Task 1

Size

1. 2187
2. EventsManager
3. It used the get() method because it is the first method.

Cohesion

1. It's the number of pairs of methods that do not share attributes minus the number of pairs that do. If that difference is greater than 0 then the answer is LCOM2. If the answer is negative, the answer is 0.
2. There are about many classes that have 0 for lack of cohesion wich means they have high cohesion, right? Sure! I don't know why the are so cohesive. So I'll pick task.java

Complexity

1. The complexity is 1.746
2. EventsManager. It is 2.5
3. EventImpl Line 139 getWorkingDays. It had an if statement that chose wether the Boolean return statement was true of false. I deleted the "if" and just returned the answer of the former if statement (the part in the parenthesis). The method dropped from a 3 on complexity to a 1.

Package-level Coupling

1. Afferent coupling are coupling where classes outside of the package need the classes within a packages. Efferent is when the classes inside the class need classes outside the class.
2. Main.java.memoranda.util. its at 57
3. Main.java.memoranda.ui its at 49

Worst Quality

1. EventManager because it has the highest complexity at a mean of 2.5. It also has a high number of parameters at 8. It had the most red (bad) lines in the Metrics window.

TASK 2

Metric	Total	Mean	Std. Dev.	Maximum	Resource causing Maximum	Method
McCabe Cyclomatic Complexity (avg/max per method)		1.74	1.546	16	/A7/src/main/java/memoranda/EventsMa...	getRepeatableEven...
Number of Parameters (avg/max per method)		0.675	1.004	8	/A7/src/main/java/memoranda/EventsMa...	createRepeatableEv...
Nested Block Depth (avg/max per method)		0.997	0.945	8	/A7/src/main/java/memoranda/NoteListI...	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	/A7/src/main/java/memoranda/Start.java	
Weighted methods per Class (avg/max per type)	583	14.22	16.06	71	/A7/src/main/java/memoranda/TaskImpl.j...	
Number of Children (avg/max per type)	23	0.561	1.624	10	/A7/src/main/java/memoranda/ProjectList...	
Number of Overridden Methods (avg/max per type)	3	0.073	0.341	2	/A7/src/main/java/memoranda/TaskImpl.j...	
Lack of Cohesion of Methods (avg/max per type)		0.093	0.211	0.679	/A7/src/main/java/memoranda/TaskListIm...	
Number of Attributes (avg/max per type)	30	0.732	1.037	4	/A7/src/main/java/memoranda/TaskListIm...	
Number of Static Attributes (avg/max per type)	46	1.122	2.549	12	/A7/src/main/java/memoranda/Task.java	
Number of Methods (avg/max per type)	274	6.683	7.687	37	/A7/src/main/java/memoranda/TaskImpl.j...	
Number of Static Methods (avg/max per type)	61	1.488	3.768	17	/A7/src/main/java/memoranda/EventsMa...	
Specialization Index (avg/max per type)		0.05	0.308	2	/A7/src/main/java/memoranda/Start.java	
Number of Classes	41					
Number of Interfaces	11					
Total Lines of Code	2186					
Method Lines of Code (avg/max per method)	1258	3.755	5.213	33	/A7/src/main/java/memoranda/EventsMa...	getRepeatableEven...

Metric	Total	Mean	Std. Dev.	Maximum	Resource causing Maximum	Method
McCabe Cyclomatic Complexity (avg/max per method)		2.046	1.748	16	/A7/src/main/java/memoranda/EventsMa...	getRepeatableEven...
Number of Parameters (avg/max per method)		0.705	1.005	8	/A7/src/main/java/memoranda/EventsMa...	createRepeatableEv...
Nested Block Depth (avg/max per method)		1.388	0.848	8	/A7/src/main/java/memoranda/NoteListI...	getNotesForPeriod
Afferent Coupling	30					
Efferent Coupling	16					
Instability	0.348					
Abstractness	0					
Normalized Distance	0.652					
Depth of Inheritance Tree (avg/max per type)		1.172	0.378	2	/A7/src/main/java/memoranda/Start.java	
Weighted methods per Class (avg/max per type)	485	16.724	18.011	71	/A7/src/main/java/memoranda/TaskImpl.j...	
Number of Children (avg/max per type)	0	0	0	0	/A7/src/main/java/memoranda/TaskListIm...	
Number of Overridden Methods (avg/max per type)	3	0.103	0.402	2	/A7/src/main/java/memoranda/TaskImpl.j...	
Lack of Cohesion of Methods (avg/max per type)		0.108	0.219	0.679	/A7/src/main/java/memoranda/TaskListIm...	
Number of Attributes (avg/max per type)	27	0.931	1.048	4	/A7/src/main/java/memoranda/TaskListIm...	
Number of Static Attributes (avg/max per type)	29	1	2.034	7	/A7/src/main/java/memoranda/History.java	
Number of Methods (avg/max per type)	176	6.069	7.965	37	/A7/src/main/java/memoranda/TaskImpl.j...	
Number of Static Methods (avg/max per type)	61	2.103	4.334	17	/A7/src/main/java/memoranda/EventsMa...	
Specialization Index (avg/max per type)		0.071	0.365	2	/A7/src/main/java/memoranda/Start.java	
Number of Classes	29					
Number of Interfaces	0					
Total Lines of Code	2041					
Method Lines of Code (avg/max per method)	1251	5.278	5.516	33	/A7/src/main/java/memoranda/EventsMa...	getRepeatableEven...

