**Superset-Id: 6364597  
Name: SRINJOY PAUL**

**WEEK -2 NUnit Hands On**

## NUnit Hands-On:

## Q: Explain the meaning of Unit Testing and how it differs from Functional Testing

ANSWER:

Unit Testing is the process of testing individual parts of a program (like methods or classes) in isolation from the rest of the system.

Dependencies are mocked to ensure only the unit's behavior is evaluated. It is typically fast and executed by developers.

On the other hand, Functional Testing evaluates the system against business requirements as a whole. It operates without mocking dependencies and is often conducted by QA testers.

## Q: List different types of software testing

ANSWER:

- **Unit Testing**: Tests the smallest pieces of code independently.

- **Functional Testing**: Validates that the system meets the specified business requirements.

- **Automated Testing**: Runs test scripts/tools to execute tests without manual intervention.

- **Performance Testing**: Measures how the application performs under different workloads.

**Q: Understand the benefit of automated testing ?**

ANSWER:

- Provides rapid feedback during the development cycle.

- Detects issues early in the development phase.

- Minimizes the amount of manual testing needed.

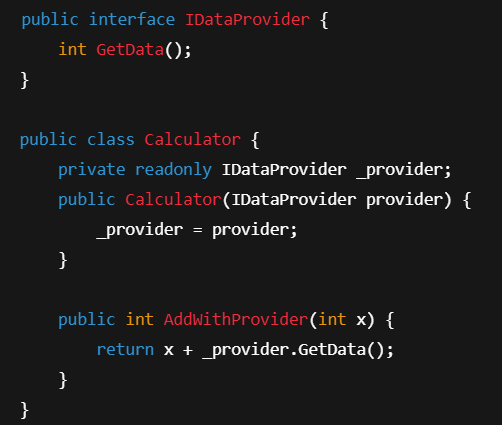
- Facilitates Continuous Integration/Continuous Delivery (CI/CD).

- Helps ensure long-term code quality and dependability.

**Q: Explain what is loosly coupled & testable design**

ANSWER:

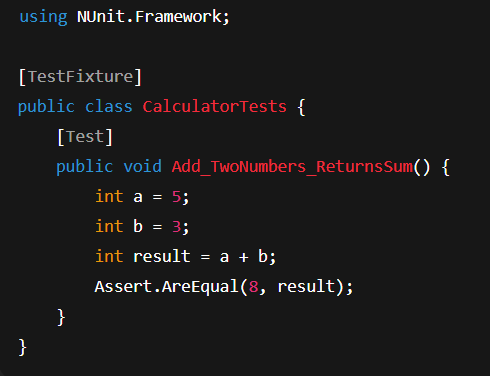
Loosely coupled systems are designed so that components have minimal dependencies on one another. This approach enhances modularity and makes the system easier to test and maintain. Using techniques like interfaces and dependency injection makes the code more adaptable and easier to validate in tests.



**Q: Write your first testing program to validate a calculator addition operation**

* **TestFixture, Test**
* **Understand the need of [SetUp], [TearDown] & [Ignore] attributes.**
* **Explain the benefit of writing parameterised test cases.**
* **TestCase**

ANSWER:



- **[SetUp**]: Executed before each test method. Ideal for initializing required data or objects.

- **[TearDown]**: Executed after each test method. Useful for cleanup tasks, such as closing connections.

- **[Ignore]**: Used to skip a test, typically when it's not yet ready or has known issues.

Why use parameterized test cases:

- They allow testing multiple sets of input/output combinations with a single test method, improving code efficiency and coverage.

An example test file would reside in CalculatorTests.cs within the CalcLibrary.

