Task 1:

Size:

1. Total LOC (in main.java.memoranda package per Instructor in Slack): 2187
2. Largest Single Code File in Project: EventsManager.java

   EventsManager.java total LOC: 329
3. It counted every line with any characters including comments and closing braces.

Cohesion:

1. $$LCOM2 = 1 - \frac{sum\ of\ the\ number\ of\ methods\ which\ access\ a\ attribute}{(the\ number\ of\ methods\ in\ a\ class) * (the\ number\ of\ attributes\ in\ a\ class)}$$

   Which is to say LCOM2 is a percentage of all the methods which do not access a specific variable averaged over all the attributes. (From unattributed material provided by instructors)

2. HistoryItem.java has the highest cohesian because it has only 2 class attributes or variables. That reduces the divisor since the number of methods are multiplied by only 2. Meanwhile many of the methods are calling at least one or more methods so the dividend is large. Many of the classes have LCOM of 0 which is not meaningful because in LCOM2 if there are zero class attributes or method we are attempting to divide by zero which is undefined so LCOM2 doesn't work.

Complexity:

1. main.java.memoranda has a mean McCabe Cyclomatic complexity of 1.746.
2. EventsManager.java has the worst McCabe Cyclomatic complexity with a mean of 2.5 and a max of 16.
3. In EventsManager.java I redid the numerous nested else ifs in getRepeatableEventsForDate as a single if test with many and and ors. I then refactored that entire block out of the method and into a private static method called repeatableEventShouldBeAdded(ev, date)
   That reduced the McCabe Cyclomatic complexity of the class from a mean of 2.5 to a mean of 2.152. But more importantly it **reduced the maximum McCabe Cyclomatic complexity for the class from 16 to 6.**

Package-Level Coupling:

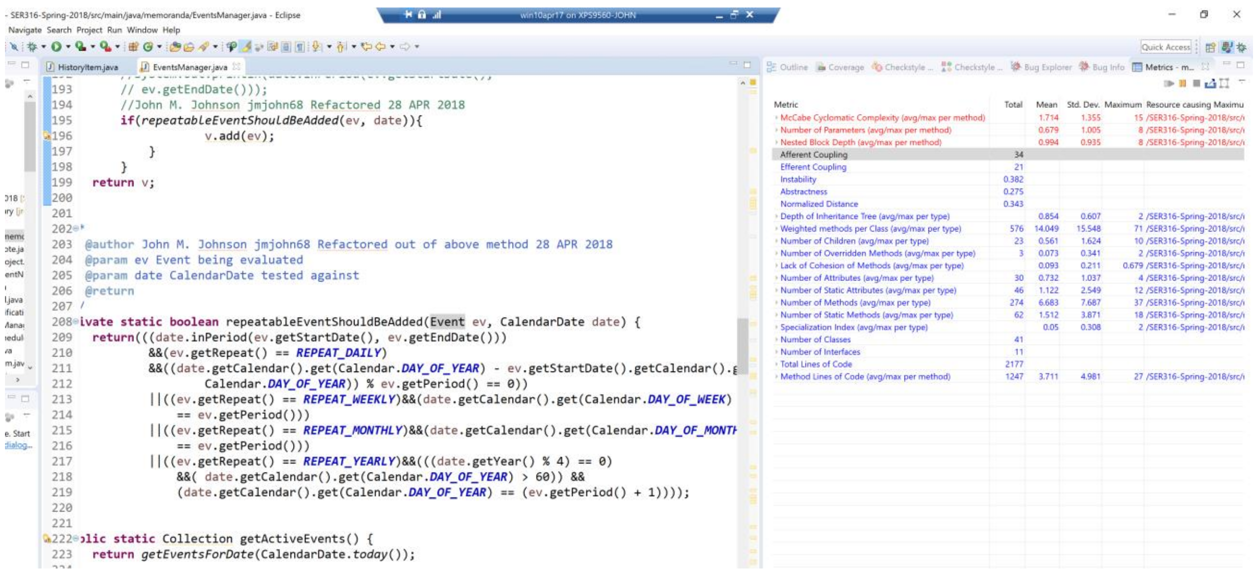
1. Afferent coupling is a measure of external package dependency on the package being measured. Efferent coupling is a measure of our package's dependency on external packages. This could also be stated as Afferent is incoming dependencies and efferent is outgoing dependencies.
2. main.java.memoranda.util has the worst afferent coupling with 57.
3. Main.java.memoranda.ui has the worst efferent coupling with 49.

