

Deliverable 2/ Chapter 2

CSCI-02 362

Team 3: Meg, Mitch, Itzayana

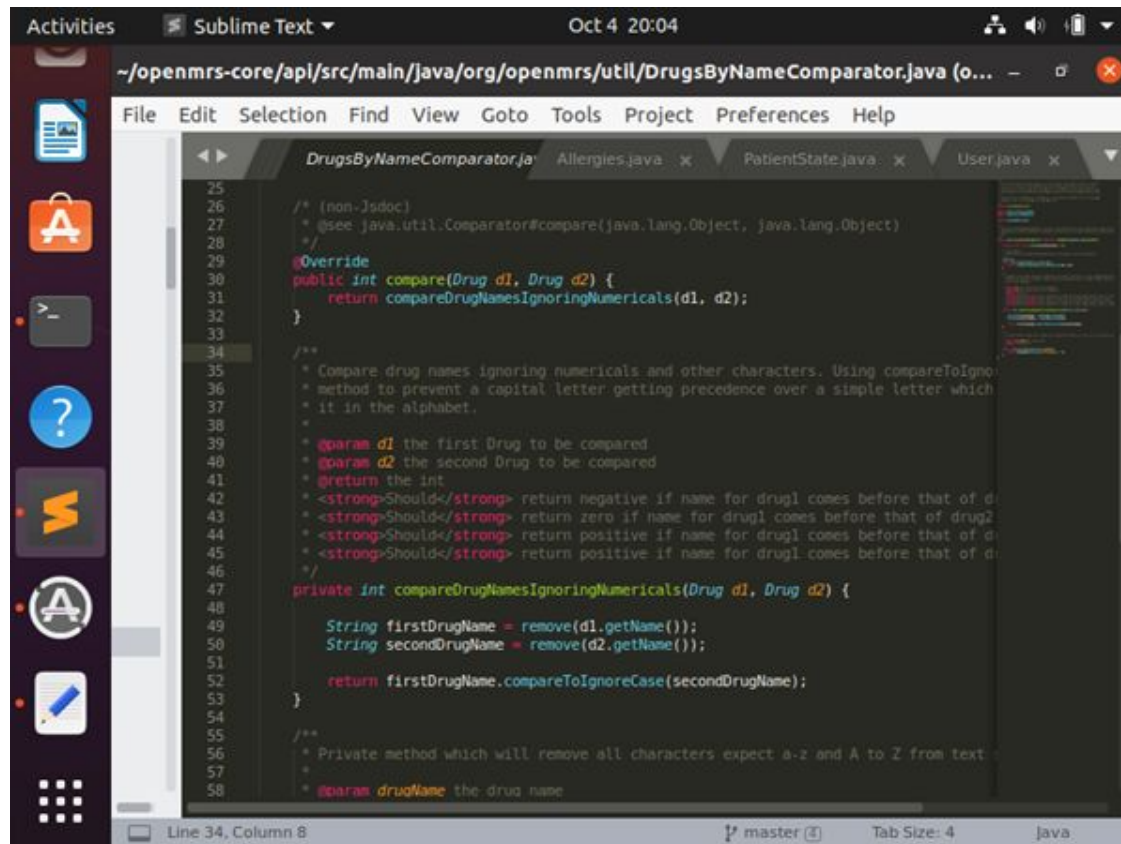
Method Selection

OpenMRS had plenty of files for our team to search through to find 5 methods to test. The sheer number of Java files was very overwhelming at first and finding methods fit for testing proved to be difficult. Since OpenMRS is more or less a Database that stores medical information it is mostly made up of getter and setter methods, but as we continued to parse through the different classes, we eventually found several methods that we considered fit for testing. As of now we are looking at more than the required amount of methods since a lot of these classes have several dependencies, and we are not sure what will be manageable and what will not be. For this Deliverable we will be looking at the `compare()` method from the `DrugsByNameComparator` class.

