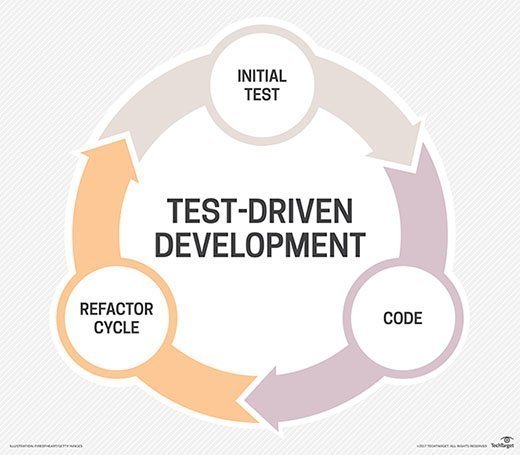
**QUESTION :** Create an infographic illustrating the Test-Driven Development (TDD) process. Highlight steps like writing tests before code, benefits such as bug reduction, and how it fosters software reliability.

**PROCESS :** Write Test → Run Tests → Write Code → Refactor → Run Tests



1. **Introduction to TDD**

Test-Driven Development is a software development approach where tests are written before the code itself. To ensure code reliability and reduce bugs early in the development cycle.

**2. TDD Cycle**

* **Red-Green-Refactor**:
  1. **Write a Test (Red)**:
     + Write a test for the new functionality.
     + The test will initially fail because the functionality isn't implemented yet.
  2. **Write Code (Green)**:
     + Write the minimum amount of code required to make the test pass.
     + The focus is on passing the test, not on writing perfect code.
  3. **Refactor**:
     + Refactor the code to improve its structure and readability.
     + Ensure all tests still pass after refactoring.

**3. Detailed Steps**

* **Step 1: Add a Test**:

Identify a small piece of functionality. Write a test that checks this functionality.

* **Step 2: Run All Tests**:

Run all existing tests. Ensure the new test fails to confirm it identifies the unimplemented functionality.

* **Step 3: Write Code**:

Write the simplest code possible to make the new test pass . Focus on getting the test to pass, not on perfecting the implementation.

* **Step 4: Run Tests Again**:

Run all tests to check if the new code passes the new and existing tests. Ensure there are no new failures introduced.

* **Step 5: Refactor**:

Refactor the code to improve its quality.

Optimize code without changing its functionality.

Run all tests after refactoring to ensure they still pass.

**4. Benefits of TDD**

* **Bug Reduction**:

Catch bugs early in the development process.

Reduce the number of defects in the final product.

* **Software Reliability**:

Ensure the code works as expected through continuous testing.

Maintain high reliability and robustness in the software.

* **Improved Design**:

Encourage writing modular and testable code.

Improve code structure and readability through refactoring.

* **Documentation**:

Tests serve as documentation for the code.

Understand what the code is supposed to do through its tests.