**Assignment 1:** 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.

**Test-Driven Development (TDD) in Action: Building a Bug-Free Future**

Imagine you're a software developer creating a program to calculate the area of a rectangle. Here's how TDD would guide you:

1. **Write a Failing Test:** Before coding, write a test that checks if the area calculation function returns the correct value for a given rectangle. Initially, this test will intentionally fail because the function doesn't exist yet.
2. **Write Minimal Code to Pass the Test:** Now, write the code for the area calculation function. Start simple, focusing only on making the failing test pass. This ensures the core functionality works.
3. **Refactor and Repeat:** Once the test passes, refactor your code to improve readability and efficiency without affecting functionality. This keeps your code clean and maintainable.
4. **New Test, New Feature:** Now, write another test! Maybe you want to test calculating the area with negative side lengths. Follow the same process: write the test, write minimal code to pass, and refactor.

**Benefits of TDD:**

* **Reduced Bugs:** By constantly testing small code chunks, you catch errors early on, preventing them from snowballing into larger issues later.
* **Improved Design:** TDD encourages you to think about the desired behavior before coding, leading to a more well-structured and maintainable codebase.
* **Increased Confidence:** With a comprehensive suite of automated tests, you can be confident that your code works as expected, even as you make changes.
* **Focus on Requirements:** Writing tests first clarifies what the code needs to achieve, ensuring you're building the right features.

**Overall, TDD fosters software reliability by emphasizing continuous testing and a focus on well-defined functionalities. It's like building a house with a strong foundation, brick by tested brick!**