Day-03

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.

→ Solution :

Test Driven Development is the process in which test cases are written before the code that validates those cases. It depends on repetition of a very short development cycle. Test driven Development is a technique in which automated Unit test are used to drive the design and free decoupling of dependencies.

The following sequence of steps is generally followed :

Step 1: Write a Test

- Visual representation of a developer writing a test code.
- Caption: "Start by writing a failing test that describes the desired behavior of a feature."

Step 2: Run the Test

- Visual representation of running the failing test.
- Caption: "Run the test and watch it fail, confirming that the feature doesn't exist yet."

Step 3: Write Code

- Visual representation of a developer writing code to make the failing test pass.
- Caption: "Write the minimum amount of code necessary to make the failing test pass."

Step 4: Run All Tests

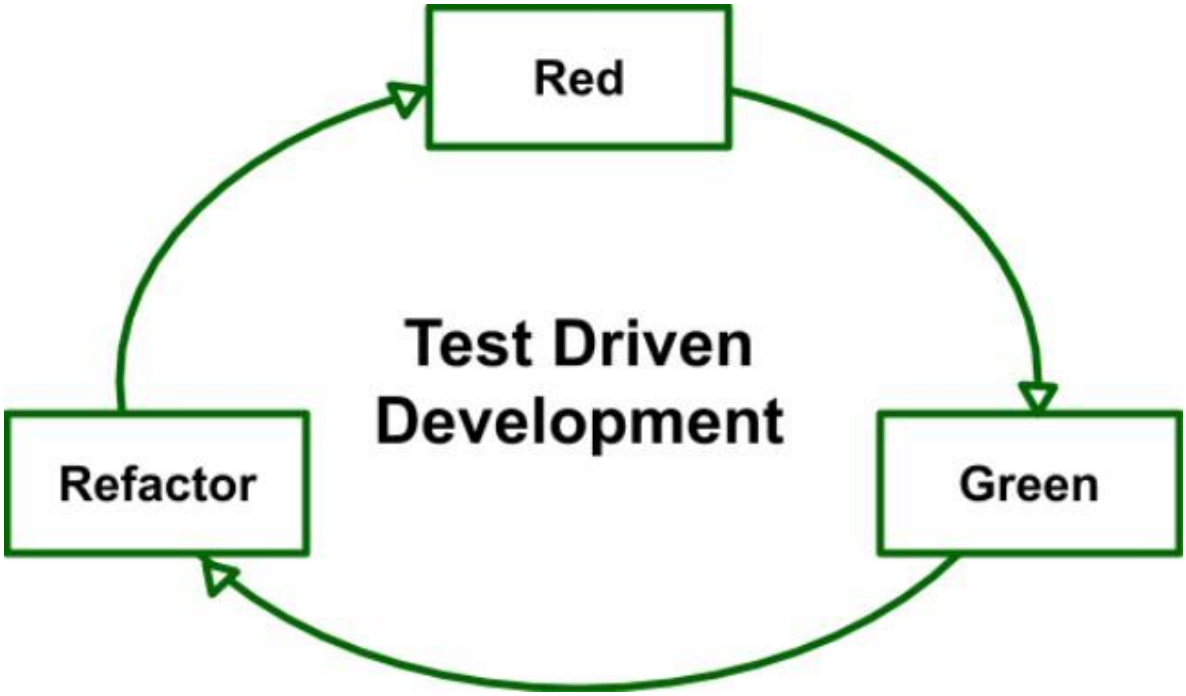
- Visual representation of running all tests after writing code.
- Caption: "Run all tests to ensure the newly written code didn't break existing functionality."

Step 5: Refactor Code

- Visual representation of a developer refactoring the code for readability and efficiency.
- Caption: "Refactor code to improve design, readability, and performance without changing behavior."

Motto of TDD:

Day-03



Red – Create a test case and make it fail.

Green – Make the test case pass by any means.

Refactor – Change the code to remove duplicate/redundancy.

Benefits:

- Unit test provides constant feedback about the functions.
- Quality of design increases which further helps in proper maintenance.
- Test driven development act as a safety net against the bugs.
- TDD ensures that your application actually meets requirements defined for it.
- TDD have very short development lifecycle.

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.

→**Solution :**

(i) TDD (Test-Driven Development):

Approach:

- Visual representation of a developer writing a failing test.
- Caption: "Write failing tests before writing code to drive development."

