TDD in 5 Languages

SYSEN-5260 Fall 2023

Objective

- You will gain hands-on experience with development tools and various different types of programming languages.
- You will learn to use community resources to teach yourself some basics in new programming languages and judge their relative strengths.
- You will gain hands-on experience doing Test-Driven Development (TDD) in a team programming environment.

Assignment

1. Write a program in **Java** that reads two matrices from csv files and writes the matrix-product to a different file.
    a. You will be provided with:
        i. Code to read the csv files into data structures and write data structures into csv files.
        ii. An empty matrix-multiply routine.
        iii. A unit- test to confirm the matrix-multiply routine produces the correct answer.
        iv. A README.md with instructions to run the program and the tests.
    b. You will:
        i. Implement the matrix-multiply routine such that the unit-test passes.

2. Use Test-Driven Development (TDD) to extend system capabilities:
    a. So far we only demonstrate that we can multiply square matrices. You will consider the following additional user-story:
        i. As a user I want to multiply an NxM matrix with an MxP matrix and get back the NxP matrix product so I can do more general math.
        ii. As a user I want to get an "Incomptable Dimensions" error when I attempt to multiply matrices with incompatible dimensions so I know why the program failed.
    b. For each user story your team will split into two sub-teams:
        i. The implementation team will:
            1. Create a branch.
            2. Implement the unit-test first. Commit the failing unit-test.
            3. Make the unit-test pass.

            4. Commit the passing unit-test.
            5. Submit a pull-request to master.
        ii. The <u>review sub-team</u> will:
            1. Review the code for correctness and readability.
            2. Leave comments; Allow the implementation team to make any required improvements.
            3. Accept the pull-request and merge to master once they're satisfied.

3. Repeat steps **1** and **2** in **C**, **Go, JavaScript,** and **Python**.

4. Prepare a written report to demonstrate your understanding:
    a. **Java**
        i. Is Java a compiled language or an interpreted language? Is Java statically typed or dynamically typed?
        ii. What is Maven? What are some of the things that we use Maven for in this project?
        iii. We use the Java keyword **new** to create new instances of our Matrix class where we specify the size of the matrix. Can we change the size of a matrix object after we create it? What do you see in the code that makes you think so.

    b. **Go**
        i. Is Go a compiled language or an interpreted language? Is Go statically typed or dynamically typed?
        ii. Go comes with a full suite of build tools that help us create and configure our modules. **What do the following commands do?**
            1. **`go run .`**
            2. **`go mod tidy`**
            3. **`go build`**
            4. **`go test`**
        iii. What does the **`:=`** operator do in Go?

    c. **JavaScript**
        i. Is JavaScript a compiled language or an interpreted language? Is JavaScript statically typed or dynamically typed?
        ii. In JavaScript is there any difference between an integer and a floating-point number?
        iii. What is npm? How did you use npm in your project?

    d. **C**
        i. Is C a compiled language or an interpreted language? Is C statically typed or dynamically typed?

ii. In **`c/matmpy.c`**, line 16 we define a structure called **`t_matrix`**. This is a data type with two attributes: **`matsize`** which is of type **`struct t_matsize`**, and **`values`** which is of type **double** **.
What does the ** mean? Explain why we'd use this type to represent a 2-dimensional matrix of numbers.

iii. When constructing this assignment I needed it to work on Windows, MacOS, and Linux hosts. One difference between these OSes is text-file line-ending markers: On the Mac and Linux we use **`\n`** to mark the end of the line. On Windows we use two characters: **`\r\n`**. This is a historical holdover from DOS, Windows' original kernel.
Docker hides most of these host-OS details from the container, but because we "mount" local folders inside the container (See lines 7 and 8 in the **`c/docker-compose.yaml`** file.) I had to deal with the possibility of either file-type.
Where in matmpy.c can you find code that deals with this line-ending difference? Can you find any other places in the c/ folder that deal with this line-ending difference?

e. **Python**

i. Is Python a compiled language or an interpreted language? Is C statically typed or dynamically typed?

ii. Unlike the Java project we didn't create a Matrix class. What class did we use to represent our Matrix in the Python project? Why is this potentially better than creating our own Matrix class?

iii. In our mat_mpy function definition, what does the -> symbol mean? Is it required for our function to operate correctly? What is the advantage of using this syntax?

f. **<u>Retrospective:</u>**

i. How did you team-members collaborate to complete this assignment?

1. How did you split up the work? Which team members worked on what?
2. How were you able to use git/GitHub to facilitate collaboration?
3. What worked well? What could be improved? What changes would you make for your next project?

ii. Rank each of the 5 language on as scale from 1 (easiest) to 10 (hardest) for each of the following aspects:

1. Expressing an algorithm.
2. Finding and fixing errors.
3. Finding and using external libraries.
4. Reading and understanding code written in this language.

5. Writing tests.
6. Controlling in detail how memory is used.

iii. Of the 5 languages you tried, which one is best suited for each of the following tasks?

1. Writing a low-latency hardware driver.
2. Prototyping a new genomic search algorithm.
3. Implementing a high-performance genomic search algorithm.
4. Building a large library for modeling complex financial instruments.
5. Building a complicated user interface.
6. Writing the embedded software that runs a robot vacuum cleaner.

Write 1-2 sentences explaining your answer for each.

iv. Rank these 5 languages from most popular to least popular based on StackOverflow Trends. Why do you think the relative popularity of these languages is so?

Technical Details

Most of the framework for each language is already implemented in this repo:

https://github.coecis.cornell.edu/dtc78/tdd5lang

Each team should create a fork of this repo then work in their fork.

Grading

This is a group assignment. Your homework group will hand in:

- Link to your GitHub repository with your completed code. This should be in a state where I can check it out and confirm your tests pass.
- Written report.

Here's how you team can get an A on this project:

- Coding:
    - The provided unit- tests pass.
    - You wrote new unit-tests to cover the new user-stories in Step-2.
    - Your new unit-tests pass.
    - Your commit-history shows that you wrote a failing test before writing the implementation.
    - Your team collaborated using git branches.
    - Your code-review history shows your team reviewed changes before merging them to master.
    - You worked together as a team and divided the effort required to complete this assignment: Everyone had the opportunity to participate in both the coding and the written report.
    - If your implementation was based on the work of others (e.g. a book, a blog-post, an online tutorial, etc.) you cite the sources and understand this implementation in enough detail to explain it and/or modify it.
    - If you implemented the calculation by calling out to an external library, you were able to integrate the library into the docker image so I can still run the test.
    - **You post questions to the class forum when you find problems in the assignment or get stuck on issues.**

- Written Report:
    - Your answers for each language are correct and complete.
    - Your retrospective answers are honest and thoughtful.

Here's how your team gets extra credit on this project:

- You respond to your classmates with helpful answers.

References

- C:
  - The C Book
- Go:
  - Go Learn Site
  - Effective Go
  - Go by Example
  - Learning Go
  - Go Book List
- Java
  - Oracle: Java Tutorials
  - Oracle: API Reference
- JavaScript
  - Mozilla JavaScript Guide
- Python
  - Office Python3 Documentation