TDD

What is TDD?

            Test-driven development is a technique where the developer first writes tests for new functionality, before writing the actual code. The first test would be the bare minimum of function, and it would fail, then you would write the code to make it pass. Then repeat with the next test that would fail, then write the code that would make that pass, and of course make sure the previous test is passing too.

What are the advantages and disadvantages of TDD?

            There are many advantages of TDD:

                        Improved quality, debugging is addressed quickly in the test, the code is usable and testable from the first test.

            There could be some potential disadvantages:

                        Time consuming, although in the long run will save time with the quality of code and reducing time spend debugging. There could be a learning curve to write the tests. There could be tests that were not thought of and left out.

Why was TDD created?

            TDD was created to reduce the bugs of the code, improve the software design, increase the productivity by catching and handling errors right away, and make the code more maintainable and easier to understand. It prevents bugs and issues in the testing and implementation phase because it is handled right away.

\*\*Side note about my TDD experience\*\*Last Friday (6-9), I did a 3 hour pair programming interview, where it was Test-Driven Development. And I had written this post before that. This was with an IT consulting firm, and it was such a great experience.

The interviewer gave one step at a time, and I had to create a test and solution for each step. We used the Apache Maven Wrapper for testing. We had the Solution class and SolutionTest class. The goal was to create a 4+ function calculator that takes a string as input and returns a string as output with the answer.

The test writing was probably the hardest thing to get the hang of. I was thinking of TDD as you have a user story and then you make the code to satisfy it. But you literally make a test with the code, and you test it every single time. It seems tedious but they were so used to it and it was super quick for them. It was such a great learning experience for me, and it really gave me motivation and something to look forward to, even if I don’t get the position. I uploaded the Solution and the SolutionTest code, and the requirements to GitHub if anyone is interested in looking at them. <https://github.com/AmandaSherman/DevOps/blob/main/TDD%20interview%20/SolutionTest.java>

<https://github.com/AmandaSherman/DevOps/blob/main/TDD%20interview%20/Solution.java>

https://github.com/AmandaSherman/DevOps/blob/main/TDD%20interview%20/Requirements.md