Worst Quality:

Before refactoring EventsManager.java had the worst quality in the package. It had the highest maximum McCabe Cyclomatic Complexity in the package at 16. That complexity made the class very difficult to understand which would have led to maintenance issues after the software was completed. Additionally, it is the largest class by measuring lines of code. Even after refactoring it has 319 LOC. This class should have been better written and originally broken up into smaller methods and probably should have been multiple classes. There are many commented out methods and statements further reducing the maintainability of the software.

Task 2:

Step1:


SER316-Spring-2018/src/main/java/memoranda/EventsManager.java - Eclipse
win10apr17 on XPS9560-JOHN
Navigate Search Project Run Window Help
HistoryItem.java
EventsManager.java
Outline
Coverage
Checkstyle ...
Bug Explorer
Bug Info
Metrics - m...
Quick Access

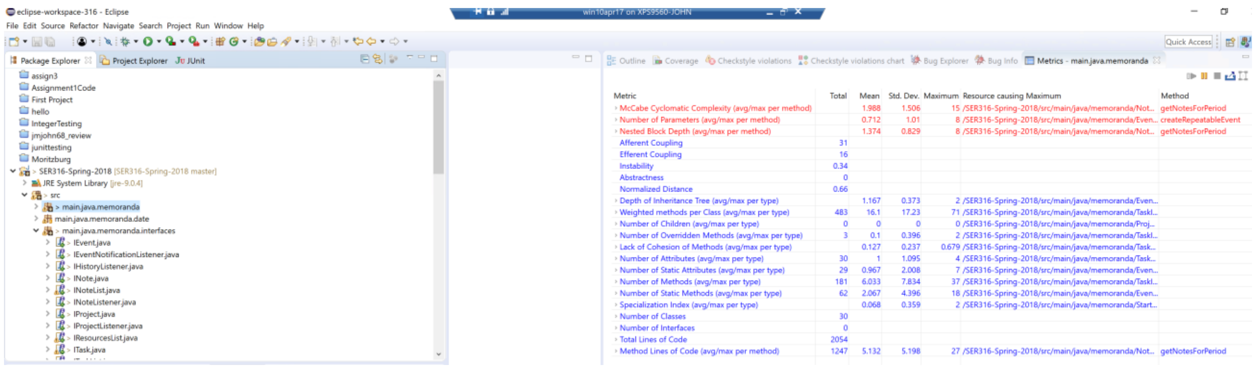
```
        // ev.getEndDate()));
        //John M. Johnson jmjohn68 Refactored 28 APR 2018
        if(repeatableEventShouldBeAdded(ev, date)){
                    v.add(ev);
            }
        }
    return v;


 *
 @author John M. Johnson jmjohn68 Refactored out of above method 28 APR 2018
 @param ev Event being evaluated
 @param date CalendarDate tested against
 @return
 /
ivate static boolean repeatableEventShouldBeAdded(Event ev, CalendarDate date) {
    return(((date.inPeriod(ev.getStartDate(), ev.getEndDate()))
            &&(ev.getRepeat() == REPEAT_DAILY)
            &&((date.getCalendar().get(Calendar.DAY_OF_YEAR) - ev.getStartDate().getCalendar().g
                    Calendar.DAY_OF_YEAR)) % ev.getPeriod() == 0))
            ||((ev.getRepeat() == REPEAT_WEEKLY)&&(date.getCalendar().get(Calendar.DAY_OF_WEEK)
                == ev.getPeriod()))
            ||((ev.getRepeat() == REPEAT_MONTHLY)&&(date.getCalendar().get(Calendar.DAY_OF_MONTH
                == ev.getPeriod()))
            ||((ev.getRepeat() == REPEAT_YEARLY)&&(((date.getYear() % 4) == 0)
                &&( date.getCalendar().get(Calendar.DAY_OF_YEAR) > 60)) &&
                (date.getCalendar().get(Calendar.DAY_OF_YEAR) == (ev.getPeriod() + 1))));


plic static Collection getActiveEvents() {
    return getEventsForDate(CalendarDate.today());
```

