Test-Driven Development (TDD) process:

Start
RED
Write a test

Clean code
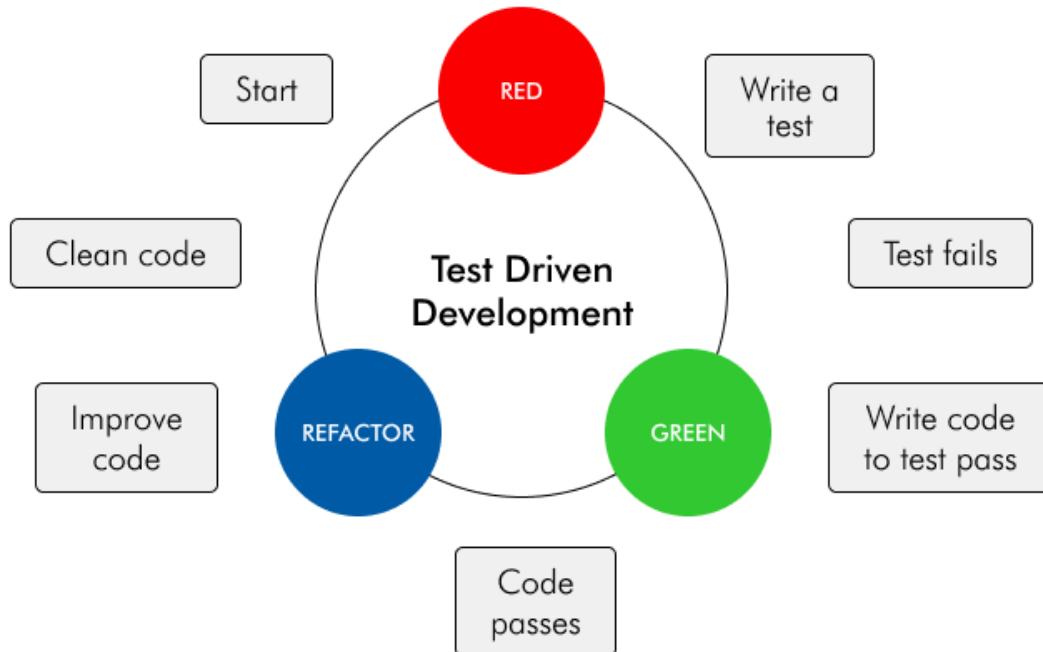Test Driven Development
Test fails

Improve code
REFACTOR
GREEN
Write code to test pass

Code passes

Step 1: Write Test Cases

- Identify the requirements and functionalities of the software.
- Write test cases that cover all possible scenarios.

Step 2: Run Test Cases

- Execute the test cases.
- Observe if any tests fail, indicating bugs or errors in the code.

Step 3: Write Code

- Develop the code that fulfills the requirements stated in the test cases.
- Keep the code simple and focused on passing the tests.

Step 4: Run Test Cases Again
- Execute the test cases to validate the code changes.
- Ensure that all tests pass, indicating that the code is functioning as expected.

Step 5: Refactor Code

- Improve the code without changing its functionality.
- Eliminate redundancies and enhance readability.

Step 6: Repeat the Process

- Continue writing tests, running them, writing code, running tests, and refactoring until all requirements are met.

<u>**Benefits of Test-Driven Development (TDD):**</u>

1. **Bug Reduction**

    - By writing tests before code, TDD helps identify and fix bugs early in the development process.
    - This reduces the number of defects in the final product.

2. **Increased Software Reliability**

    - TDD ensures that software meets the specified requirements.
    - Testing at every step helps identify and address issues, leading to more reliable software.

3. **Faster Development**

    - TDD promotes a systematic approach, resulting in faster development.
    - It reduces the time spent debugging and fixing issues later in the process.

4. **Better Code Design**

    - TDD encourages writing simple, modular code.
    - It improves code maintainability and facilitates future changes or enhancements.