Tutorial 5: Debugging

CSE1100 - Introduction to Programming

1 Checkstyle

You are given a `HomeApplication`. This software is supposed to manage a smart home application, but for now, it will just print when a connected appliance is supposed to do something. Although the application is functional, not much care was put into code quality. Checkstyle will complain quite a bit. Use the provided `checkstyle.xml` file to find the problems and fix them.

1.1 Installing Checkstyle

If you have not installed Checkstyle before, you can do so by going to 'Settings →Plugins' and installing the 'CheckStyle-IDEA' plugin. The provided checkstyle file should be configured automatically, but if it is not you can configure it as follows:

- Go to 'Settings →Tools →Checkstyle'
- Click the '+' button under 'Configuration File'
- Add a name and point to the local 'checkstyle.xml' file
- Check the 'Active' checkbox for the new configuration
- Click 'Apply' and 'OK'

2 Simple Debugging

You are given a programme that calculates the primes from 0 to 100 and prints them. The `printPrimes` function contains no bugs. The `calculatePrimes` function however contains some bugs. Use the debugger to figure out what is wrong and fix the method. Feel free to rewrite some parts to make it more clear what is happening.

3 Container Class

You are given a `FruitApplication`. There is a `FruitBasket` that contains `Fruits`. `Fruits` have a name and a price. A `FruitBasket` has an additional discount you get when you buy the entire basket. The application adds some `Fruits` to a `FruitBasket` and prints the calculated price. Try to run the application.

3.1 :(

You will note that a `NullPointerException` is thrown. Use the debugger to figure out why and fix the problem.

3.2 Discount

Now that the exception is fixed, we can run the application. It seems that the discount is not applied to the basket however. Use the debugger to figure out why and fix the problem.

4 Complex Debugging

You are given a `UniversityApplication`. A `Year` contains `Quarters`, which in turn contain `Courses`. It reads a file located in `resources` called `curriculum.txt`. The file is structured as follows. Every `Year` starts with a line containing only *Year n.* Where *n* is the number of the `Year`, but this will be ignored by the reader. Then there are 4 lines of `Quarters`. The `Courses` in a `Quarter` are separated by semicolons. Every `Course` is a comma separated string consisting of 3 parts: the `name`, `abbreviation` and `code`.

4.1 It's broken

The reading however, does not seem to work. Use the debugger to figure out what is going wrong and fix it. You can assume the file is found and the contents of the file are correct.

4.2 Testing

Uncomment part two of the main method. In part two of the main method, the names of all mathematics `Courses` are printed. (A mathematics `Course` is a `Course` that has a 2 as the second character of the numeric part of its `code`, e.g. *CSE1205.*) As you might expect, this also contains bugs. Write tests for `getAllMathematicsCourseNames` in `UniversityApplication`, for `getAllCourses` is `Year`, and for `isMathematics` in `Course`. Try to use your tests to find where the problems lie and fix them.