**Assignment no 2**

**Question:** 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.

1. **Test-Driven Development (TDD):**

**Flowchart:**

Start

Write a test.

Run all tests (the new test should fail).

Write the minimum code to pass the test.

Run all tests (all tests should pass).

Refactor the code.

Repeat.

**Benefits:**

* Ensures code quality and reduces bugs.
* Provides clear documentation of code behavior through tests.
* Facilitates refactoring and maintenance.

**Disadvantages:**

* Can be time-consuming.
* Requires developers to write extensive tests, which can be seen as overhead.

**Suitability:**

Best for projects where code quality and reliability are paramount.

Suitable for critical systems where regression bugs must be avoided.

1. **Behavior-Driven Development (BDD):**

**Flowchart:**

Start

Identify a user story.

Write a scenario in plain language.

Convert the scenario into automated tests.

Run the tests (they should fail initially).

Develop the code to pass the tests.

Run the tests (all tests should pass).

Refactor the code.

Repeat.

**Benefits:**

* Improves communication between non-technical stakeholders and developers.
* Ensures the application meets business requirements.
* Helps in creating a shared understanding of requirements.

**Disadvantages:**

* Can be challenging to write clear, unambiguous user stories.
* Requires collaboration and involvement of non-technical stakeholders.

**Suitability:**

* Ideal for projects with significant stakeholder involvement.
* Useful when requirements need to be clearly defined and understood by all parties.

1. **Feature-Driven Development (FDD):**

**Flowchart:**

Start

Develop an overall model.

Build a features list.

Plan by feature.

Design by feature.

Build by feature.

Review and repeat for each feature.

**Benefits:**

* Focuses on delivering tangible, working functionality.
* Scalable for large teams and complex projects.
* Emphasizes documentation and design.

**Disadvantages:**

* Can be too rigid for projects that require rapid changes.
* Requires upfront planning and a clear understanding of features.

**Suitability:**

* Suitable for large, complex projects with clearly defined features.
* Effective in environments where detailed documentation and planning are necessary.