Metric	Total	Mean	Std. Dev.	Maximum	Resource causing Maximu
McCabe Cyclomatic Complexity (avg/max per method)		1.714	1.355	15	/SER316-Spring-2018/src/
Number of Parameters (avg/max per method)		0.679	1.005	8	/SER316-Spring-2018/src/
Nested Block Depth (avg/max per method)		0.994	0.935	8	/SER316-Spring-2018/src/
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/
Weighted methods per Class (avg/max per type)	576	14.049	15.548	71	/SER316-Spring-2018/src/
Number of Children (avg/max per type)	23	0.561	1.624	10	/SER316-Spring-2018/src/
Number of Overridden Methods (avg/max per type)	3	0.073	0.341	2	/SER316-Spring-2018/src/
Lack of Cohesion of Methods (avg/max per type)		0.093	0.211	0.679	/SER316-Spring-2018/src/
Number of Attributes (avg/max per type)	30	0.732	1.037	4	/SER316-Spring-2018/src/
Number of Static Attributes (avg/max per type)	46	1.122	2.549	12	/SER316-Spring-2018/src/
Number of Methods (avg/max per type)	274	6.683	7.687	37	/SER316-Spring-2018/src/
Number of Static Methods (avg/max per type)	62	1.512	3.871	18	/SER316-Spring-2018/src/
Specialization Index (avg/max per type)		0.05	0.308	2	/SER316-Spring-2018/src/
Number of Classes	41				
Number of Interfaces	11				
Total Lines of Code	2177				
Method Lines of Code (avg/max per method)	1247	3.711	4.981	27	/SER316-Spring-2018/src/

Step 7:



Metric	Total	Mean	Std. Dev.	Maximum	Resource causing Maximum	Method
McCabe Cyclomatic Complexity (avg/max per method)		1.988	1.506	15	/SER316-Spring-2018/src/main/java/memoranda/Not...	getNotesForPeriod
Number of Parameters (avg/max per method)		0.712	1.01	8	/SER316-Spring-2018/src/main/java/memoranda/Even...	createRepeatableEvent
Nested Block Depth (avg/max per method)		1.374	0.829	8	/SER316-Spring-2018/src/main/java/memoranda/Not...	getNotesForPeriod
Afferent Coupling	31					
Efferent Coupling	16					
Instability	0.34					
Abstractness	0					
Normalized Distance	0.66					
Depth of Inheritance Tree (avg/max per type)		1.167	0.373	2	/SER316-Spring-2018/src/main/java/memoranda/Even...	
Weighted methods per Class (avg/max per type)	483	16.1	17.23	71	/SER316-Spring-2018/src/main/java/memoranda/TaskI...	
Number of Children (avg/max per type)	0	0	0	0	/SER316-Spring-2018/src/main/java/memoranda/Proj...	
Number of Overridden Methods (avg/max per type)	3	0.1	0.396	2	/SER316-Spring-2018/src/main/java/memoranda/TaskI...	
Lack of Cohesion of Methods (avg/max per type)		0.127	0.237	0.679	/SER316-Spring-2018/src/main/java/memoranda/Task...	
Number of Attributes (avg/max per type)	30	1	1.095	4	/SER316-Spring-2018/src/main/java/memoranda/Task...	
Number of Static Attributes (avg/max per type)	29	0.967	2.008	7	/SER316-Spring-2018/src/main/java/memoranda/Even...	
Number of Methods (avg/max per type)	181	6.033	7.834	37	/SER316-Spring-2018/src/main/java/memoranda/TaskI...	
Number of Static Methods (avg/max per type)	62	2.067	4.396	18	/SER316-Spring-2018/src/main/java/memoranda/Even...	
Specialization Index (avg/max per type)		0.068	0.359	2	/SER316-Spring-2018/src/main/java/memoranda/Start...	
Number of Classes	30					
Number of Interfaces	0					
Total Lines of Code	2054					
Method Lines of Code (avg/max per method)	1247	5.132	5.198	27	/SER316-Spring-2018/src/main/java/memoranda/Not...	getNotesForPeriod

