Kameron Mace

CS 222

8/27/24

Reading and Reflection

There are three approaches to TDD or test-driven development, the first approach involves first, writing a test that outlines what the code is supposed to do, this should start out simple and get more detailed as you go. Once the first test is decided, the programmer should work to make code that will compile and pass the test. After that passes, the programmer should add another little test and fail it. After failing that test, make a change to then pass the test. Then refactor and make the code clean by removing duplication and other errors. After that, repeat the cycle. This approach is mainly for making very clean code, not for coding day-to-day as it takes longer than other approaches to implement. It is most useful for if you get stuck in your code and need to break it down.

The second approach prioritizes getting the code to work before making it clean. To use this approach, simply type in what the right implementation could be, run it and hope that the tests work. If the tests do not end up working, in this approach, the programmer should “Fake it,” return a constant and gradually replace these with variables until the real code is there. This approach also recommends obvious implementation or typing in the real implementation.

The third approach involves translating a design objectification into a test case that fails because of the objection, then compiling it with a stub implementation, and finally making the test work by typing in what seems to be the right code. This approach works against aliasing because value objects are used. Triangulation is mainly used when the programmer is unsure of how they should refactor.

In the Leap Year Project, I mainly used a combination of the first approach and the second approach to complete my code. I did plenty of testing throughout the whole time I was working on it. I started mostly with the second approach until I got everything to work, then I moved to the first approach where I made the code much cleaner. `

Beck manages his tasks by writing them down on a physical piece of paper, while I do enjoy physical paper, and the enjoyment of crossing off items on a list, I find it quicker to make a list on my computer and check them off that way.