Method

The `compare()` method in `DrugsByNameComparator` takes two objects of type `Drug` and compares their names by using the `compareTo` interface. An image of the code is below. This method should take two objects of type `Drug`, get each one's name, remove any numbers from each name, and then compare them alphabetically. If the first drug should come before the second, then the method should return a positive integer. If the second drug should come first, then it should return a negative integer. If both drugs have the same name, then it should return a zero. Since this method is more or less just comparing two strings, our team may tweak the code so that `compare()` takes two strings instead of two `Drug` objects for ease of

testing and to avoid any unnecessary dependencies.



```
25
26  /* (non-Javadoc)
27   * @see java.util.Comparator#compare(java.lang.Object, java.lang.Object)
28   */
29  @Override
30  public int compare(Drug d1, Drug d2) {
31      return compareDrugNamesIgnoringNumericals(d1, d2);
32  }
33
34  /**
35   * Compare drug names ignoring numerals and other characters. Using compareToIgnore
36   * method to prevent a capital letter getting precedence over a simple letter which
37   * it in the alphabet.
38   *
39   * @param d1 the first Drug to be compared
40   * @param d2 the second Drug to be compared
41   * @return the int
42   * <strong>Should</strong> return negative if name for drug1 comes before that of d
43   * <strong>Should</strong> return zero if name for drug1 comes before that of drug2
44   * <strong>Should</strong> return positive if name for drug1 comes before that of d
45   * <strong>Should</strong> return positive if name for drug1 comes before that of d
46   */
47  private int compareDrugNamesIgnoringNumericals(Drug d1, Drug d2) {
48
49      String firstDrugName = remove(d1.getName());
50      String secondDrugName = remove(d2.getName());
51
52      return firstDrugName.compareToIgnoreCase(secondDrugName);
53  }
54
55  /**
56   * Private method which will remove all characters except a-z and A to Z from text
57   *
58   * @param drugName the drug name
```

Testing

- **Testing Process**

- We will create an automated testing script that will run through the 25 test cases our group has created for OpenMRS, parse the results, and save them in an HTML file.

- **Items Tested**

- As stated before, we have selected more methods than needed to ensure we can find a method that doesn't have too many dependencies or that we can tweak enough to keep the logic the same but make it easier to test. Below are the classes and methods we plan to test.

Class	Method
Allergies	Add(Allergy a1)
DateUtil	truncateToSeconds(Date date)
DrugByNameComparator	compare(Drug d1, Drug d2)
DrugOrder	hasSameOrderableAs(DrugOrder d1)
Graph	topographicalSort()
OrderUtil	isType(OrderType o1, OrderType o2)
PateintState	Copy(PatientState p)/compareTo(PatientState p)
PersonMergeLogData	computeHashTable()
UserByNameComparator	compare(User u1, User u1)

- **Testing Schedule**

- Deliverable 3: Automated Testing Framework. (11/05/20)
- Deliverable 4: Run 25 Test Cases and Pipe them to HTML File (11/17/20)
- Deliverable 5: Test deliberate errors in the Code (11/24/20)
- Final Report and Presentation (12/03/20)

- **Test recording procedures**

- All output will be sent to an HTML file and displayed in a web browser.

- **Hardware and software requirements**

- Hardware: Virtual Machine/Computer
- Software: Terminal, Java, Github, and Git

- **Constraints**

- Time, Meetings, Hardware/Software Constraints

- **System tests**

- For testing our methods, we have not yet created an automated testing script, but we have a few preliminary test cases in our Github Repository¹. The basic information in them can be found in the graph below.

TestCase Id	Input 1	Input 2	Expected Output
01	Allegra	Dayquil	1
02	Dayquil	Allegra	-1
03	Allegra	Allegra	0
04	2Allegra	1Dayquil	1
05	Allegra	NULL	Error

Final Thoughts and Evaluations so far

We have been learning a lot as a team finding good testable methods has proven difficult since most of the methods in OpenMRS are setter and getter methods. We've also did a team exercise to help learn how to compile a java program from the command line. We are still running into some error when we try to run the OpenMRS, namely with packages and other strange dependencies, Hopefully, we can overcome that and continue with our testing.

¹ <https://github.com/csci-362-02-2020/Team3/tree/master/testCases>