Efferent couplings decreased and I believe that is a good thing. I think the more external dependencies a package has, the more opportunity for things to go wrong…like machinery with moving parts.

PART 3

1. Code Smell Within a Class: Assign7\SER316-Spring-2018\src\main\java\memoranda\ TaskListImpl.java Line 341. This was an example of duplicate code. The interface requires a getTopLevelTasks method. In the class, the getTopLevelTasks only returned the private method getAllRootTasks. That method was not required and there was no benefit to having it private. I renamed getAllRoottasks to getTopLevelTasks and made it public and deleted the former getTopLevelTasks. This provided the same functionality and didnt have any ripple effects/
2. Code Smell Between Classes: IEventNotificationListener line 20, DefaultEventNotifier line 39, EventsScheduler line 62, 107, 138. This was an example SPECULATIVE GENERALITY. This looks

like something they wanted to implement in the future but the method was never actually implemented. There was just a comment in the body. So I had to remove it which caused ripples in other classes. I LEFT THE “DELETED” CODE IN AS COMMENTS SO YOU COULD SEE WHAT I WAS “DELETING”.

3.

Metric	Total	Mean	Std. Dev.	Maximum	Resource causing Maximum	Method
McCabe Cyclomatic Complexity (avg/max per method)		2.056	1.757	16	/A7/src/main/java/memoranda/EventsManager.java	getRepeatableEventsFor...
Number of Parameters (avg/max per method)		0.714	1.008	8	/A7/src/main/java/memoranda/EventsManager.java	createRepeatableEvent
Nested Block Depth (avg/max per method)		1.393	0.852	8	/A7/src/main/java/memoranda/NoteListImpl.java	getNotesForPeriod
Afferent Coupling	30					
Efferent Coupling	16					
Instability	0.348					
Abstractness	0					
Normalized Distance	0.652					
Depth of Inheritance Tree (avg/max per type)		1.172	0.378	2	/A7/src/main/java/memoranda/History.java	
Weighted methods per Class (avg/max per type)	481	16.586	17.974	71	/A7/src/main/java/memoranda/TaskImpl.java	
Number of Children (avg/max per type)	0	0	0	0	/A7/src/main/java/memoranda/TaskListImpl.java	
Number of Overridden Methods (avg/max per type)	3	0.103	0.402	2	/A7/src/main/java/memoranda/TaskImpl.java	
Lack of Cohesion of Methods (avg/max per type)		0.108	0.219	0.679	/A7/src/main/java/memoranda/TaskListImpl.java	
Number of Attributes (avg/max per type)	27	0.931	1.048	4	/A7/src/main/java/memoranda/TaskListImpl.java	
Number of Static Attributes (avg/max per type)	29	1	2.034	7	/A7/src/main/java/memoranda/History.java	
Number of Methods (avg/max per type)	174	6	7.913	37	/A7/src/main/java/memoranda/TaskImpl.java	
Number of Static Methods (avg/max per type)	60	2.069	4.283	17	/A7/src/main/java/memoranda/EventsManager.java	
Specialization Index (avg/max per type)		0.071	0.365	2	/A7/src/main/java/memoranda/Start.java	
Number of Classes	29					
Number of Interfaces	0					
Total Lines of Code	2030					
Method Lines of Code (avg/max per method)	1235	5.278	5.506	33	/A7/src/main/java/memoranda/EventsManager.java	getRepeatableEventsFor...

4. Lines of Code dropped from 2041 to 2030. This is because I deleted a method from an interface which caused a ripple effect. I was say less lines of code is an improvement.