**Exercise 1: Mocking and Stubbing**

**Mocking** means creating a **fake version** of a real object (class or interface) so you can:

* Isolate the unit under test.
* Simulate dependencies.
* Track method calls and arguments.

Mocks don’t have real logic unless you define it using **stubbing**.

**Stubbing** is when you **define behavior** of a mock’s method:

*“If method X is called with input Y, return Z.”*

**Real classes**

public class Calculator {

public int add(int a, int b) {

return a + b;

}

}

public class MathService {

Calculator calculator;

public MathService(Calculator calculator) {

this.calculator = calculator;

}

public int doAddition(int a, int b) {

return calculator.add(a, b); // We will mock this

}

}

**JUnit 5 + Mockito Test: Mocking and Stubbing**

import org.junit.jupiter.api.Test;

import org.mockito.Mockito;

import static org.mockito.Mockito.\*;

import static org.junit.jupiter.api.Assertions.\*;

public class MathServiceTest {

@Test

void testDoAdditionWithMockingAndStubbing() {

Calculator mockCalc = Mockito.mock(Calculator.class);

when(mockCalc.add(10, 20)).thenReturn(100);

// Inject mock into service

MathService service = new MathService(mockCalc);

// Act

int result = service.doAddition(10, 20);

// Assert

assertEquals(100, result); // Stubbed result

//Verify the interaction

verify(mockCalc).add(10, 20);

}

}

Mocking

|  |
| --- |
| mock(SomeClass.class) |

|  |
| --- |
| Create a fake object |
| |  | | --- | | Stubbing |  |  | | --- | | when(mock.method(x)).thenReturn(y) |  |  | | --- | | Define what the mock should return | |

|  |
| --- |
| Verifying |

|  |
| --- |
| verify(mock).method(x) |

|  |
| --- |
| Check if a method was called |