Day 3:

Assignment 1:- Q) 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.

A) Info graphic Title:

The Test-Driven Development (TDD) Process

Step 1: Write Test Cases

Write test cases based on the requirements of the feature or functionality.

Test cases define the expected behavior of the code.

Step 2: Run Tests

Run the test cases to ensure they fail initially.

This verifies that the tests are correctly assessing the functionality.

Step 3: Write Code

Write the minimum code required to pass the failing tests.

Code is written incrementally to fulfill the requirements of the tests.

Step 4: Run Tests Again

Run the tests again to ensure the newly written code passes.

This validates that the code changes did not break existing functionality.

Step 5: Refactor Code

Refactor the code to improve its structure, readability, and efficiency.

Tests ensure that refactoring does not introduce bugs.

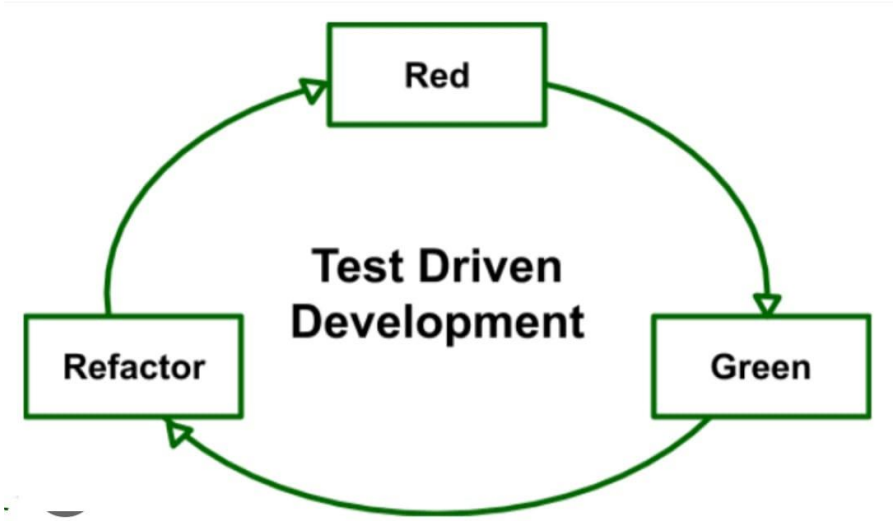
Benefits of TDD:

Bug Reduction: By writing tests before code, bugs are caught early in the development process, reducing the likelihood of bugs reaching production.

Improved Software Reliability: TDD leads to more reliable software as it ensures that code behaves as expected and continues to do so even as it evolves.

Faster Development: Although it may seem counterintuitive, TDD often results in faster development as it reduces the time spent debugging and fixing issues later in the development cycle.

Remember: TDD is a cyclical process that promotes continuous improvement and ensures that the codebase remains robust and reliable over time.



Above Picture is represented the Test driven Development

Assignment2:-

Q) Produce a comparative info graphic of TDD, BDD, and FDD methodologies. Illustrate their unique approaches, benefits, and suitability for different software development contexts. Use visuals to enhance understanding.

A) Test-Driven Development (TDD)

Approach: Write tests before writing code. Red-Green-Refactor cycle.

Benefits:

- Early detection of defects.
- Encourages modular design..
- Provides living documentation.

Suitability:

- Works well for small to medium-sized projects.
- Projects where requirements are well-defined and unlikely to change frequently.
- Behaviour-Driven Development (BDD)

Approach: the Focuses on behaviour from user's perspective using Given-When-Then scenarios.

Benefits:

- Promotes collaboration between developers, testers, and business stakeholders.
- Improves understanding of user requirements .
- Reduces ambiguity in requirements.

Suitability:

- Particularly useful for projects with complex business logic.
- Projects where stakeholders have diverse backgrounds and need a common
- language to discuss requirements

Feature-Driven Development (FDD)

Approach: Iterative and incremental development based on features.

Benefits:-

- Emphasizes domain object modelling.
- Allows for rapid development and delivery of features.
- Encourages regular communication and collaboration among team members.

Suitability: Suitable for large projects with multiple teams. Projects where feature delivery needs to be prioritized. Comparison Table is given below.

Aspect	TDD	BDD	FDD
Focus	Testing	Behaviour	Features
Approach	Write tests first	Given-When-Then scenarios	Iterative feature development
Collaboration	Minimal	Extensive	Moderate
Documentation	Living documentation	Clear behaviour specs	Feature-centric
Project Size Suitability	Small to Medium	Any size	Large

This info graphic provides a visual comparison of TDD, BDD, and FDD methodologies, highlighting their unique approaches, benefits, and suitability for different software development contexts. Visual elements such as icons, charts, and color coding can be added.