Labs

TAN, Shin Hwei

陈馨慧

Southern University of Science and Technology

No class for Week 8

- No lecture will be held on Week 8 due to the 清明holiday.
- There will also be no lab for the labs on Monday (6 March)
- Other lab sessions (not on 6 March) will use these slides for revision and reminder for previous lab assignments, and project.

Previous Lab Assignments

- Below are lab assignments that have been posted in the past few weeks:
- Coverage Lab:
- https://classroom.github.com/a/S8hiiL1J
- Junit Lab (Pair Programming) If you couldn't find a partner, you can complete this alone:
- https://classroom.github.com/g/trrjQPYs
- Metrics lab:
- https://classroom.github.com/a/qMOJKfSm

 The lab assignments should be completed after each lab. The deadline for all lab assignments is the last week of class.

Assignments/Project

- Below are assignments/project:
- MP1 Part 2: due on April 14 (require a few hours of installation and running the tools for 24 hours)
- https://classroom.github.com/a/RY-J75Dp
- Progress report due on April 23
- https://classroom.github.com/g/LXkbUCAc

 The lab assignments should be completed after each lab. The deadline for all lab assignments is the last week of class.

Junit Lab: Simple TriType Example

- TriTyp: Given three integers for the lengths of the sides of a triangle, find the type of triangle
- Try testing the "Triang(int, int,int)" method
- Go to this link to get the program:

https://classroom.github.com/g/zkq_fzQ6

This is an graded (passed/failed) lab assignment. You get full score as long as you accept the invitation link and commit your test ("TriTypTest.java" file) at the last step.

Junit Lab: Pair Testing

1. Form a team of two with your neighbors



2. Write two JUnit tests for TriTyp individually



3. Write two JUnit tests for TriTyp to check for exceptions individually



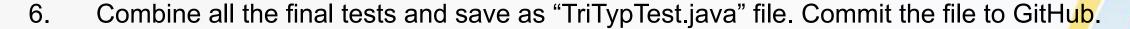
4. Compile & run your tests & theory to see if they are passing



5. Show all your tests to your teammate and try the following:



- a) Convince your teammate that your tests are better
- b) Convince your teammate that your tests are testing different behavior than his/her tests
- c) Convince your teammate that your tests are of the same quality
- d) Let your teammate knows if you feel that his/her test is better



Junit Lab: How many tests you have now?

- 2 JUnit Tests?
- 3 JUnit Tests?
- 4 JUnit Tests?
- > 4 JUnit Tests?

Coverage Lab: Generate Test

- Generate test using Evosuite for the MessageBuilder class
 - Use the command:
 - mvn -DmemoryInMB=2000 -Dcores=2 evosuite:generate evosuite:export test
 - Or use the "Generate Test" interface

Coverage Lab: Analyze each generated test

- For each test, answer the following questions:
 - Which line will this test cover?
 - Rename each test to the line where each test cover.
 - For example, if the test cover line 1, 2 and 3. Then change its name to "testL1a2a3"
- Could Evosuite achieve 100% line coverage?
- For each uncovered block of code, explain why the tool cannot cover.
- Write more tests to increase the code coverage.
- Put the answers in README.md. Put the JUnit test in TestMessageBuilder.java and commit all the files.

Metrics Lab Exercise Part 1

- What is the Cyclomatic complexity for the class according to the plugin for the class below? Explain how to calculate the cyclomatic complexity (e.g., how many branches it have)
 - 1. SwitchExample.java
 - Example1.java
 - 3. Example2.java
- Add a README.md with the following information:
 - Name:
 - Student id:
 - Screenshot showing the results of the plugin

Metrics Lab Exercise Part 2

- Answer the following questions by modifying the README.md that you have created in Lab Exercise part 1
 - Part 2: Metrics for my project
 - What is the Lines of Codes of your selected project? (If you selected two projects, you just need to select one of the projects to answer this question)
 - What is the maximum Cyclomatic Complexity of the classes of your project?
 - Do your project has any method with cyclomatic complexity >10? If yes, explain why the cyclomatic complexity is high for your project.
- *You don't need to include any screenshot for this part *You don't need to upload your project to the invitation link. Only need to answer questions in README.md

Project

Follow the plan for your project

- Start writing tests to reproduce the bugs in the selected issues
- Implement the selected issues according to plan
- Don't forget to write documents (Javadoc comments) and tests!

Link to issues	Type of issues (Bug/Feature)	Estimated Time to fix each issue	Number of people for fixing this issue (≤2 members for each issue)	Estimated Difficulty (in a scale of 1-5, 1 means very easy to fix and 5 means very difficult to fix)
https://github .com/INRIA/s poon/issues/3 118	Feature	3 weeks	1	5

Total: 50 weeks