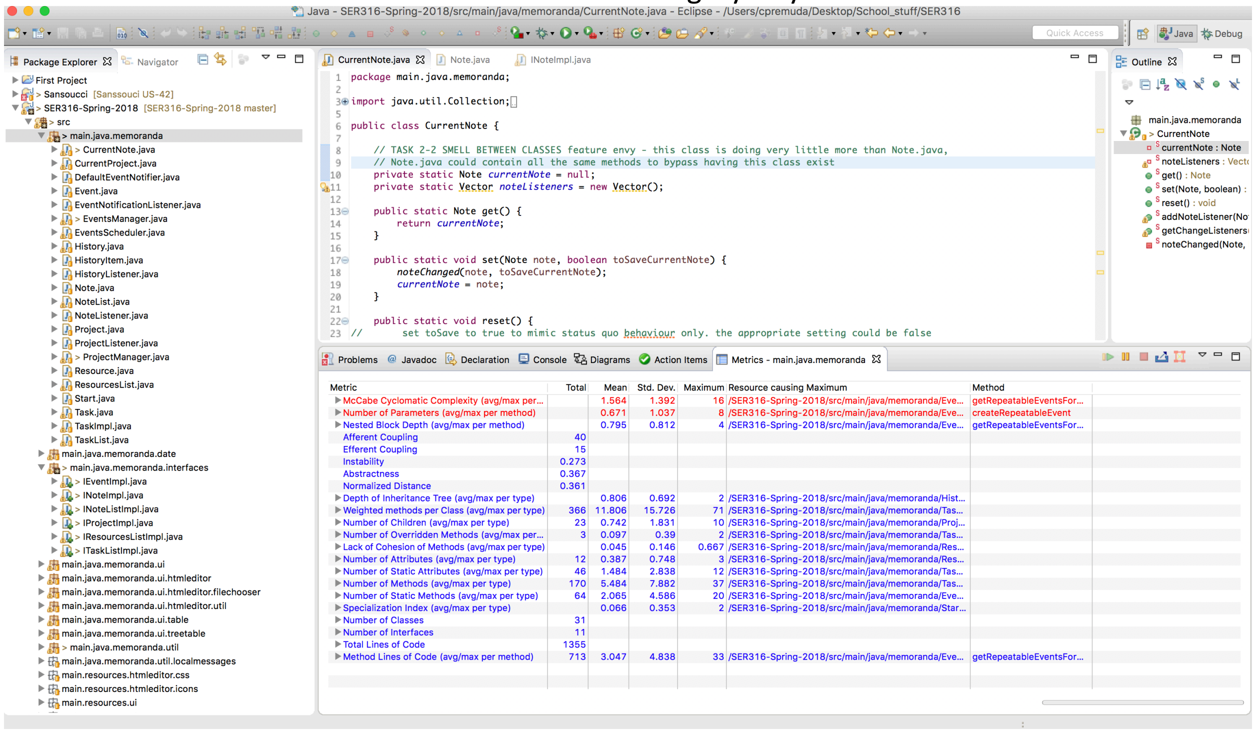
main.java.memoranda - SER316-Spring-2018/src

8. Many metrics improved after moving all the interface classes out of the package just by nature of there being less code, and therefore, less things wrong with the code. For example,

the cyclomatic complexity improved (the metric went down) because many of the classes that we moved happened to have methods with high cyclomatic complexity, so the main.java.memoranda no longer reflects that complexity (however, the new package will probably be a little high with this metric).

Task 3

1. The smell I decided to refactor was a cyclomatic complexity smell (that I noticed in task 1 where a function should have been broken up into smaller functions. The function that I refactored is called createDay() in the EventsManager.java class which is in the main.java.memoranda package. The was able to fix this code smell by making separate functions to the all 3 parts of the date before returning the day; more specifically, I broke out functions to first get the year, and then the month, and then with day all with null checks in place.
2. The code smell I identified between classes is in CurrentNote.java where I noticed this class does little more than the Note.java class already does. The refactoring that should occur should be to combine the two classes so that Note also keeps track of the current note as well. This smell could also fall under the category lazy class.



```java
package main.java.memoranda;

import java.util.Collection;

public class CurrentNote {

    // TASK 2-2 SMELL BETWEEN CLASSES feature envy - this class is doing very little more than Note.java,
    // Note.java could contain all the same methods to bypass having this class exist
    private static Note currentNote = null;
    private static Vector noteListeners = new Vector();

    public static Note get() {
        return currentNote;
    }

    public static void set(Note note, boolean toSaveCurrentNote) {
        noteChanged(note, toSaveCurrentNote);
        currentNote = note;
    }

    public static void reset() {
//        set toSave to true to mimic status quo behaviour only. the appropriate setting could be false
```

Metric	Total	Mean	Std. Dev.	Maximum	Resource causing Maximum	Method
McCabe Cyclomatic Complexity (avg/max per...		1.564	1.392	16	/SER316-Spring-2018/src/main/java/memoranda/Eve...	getRepeatableEventsFor...
Number of Parameters (avg/max per method)		0.671	1.037	8	/SER316-Spring-2018/src/main/java/memoranda/Eve...	createRepeatableEvent
Nested Block Depth (avg/max per method)		0.795	0.812	4	/SER316-Spring-2018/src/main/java/memoranda/Eve...	getRepeatableEventsFor...
Afferent Coupling	40					
Efferent Coupling	15					
Instability	0.273					
Abstractness	0.367					
Normalized Distance	0.361					
Depth of Inheritance Tree (avg/max per type)		0.806	0.692	2	/SER316-Spring-2018/src/main/java/memoranda/Hist...	
Weighted methods per Class (avg/max per type)	366	11.806	15.726	71	/SER316-Spring-2018/src/main/java/memoranda/Tas...	
Number of Children (avg/max per type)	23	0.742	1.831	10	/SER316-Spring-2018/src/main/java/memoranda/Proj...	
Number of Overridden Methods (avg/max per...	3	0.097	0.39	2	/SER316-Spring-2018/src/main/java/memoranda/Tas...	
Lack of Cohesion of Methods (avg/max per type)		0.045	0.146	0.667	/SER316-Spring-2018/src/main/java/memoranda/Res...	
Number of Attributes (avg/max per type)	12	0.387	0.748	3	/SER316-Spring-2018/src/main/java/memoranda/Res...	
Number of Static Attributes (avg/max per type)	46	1.484	2.838	12	/SER316-Spring-2018/src/main/java/memoranda/Tas...	
Number of Methods (avg/max per type)	170	5.484	7.882	37	/SER316-Spring-2018/src/main/java/memoranda/Tas...	
Number of Static Methods (avg/max per type)	64	2.065	4.586	20	/SER316-Spring-2018/src/main/java/memoranda/Eve...	
Specialization Index (avg/max per type)		0.066	0.353	2	/SER316-Spring-2018/src/main/java/memoranda/Star...	
Number of Classes	31					
Number of Interfaces	11					
Total Lines of Code	1355					
Method Lines of Code (avg/max per method)	713	3.047	4.838	33	/SER316-Spring-2018/src/main/java/memoranda/Eve...	getRepeatableEventsFor...

3.
4. One metric that changed after my refactoring is cyclomatic complexity improved after my refactoring. The main reason for this is probably because of the refactor I did where I split a method out into many functions, therefore, reducing this metric. Since this metric went down, the change was for the better.