Assignment 1: Test Suite Analysis and Improvement

Course: EECS 4313 - Software Engineering Testing
12.5 marks
Due date: Mon OCT 24, 5pm

Objective

Evaluate and improve unit test coverage of a set of standalone simple Java classes, provided in JavaAlgs project in GitHub, using coverage analysis, LLM-generated tests, and manual refinement.

Instructions

1. **Clone the repository at** JavaAlgs

2. **Initial Coverage Analysis (2.5 marks)**

   - Run the existing unit tests per program and calculate the following:
     - Statement, branch, condition coverage (use tools like CodeCover, JaCoCo, Coverage.py, Istanbul). // for condition coverage you can use all conditions or MC/DC
   - Submit:
     - Pre-improvement coverage **report** (screenshots/logs).
     - A table comparing all three metrics.

3. **LLM-Generated Test Improvement (3 marks)**

   - Use an LLM (e.g., ChatGPT, Gemini, DeepSeek) to generate additional tests targeting low-coverage areas.
   - Ensure:
     - Generated tests are runnable and relevant (no irrelevant/bloated tests).
     - For each program the coverage should improve by ≥10% in at least one metric.
   - Submit:
     - LLM-generated test code (highlight new ones/changes). Include the detail about which LLM(s) you used, what was/were the prompts and if you had to make multiple runs include all.
     - Post-LLM coverage **report** (table showing before/after improvement).

4. **LLM Test Analysis (2.5 marks)**

   - Critically evaluate the LLM's output:
     - Good Practices: E.g., meaningful assertions, edge-case coverage.

    - Bad Practices: E.g., redundant tests, incorrect mocks.
    - Include ≥3 strengths and ≥3 weaknesses with concrete examples across all programs.
- For any weak or poorly generated test, fix the test manually
- Submit:
    - An analysis report.
    - The fixed test cases.

5. Manual Test Improvement (2.5 marks)

- Manually add tests to increase each of the 3 coverage metrics in at least 2 programs
- Manually add more tests to achieve 100% statement and decision coverage in at least 2 programs that were not 100% before
- Submit:
    - Manual test code (annotate improvements).
    - Post-manual coverage report (table showing the three states of improvement).

6. Final Reflection (2 marks)

- Reflect on:
    - Challenges in selecting/running tests.
    - LLM limitations vs. human intuition.
    - What LLMs are missing? + ideas on how to improve LLMs in TestGen.
    - Insights into test adequacy and quality.
- Submit:
    - Reflection as part of the final report.

Grading Scheme

Criteria	Marks	Requirements
Initial Coverage	2.5	Metrics reported.
LLM Test Improvement	3	Tests runnable, ≥10% coverage boost.
LLM Test Analysis	2.5	≥3 str./wk., Fixed properly, depth of examples.
Manual Test Improvement	2.5	≥2 units improved per metrics, ≥2 at 100%
Final Reflection	2	Clear reflection, actionable insights.
Penalties	**up to -5**	**Execution errors on tests, tool misuse**
	-up to 5	**Poorly written report**

Submission

One document containing the following report/contents:

- Coverage reports (initial, post-LLM, post-manual) with the tables on the metrics.
- Test code (LLM and manual sections marked clearly).
- LLM generated test analysis and final reflection.

Questions? Ask early—don't wait until the deadline!