Step 8:

The Mean McCabe Cyclomatic complexity in the memoranda package got worse moving from 1.714 to 1.988. However the mean just got worse do to the interface classes with low CC being removed from the package. When the maximum CC is compared there was no change.

Task 3:

Step 1:

Main.java.memoranda.EventsManager.java's method createRepeatableEvent had the following parameters :(int type, CalendarDate startDate, CalendarDate endDate, int period, int hh, int mm, text, boolean workDays)

This was an obvious code smell of a too-long parameter. I refactored the parameter into one SingleEvent object.

public static IEvent createRepeatableEvent(SingleEvent parameterObject)

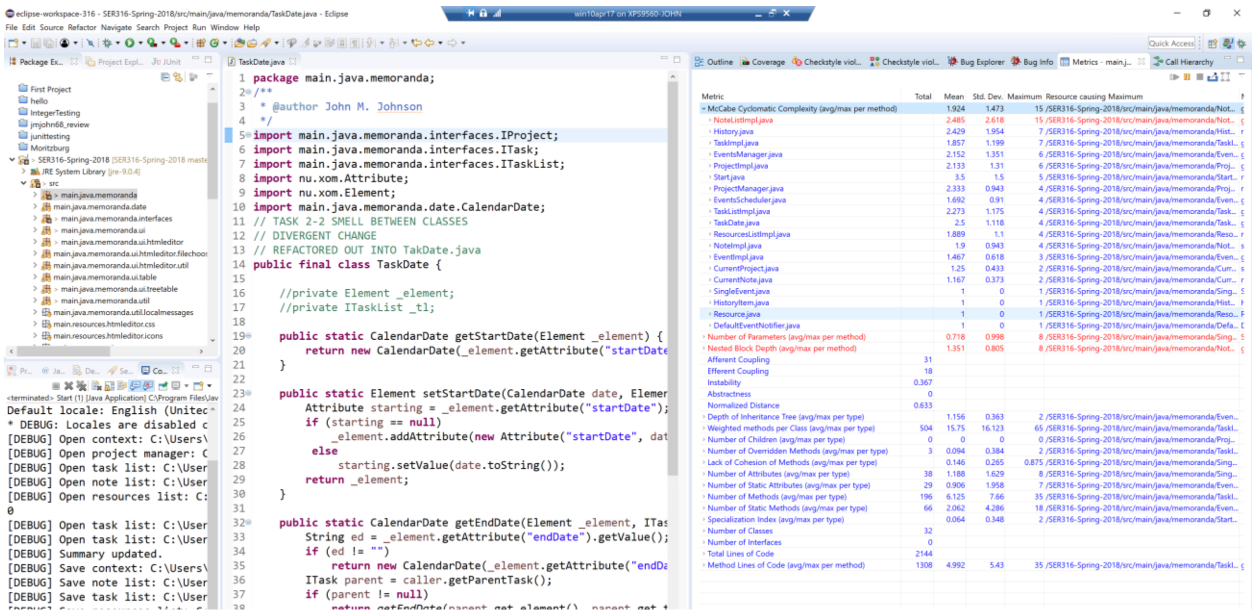
This makes the code more easily understood. Additionally as suggested by Fowler in extensive refactoring this construct could probably be reused.

