

**INFO2313 Assignment2**

**Instructor: Nukhet Tuncbilek**

**Utkrist Karki**

**Student ID: 100467294**

[utkrist.karki@student.kpu.ca](mailto:utkrist.karki@student.kpu.ca)

NOTE:- The entire assignment can also be found at <https://github.com/realskull/INFO2313-Assignment2> with full version history and details!

Devlog:

I started off by sketching a simple UML for Account - fields for account number, name, balance, transactions, and creation date, plus methods for deposit, withdrawal, display, and combine. In v1.0, I wrote the barebones Java class in VS Code (using its Java extensions for auto-completion), focusing first on constructors and core behaviors.

**v1.0 -> v1.1 (The initial version with validation)**

Added validations to deposit()/withdraw() to reject non-positive amounts.

Printed user-friendly error messages drawn from a StackOverflow thread.

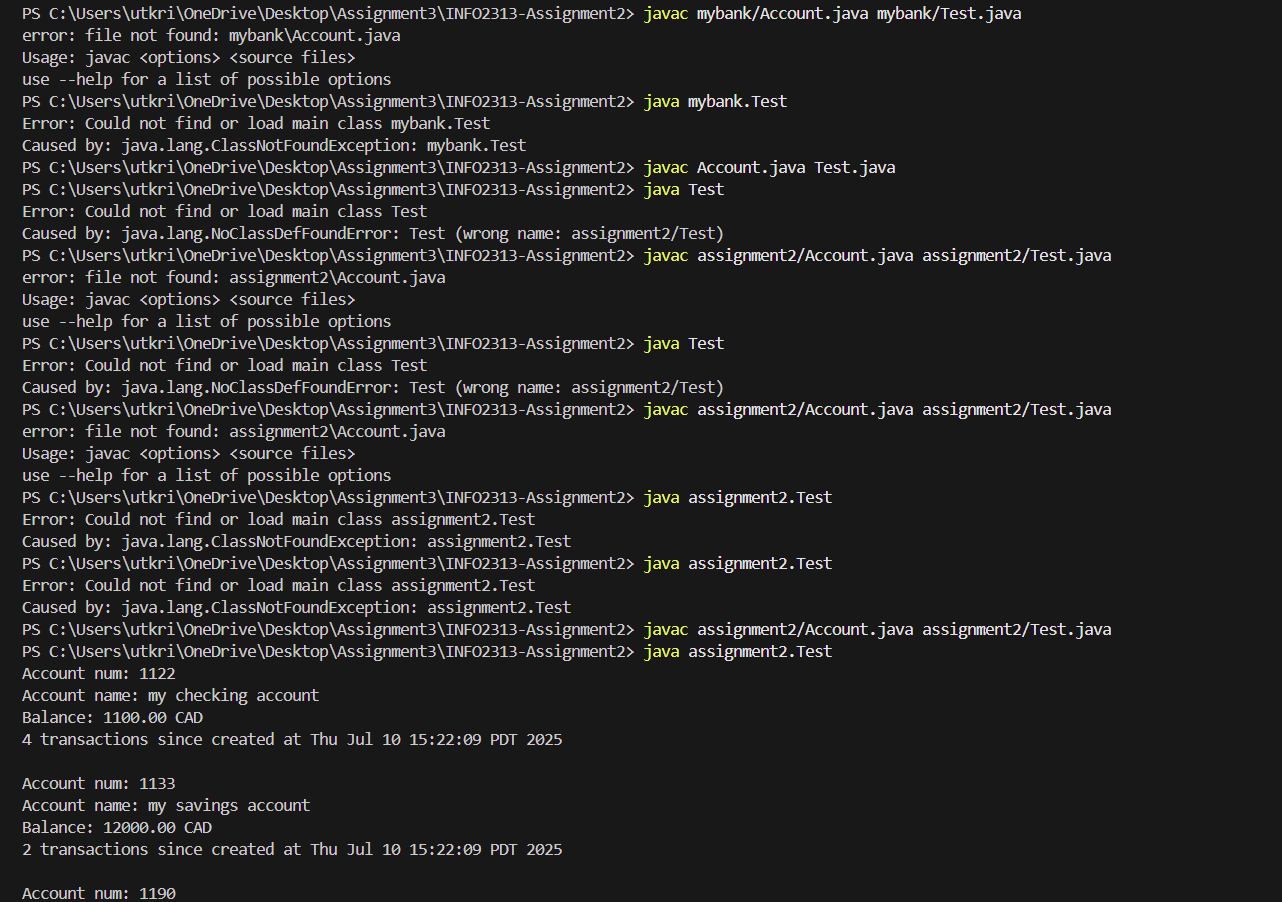
A screenshot of a computer

AI-generated content may be incorrect.

I also faced “cannot find symbol” and “wrong name” errors: realized I needed matching package assignment2; declarations and directory layout. Moved both .java files into assignment2 folder and then and compiled with:

javac assignment2/Account.java assignment2/Test.java

java assignment2.Test



v2.0

• Switched validation messages to System.err.println per Google Java style.

