1. TDD involves writing unit tests just until the unit test fails, and then writing code for the test just until the unit code passes, and repeating this process until the code development it done.
2. In general, we agree. Having developed a full suite of tests for the code, as much as the code as possible will be tested, and creating the code incrementally allows for changes to the abstraction as you go.
3. It clarifies exactly what the code is needed to do and identifies what the step-by-step process that needs to be taken will be. However, it involves a lot of switching between writing test and production code, which adds time, and it also adds a lot of code rewriting when you move from early tests where the easiest way to pass a test is to return a value, when the later tests will require calculating that value.