Step 2:

Main.java.memoranda.TaskImpl.java had getStartDate, setStartDate, getEndDate, and setEnd Date which were an example of the divergent change which our powerpoint slides list as a smell between classes. I felt this was an example of divergent change because they did not really fit with the other methods of the too long class and all were grouped around their use of CalendarDate. I extracted those related methods into a new class TaskDates.

I had to manually refactor the rest of the program to work with TaskDates which took nearly two hours.

Step 3:


eclipse-workspace-316 - SER316-Spring-2018/src/main/java/memoranda/TaskDate.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
win10apr17 on XPS9560-JOHN
Quick Access
Package Ex...
Project Expl...
JUnit
First Project
hello
IntegerTesting
jmjohn68_review
junittesting
Moritzburg
SER316-Spring-2018 [SER316-Spring-2018 maste
JRE System Library [jre-9.0.4]
src
main.java.memoranda
main.java.memoranda.date
main.java.memoranda.interfaces
main.java.memoranda.ui
main.java.memoranda.ui.htmleditor
main.java.memoranda.ui.htmleditor.filechoos
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
TaskDate.java
Outline
Coverage
Checkstyle viol...
Checkstyle viol...
Bug Explorer
Bug Info
Metrics - main.j...
Call Hierarchy

```java
package main.java.memoranda;
/**
 * @author John M. Johnson
 */
import main.java.memoranda.interfaces.IProject;
import main.java.memoranda.interfaces.ITask;
import main.java.memoranda.interfaces.ITaskList;
import nu.xom.Attribute;
import nu.xom.Element;
import main.java.memoranda.date.CalendarDate;
// TASK 2-2 SMELL BETWEEN CLASSES
// DIVERGENT CHANGE
// REFACTORED OUT INTO TakDate.java
public final class TaskDate {

    //private Element _element;
    //private ITaskList _tl;

    public static CalendarDate getStartDate(Element _element) {
        return new CalendarDate(_element.getAttribute("startDate
    }

    public static Element setStartDate(CalendarDate date, Elemer
        Attribute starting = _element.getAttribute("startDate");
        if (starting == null)
            _element.addAttribute(new Attribute("startDate", dat
         else
             starting.setValue(date.toString());
        return _element;
    }

    public static CalendarDate getEndDate(Element _element, ITas
        String ed = _element.getAttribute("endDate").getValue();
        if (ed != "")
            return new CalendarDate(_element.getAttribute("endDa
        ITask parent = caller.getParentTask();
        if (parent != null)
```

```
<terminated> Start (1) [Java Application] C:\Program Files\Jav
Default locale: English (Unitec
* DEBUG: Locales are disabled c
[DEBUG] Open context: C:\Users\
[DEBUG] Open project manager: C
[DEBUG] Open task list: C:\User
[DEBUG] Open note list: C:\User
[DEBUG] Open resources list: C:
0
[DEBUG] Open task list: C:\User
[DEBUG] Open task list: C:\User
[DEBUG] Summary updated.
[DEBUG] Save context: C:\Users\
[DEBUG] Save note list: C:\User
[DEBUG] Save task list: C:\User
```

