EECS2030 Fall 2019
Guide to Programming Lab Test 1
When: Your Registered Lab Session on **Tuesday, October 8**

Jackie Wang

1 Format

- This in-lab programming test accounts for **20%** of your course grade.
  - An on-screen instruction sheet (in HTML or PDF) will be available in the beginning of the test.
  - You are required to develop Java programs (**with no syntax or type errors**) in Eclipse.
  - By the end of the test, you are required to submit your Java files via a terminal (like how you did in Lab0).
- This programming test is meant for **assessing your ability to write valid object-oriented programs that are executable**.
  - When a collection of objects need to be stored, you are **only allowed** to use a primitive array (e.g., `Point[]`, `Account[]`). Use of any Java collection class (e.g., `ArrayList`, `LinkedList`, `HashMap`) is **strictly forbidden**.

    Rationale: We expect you to develop and demonstrate the ability to build software from scratch, rather than relying on pre-implemented methods supplied by Java library.
  - Your marks will be determined by:

    1) If your submitted Java classes and the given JUnit test class altogether compile; and

    2) If compilation succeeds, the number of JUnit tests that your code passes.
  - If **1)** is satisfied, then:
    * You already receive 20 marks (out of 100) of the test.
    * To determine the remaining 80% of your marks, we will run a number of test cases on your submitted class.

      For example, say we run 10 test cases (and say they happen to have equal weights) on your submitted code, and your submitted code complies and passes 6 of them, then your final marks are: $20 + 80 \times \frac{6}{10} = 68$.
  - If **1)** is **<u>not</u>** satisfied, then:
    * A TA will spend a short amount of time attempting to fix simple errors in your code (e.g., missing semicolon, mismatching brackets), or to remove parts of your code, so that your code compiles.
    * Then we run the fixed program against the set of grading tests.
    * A **50% penalty** will be taken on the result.
    * For example, say the fixed program receives 68 marks, then you would receive $68 \times 50\% = 34$.

    This requirement will be imposed strictly, so your best preparation strategy is to familiarize yourself with all syntax and concepts of Java that's so far covered in the lectures and complete the practice exercises.

2 Rationale for the Grading Standard

The two most important learning outcome of this course are:

1. Computational and Object-Oriented thinking (for which you build through lab exercises and assessed by written tests and exam)
2. Being able to write *runnable* programs (for which you are assessed through programming tests)

When you write an essay, if there are grammatical mistakes, it can still be interpreted by a human. Computer programs are unlike essays: when your program contains compile-time syntax or type errors, it just cannot be run, end of story. When a computer program cannot be run, its runtime behaviour is simply unknown; and this is particularly the case when your program contains if-statements and loops.

When you land a job upon graduation, you would not expect your supervisor or colleagues to read your code that does not run, because it does not even compile, would you? True, this is only your second Java course, and you're still learning. But it is exactly this mind set that restricts your potential of becoming a competent programmer. If we want to train you to be a competent programmer, NOW is the time to enforce the strict (but justifiable) standard. Nonetheless, a TA would still attempt to fix your code even if it does not compile, so that if you only make a few minor mistakes, it would not cause you to fail. Take proper action now (e.g., more time spent on practicing to program) to strengthen your programming skills!

3 Rules

- You must show up for your registered session only.
- There will be a **seating plan**: you must sit at your assigned seat.
- Bring a piece of photo ID (i.e., YU card or driver license).
- No mobile phone usage is allowed during the test.
- No data sheet will be allowed.
- You may bring pen/pencil and a piece of blank paper for sketching your solutions.

4 Coverage for the Programming Test

The main focus is for you to write valid Java programs. But you may still want to review:

- Slides
  - Classes and Objects
  - Exceptions
  - Test-Driven Development (TDD) with JUnit
  - The `equals` method
- Reading on the programming pattern: Point and PointCollector
- **Optional**: If you still struggle with the basics of Java (such as if statements, nested loops, logical operators, classes and objects, *etc.*), which this course assumes, refer to parts of the tutorial series:

  `https://www.youtube.com/playlist?list=PL5dxAmCmjv_5NRNPG3OiWZWAqmvCjiLfG`

You need **not** worry about the weekly labs for this test.

5 Two Importance Practice Sets to Complete

- First, familiarize yourself with the expectation and strategy for a programming lab test by following the tutorial here:

  `https://youtu.be/_B-bNZSul3o`

  You can find the starter project of this tutorial here:

  `https://www.eecs.yorku.ca/~jackie/teaching/lectures/2019/F/EECS2030/exercises/ExampleLabTest.zip`

- Once you are clear about the expectation and strategy, attempt the practice exercise here:

  `http://www.eecs.yorku.ca/~jackie/teaching/lectures/2019/F/EECS2030/exercises/EECS2030_F19_Programming_Labtest_1_Practice_Questions.pdf`

  The level of difficulty of the test will be similar to this practice exercise.