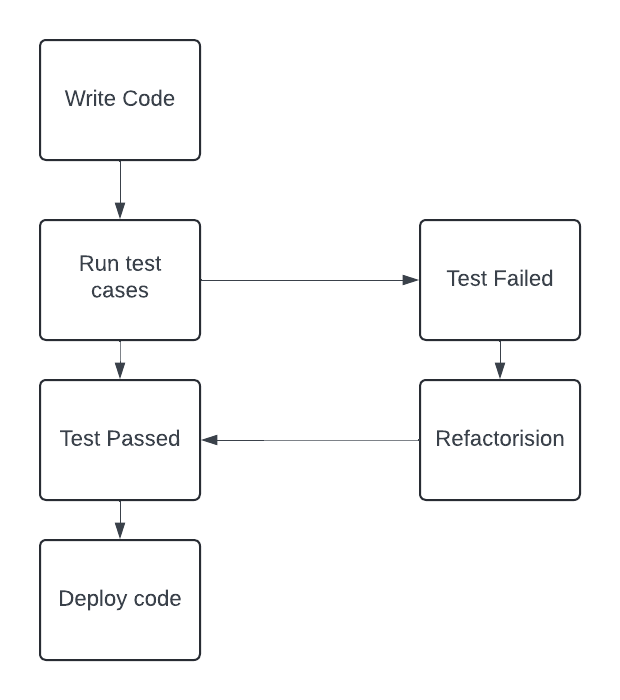
**Assignment 1: Create an infographic illustrating the Test-Driven Development (TDD) process. Highlight steps like writing tests before code, benefits such as bug reduction, and how it fosters software reliability.**

Title: Test-Driven Development (TDD) Process

* **Introduction:** Test driven development is a programming style where we test first and then development the code.
* **Infographic**

****

* **Steps of TDD:-**
* **Write the code:**

Tdd starts with write the code test cases for each function or feature.

* **Run the test:**

Computer running the test case. Initially the test cases are failed because the test cases develop before the development of the code.

* **Write the code:**

Programmer writing code to pass the test.

* **Refactore:**

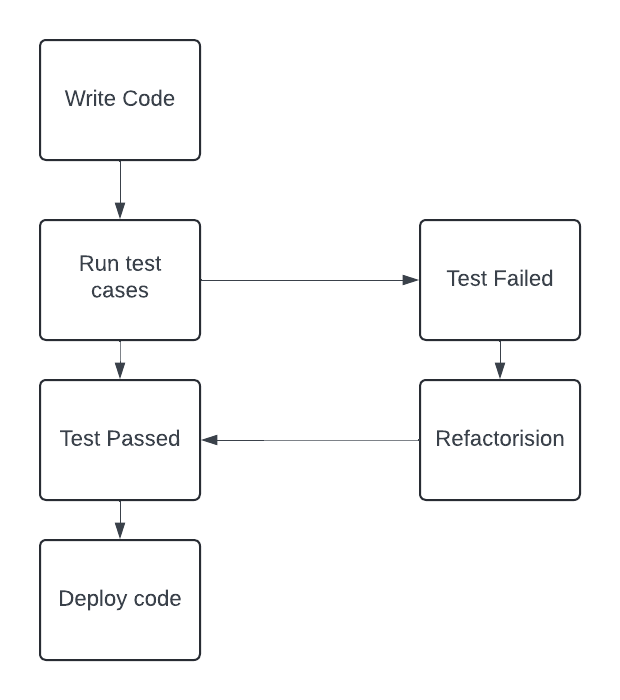
Programmer optimizing and refactoring the code. Programmer optimizing and refactoring the code.

* Benefits of TDD:
* Bug Reduction:
  + **Illustration**: Bugs being caught early in the development process.
  + **Caption**: By writing tests first, many bugs are caught early in the development cycle, reducing the cost of fixing them later.
* Improved Software Reliability:
  + **Illustration:** Reliable software depicted as a sturdy building.
  + **Caption:** TDD fosters software reliability by continuously testing and refining code, leading to robust and dependable software.
* **Disadvantages:-**
* Its come with a slow process.
* All the members of the got to do it either all the members of a team use Tdd or nobody in the group.

**Assignment 2: Produce a comparative infographic of TDD, BDD, and FDD methodologies. Illustrate their unique approaches, benefits, and suitability for different software development contexts. Use visuals to enhance understanding.**

* **TDD:-**

**Diagram:**

****

**Approach:**

1. **Test-first approach:** Write tests before writing the functional code.
2. **Refactor:** Write a failing test (red), write just enough code to pass the test (green), refactor the code.
3. **Incremental development:** Develop code incrementally with small, manageable units.

**Benefits:**

1. **Improved code quality:** Focus on writing code to meet specific test cases enhances code quality.
2. **Regression testing:** Ensures that changes don't break existing functionalities.
3. **Clear requirements:** Tests serve as documentation of expected behavior.

**Suitability:**

1. **Low-level development:** Suited for unit testing and lower-level code.
2. **Agile environments:** Fits well with agile principles of iterative development and continuous integration.

* **Bdd:**

-**Diagram**

A diagram of a process

Description automatically generated

**Approach:**

1. **Focus on behavior:** Describe the behavior of the system from the user's perspective using a domain-specific language (DSL).
2. **Collaborative approach:** Involves stakeholders, developers, and testers in defining behaviors.
3. **Test automation:** Tests are automated but focus on behavior rather than implementation details.

**Benefits:**

1. **Improved collaboration:** Stakeholders and team members understand the system's behavior better.
2. **Clear communication:** Using a common DSL helps in clearly defining requirements and expectations.
3. **Reduced rework:** Early involvement of stakeholders reduces misunderstandings and rework.

**Suitability:**

1. **User-focused development:** Ideal for scenarios where understanding user behavior is crucial.
2. **Cross-functional teams:** Suits environments where collaboration between different roles is emphasized.

* **FDD:-**

**Diagram**

A diagram of a process

Description automatically generated

**Approach:**

1. **Feature-centric:** Breaks down development into small, manageable features.
2. **Iterative development:** Features are developed iteratively and continuously integrated.
3. **Domain modeling:** Emphasizes on domain modeling to understand and develop features.

**Benefits:**

1. **Clear progress:** Features provide tangible progress indicators.
2. **Scalability:** Scales well to large projects with many features and teams.
3. **Reduced complexity:** Focus on individual features reduces complexity and improves manageability.

**Suitability:**

**Large-scale projects:** Ideal for projects with multiple features and teams.