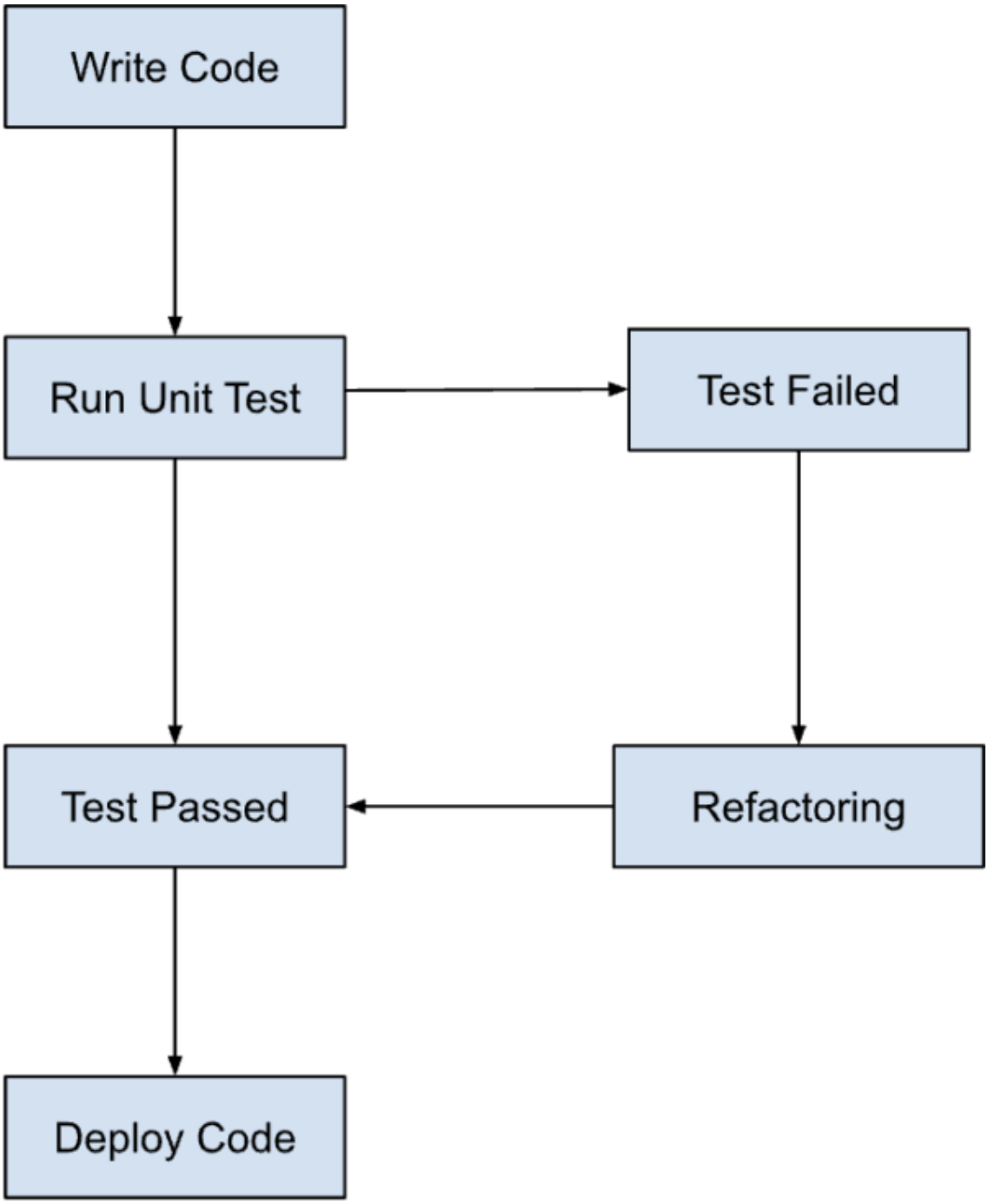
Benefits:

- Visual representation of a scale tipping in favor of TDD.
- Bulleted list of benefits, including:
- Early bug detection.
- Improved code quality.
- Enhanced software reliability.
- Faster development cycles.

Suitability:

- Visual representation of TDD fitting into a puzzle piece labeled "Iterative Development."
- Caption: "Best suited for iterative development processes and projects requiring frequent changes."

Refactoring refers to modifying the code without changing its main functionality or behavior.



(ii) BDD (Behavior-Driven Development):

Approach:

- Visual representation of a team discussing user stories and behaviors.
- Caption: "Define behavior through scenarios and acceptance criteria before writing code."

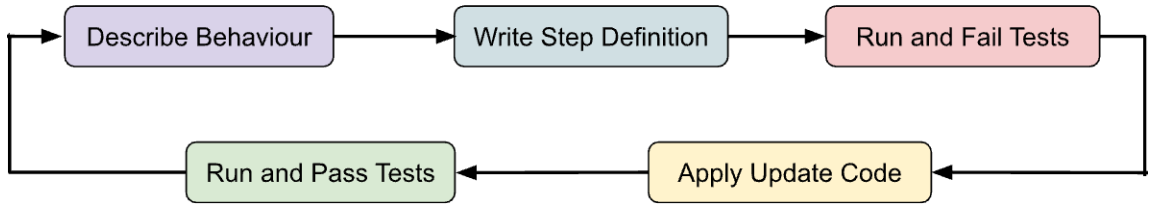
Benefits:

- Visual representation of a scale tipping in favor of BDD.
- Bulleted list of benefits, including:
- Improved collaboration between stakeholders.
- Focus on business requirements.
- Enhanced clarity and understanding of features.
- Better alignment between development and business goals.

Suitability:

- Visual representation of BDD fitting into a puzzle piece labeled "Collaborative Environments."
- Caption: "Best suited for projects with complex business logic and involving multiple stakeholders."

The image below depicts a typical BDD operation:



(iii) FDD (Feature-Driven Development):

Approach:

- Visual representation of breaking down features into manageable chunks.
- Caption: "Focus on delivering features iteratively based on client priorities."

Benefits:

- Visual representation of a scale tipping in favor of FDD.
- Bulleted list of benefits, including:
- Clear focus on features and deliverables.
- Improved project visibility and progress tracking.
- Efficient resource allocation.
- Enhanced client satisfaction.

Suitability:

- Visual representation of FDD fitting into a puzzle piece labeled "Client-Centric Projects."
- Caption: "Best suited for projects with well-defined requirements and a focus on delivering features based on client needs."

Conclusion:

Visual representation of a checkmark indicating successful completion.

Caption: "Each methodology offers unique approaches and benefits, catering to different software development contexts and project requirements."