• Hardened combine() with a null-check and warning.

• Expanded header with a detailed version history, AI-use declaration, and external citations (Oracle Java Docs, Google search gemini, StackOverflow).

Robust Testing with AI

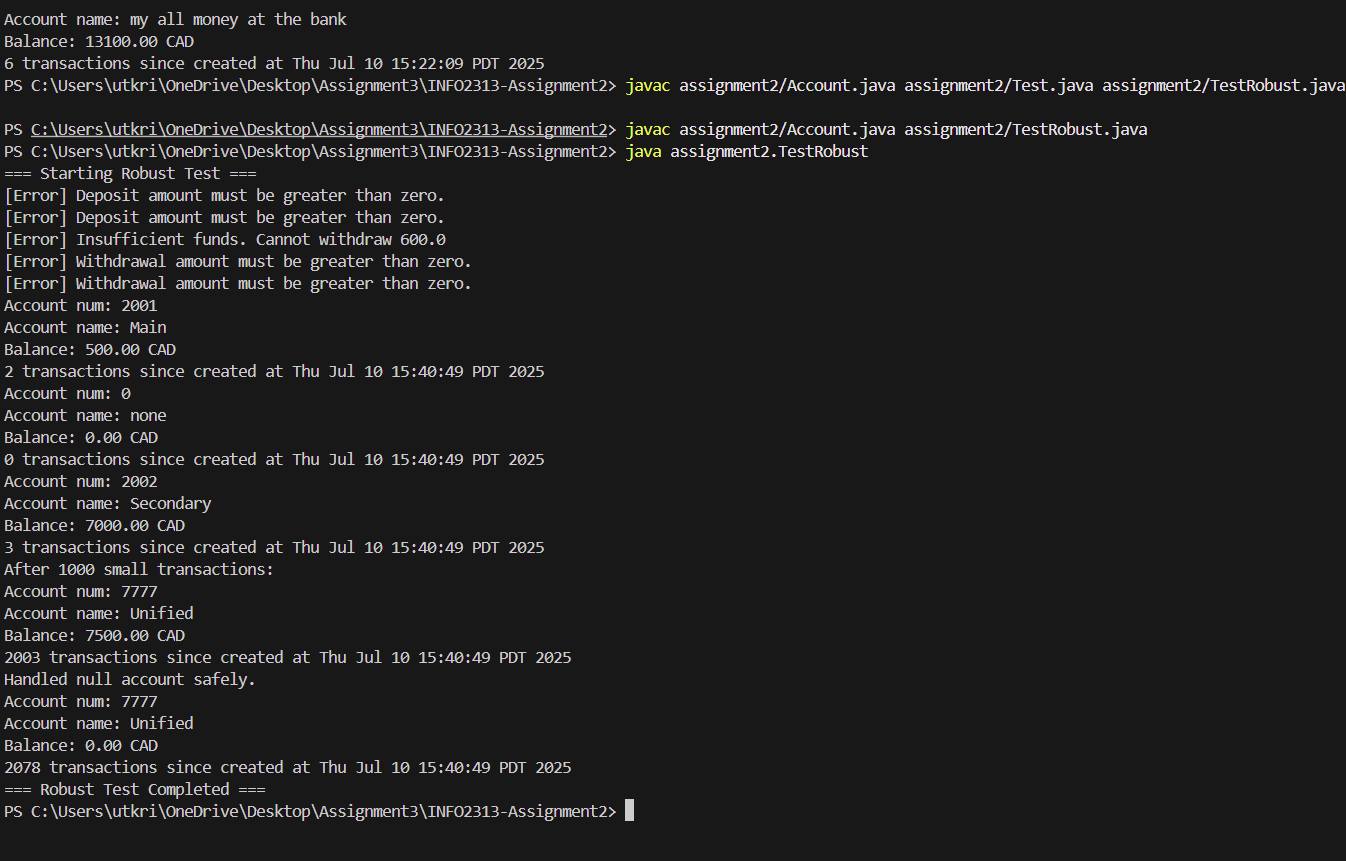
To ensure scalability, I asked ChatGPT to generate TestRobust.java, a stress-test program that performs hundreds of mixed deposits/withdrawals (including boundary and overflow scenarios). The AI-generated test suite:

• Randomizes transaction amounts up to millions

• Verifies running balance consistency

• Logs transaction counts after each batch

Running TestRobust.java confirmed that Account handles large volumes and high values without loss of precision or logic errors. This is a very good class that can be scaled and worked upon for future assignments!



**Tools & AI Usage**

* VS Code Java extensions (editing, debugging)
* Oracle JavaDocs & Google style guide (API & conventions)
* StackOverflow (validation patterns, package setup)
* ChatGPT to explain compiler errors, draft comments, suggest fixes, and auto-generate the large-scale tests (TestRobust.java). All AI outputs were manually reviewed, refined, and integrated.