Test-Driven Development (TDD) Process

(Write failing tests first, then code to make them pass)

Definition:

- It is a software development methodology where tests are written before the actual code is implemented.
- The process focuses on writing a small amount of test code first, ensuring the test fails initially (since the functionality hasn't been implemented yet), and then writing the minimum amount of code necessary to make the test pass.
- Once the test passes, the code is refactored and improved, and the cycle repeats for the next piece of functionality.
- This approach emphasizes writing automated tests to drive the design and development of the software

TDD Life Cycle

- Quickly add a test

  Start by writing a test for a small piece of functionality.

- Run all tests and see the new one fail

  Run the test suite to see the new test fail (this confirms that the test is valid).

- Make a simple change

  Write the minimum amount of code required to pass the test.

- Run all tests and see them all pass

  Run all tests again to ensure the new code passes all tests.

- Refactor to remove duplication

  Clean up the code while ensuring all tests still pass

<u>Principles of TDD</u>

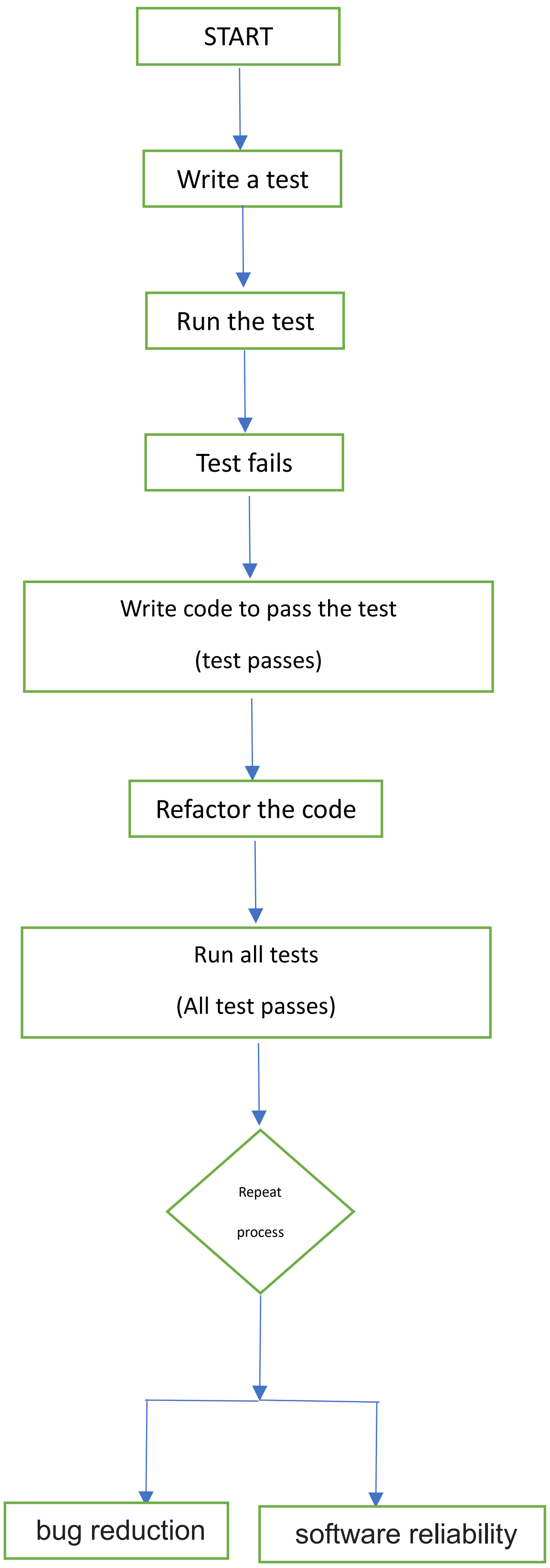
- Testing List
  - keep a record of where you want to go
  - Beck keeps two lists, one for his current coding session and one for “later” You won’t necessarily finish everything in one go!

- Test First
  - Write tests before code, because you probably won’t do it after
  - Writing test cases gets you thinking about the design of your implementation.

<u>Benefits of TDD:</u>

- **Reduced Bugs:** Tests catch errors early, preventing them from reaching later stages.
- **Improved Design:** Focus on functionality leads to cleaner and more maintainable code.
- **Increased Confidence:** Reliable tests provide a safety net for code changes.
- **Faster Development:** Less time spent debugging leads to quicker development cycles.

Infographic: Test-Driven Development (TDD)


START
Write a test
Run the test
Test fails
Write code to pass the test
(test passes)
Refactor the code
Run all tests
(All test passes)
Repeat
process
bug reduction
software reliability