Metric	Total	Mean	Std. Dev.	Maximum	Resource causing Maximum
McCabe Cyclomatic Complexity (avg/max per method)		1.924	1.473	15	/SER316-Spring-2018/src/main/java/memoranda/Not...
NoteListImpl.java		2.485	2.618	15	/SER316-Spring-2018/src/main/java/memoranda/Not...
History.java		2.429	1.954	7	/SER316-Spring-2018/src/main/java/memoranda/Hist...
TaskImpl.java		1.857	1.199	7	/SER316-Spring-2018/src/main/java/memoranda/TaskI...
EventsManager.java		2.152	1.351	6	/SER316-Spring-2018/src/main/java/memoranda/Even...
ProjectImpl.java		2.133	1.31	6	/SER316-Spring-2018/src/main/java/memoranda/Proj...
Start.java		3.5	1.5	5	/SER316-Spring-2018/src/main/java/memoranda/Start...
ProjectManager.java		2.333	0.943	4	/SER316-Spring-2018/src/main/java/memoranda/Proj...
EventsScheduler.java		1.692	0.91	4	/SER316-Spring-2018/src/main/java/memoranda/Even...
TaskListImpl.java		2.273	1.175	4	/SER316-Spring-2018/src/main/java/memoranda/Task...
TaskDate.java		2.5	1.118	4	/SER316-Spring-2018/src/main/java/memoranda/Task...
ResourcesListImpl.java		1.889	1.1	4	/SER316-Spring-2018/src/main/java/memoranda/Reso...
NoteImpl.java		1.9	0.943	4	/SER316-Spring-2018/src/main/java/memoranda/Not...
EventImpl.java		1.467	0.618	3	/SER316-Spring-2018/src/main/java/memoranda/Even...
CurrentProject.java		1.25	0.433	2	/SER316-Spring-2018/src/main/java/memoranda/Curr...
CurrentNote.java		1.167	0.373	2	/SER316-Spring-2018/src/main/java/memoranda/Curr...
SingleEvent.java		1	0	1	/SER316-Spring-2018/src/main/java/memoranda/Sing...
HistoryItem.java		1	0	1	/SER316-Spring-2018/src/main/java/memoranda/Hist...
Resource.java		1	0	1	/SER316-Spring-2018/src/main/java/memoranda/Reso...
DefaultEventNotifier.java		1	0	1	/SER316-Spring-2018/src/main/java/memoranda/Defa...
Number of Parameters (avg/max per method)		0.718	0.998	8	/SER316-Spring-2018/src/main/java/memoranda/Sing...
Nested Block Depth (avg/max per method)		1.351	0.805	8	/SER316-Spring-2018/src/main/java/memoranda/Not...
Afferent Coupling	31				
Efferent Coupling	18				
Instability	0.367				
Abstractness	0				
Normalized Distance	0.633				
Depth of Inheritance Tree (avg/max per type)		1.156	0.363	2	/SER316-Spring-2018/src/main/java/memoranda/Even...
Weighted methods per Class (avg/max per type)	504	15.75	16.123	65	/SER316-Spring-2018/src/main/java/memoranda/TaskI...
Number of Children (avg/max per type)	0	0	0	0	/SER316-Spring-2018/src/main/java/memoranda/Proj...
Number of Overridden Methods (avg/max per type)	3	0.094	0.384	2	/SER316-Spring-2018/src/main/java/memoranda/TaskI...
Lack of Cohesion of Methods (avg/max per type)		0.146	0.265	0.875	/SER316-Spring-2018/src/main/java/memoranda/Sing...
Number of Attributes (avg/max per type)	38	1.188	1.629	8	/SER316-Spring-2018/src/main/java/memoranda/Sing...
Number of Static Attributes (avg/max per type)	29	0.906	1.958	7	/SER316-Spring-2018/src/main/java/memoranda/Even...
Number of Methods (avg/max per type)	196	6.125	7.66	35	/SER316-Spring-2018/src/main/java/memoranda/TaskI...
Number of Static Methods (avg/max per type)	66	2.062	4.286	18	/SER316-Spring-2018/src/main/java/memoranda/Even...
Specialization Index (avg/max per type)		0.064	0.348	2	/SER316-Spring-2018/src/main/java/memoranda/Start...
Number of Classes	32				
Number of Interfaces	0				
Total Lines of Code	2144				
Method Lines of Code (avg/max per method)	1308	4.992	5.43	35	/SER316-Spring-2018/src/main/java/memoranda/TaskI...

Step 4:

McCabe Cyclomatic Complexity got better because of my refactoring with the mean moving from 1.988 to 1.924. This was in part because my extracted class TaskDate had a maximum CC(G) of 4. TaskImpl's max CC(G) was only 7 after my refactoring.