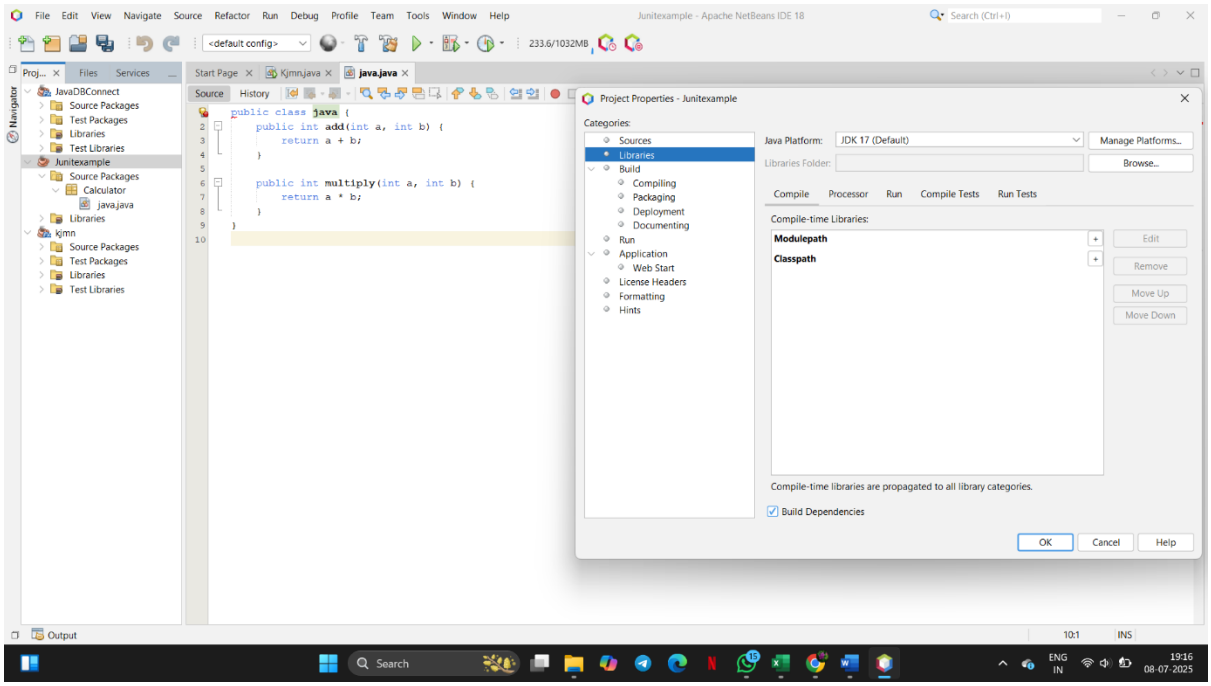
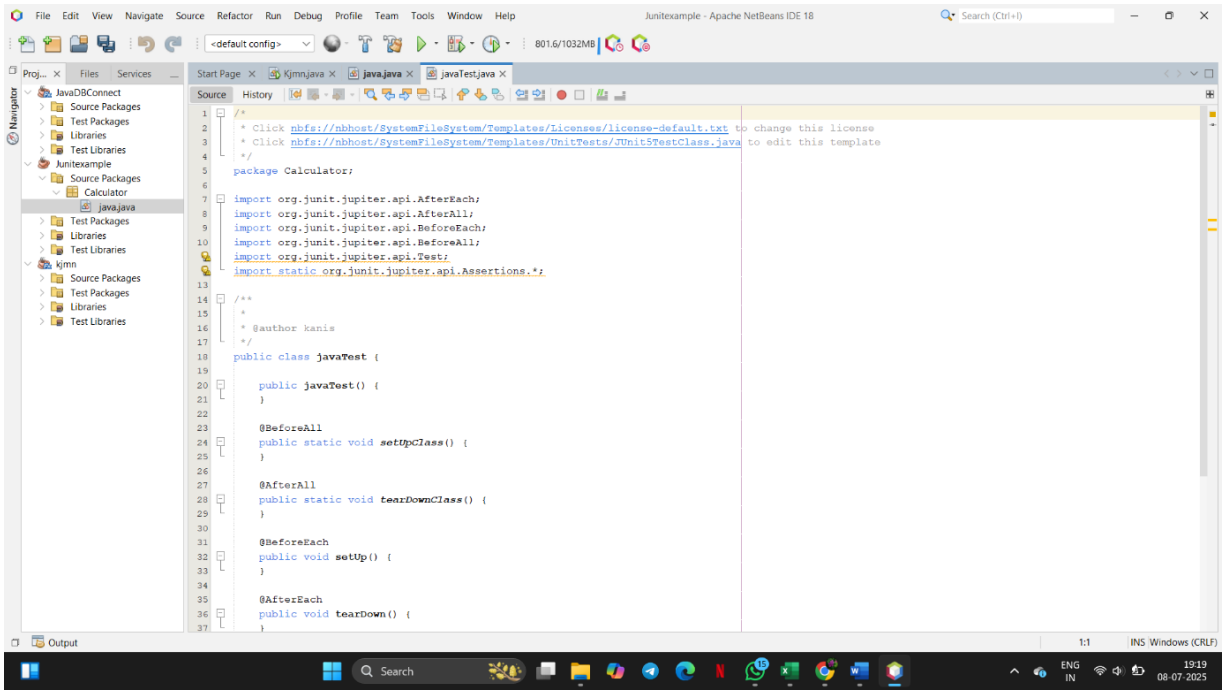
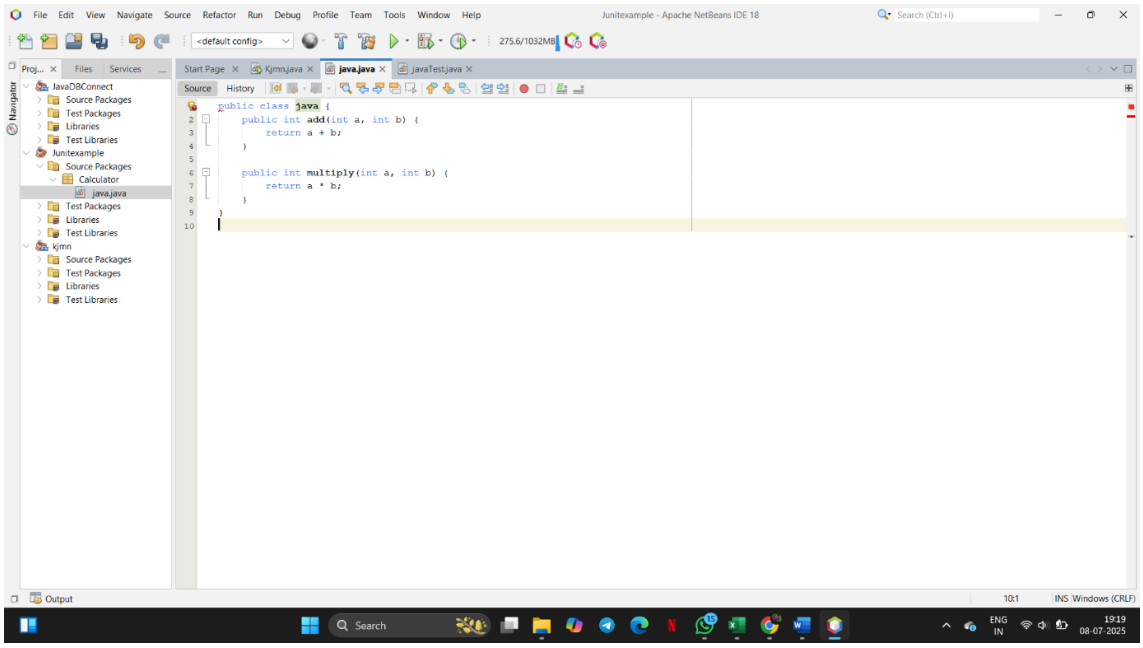
JUnit_Basic Testing Exercises

Exercise 1: Setting Up JUnit







Exercise 3: Assertions in JUnit

```
import org.junit.jupiter.api.Test;
import static org.junit.jupiter.api.Assertions.*;

public class CalculatorTest {

    Calculator calc = new Calculator();

    @Test
    void testAdd() {
        assertEquals(10, calc.add(6, 4));
    }

    @Test
    void testNotEqual() {
        assertNotEquals(12, calc.add(6, 4));
    }

    @Test
    void testDivideByZero() {
        assertThrows(ArithmeticException.class, () -> {
            calc.divide(10, 0);
        });
    }

    @Test
    void testTrueCondition() {
```

```
    assertTrue(5 < 10);
  }

  @Test
  void testFalseCondition() {
    assertFalse(10 < 5);
  }
}
```

Output:
Tests run: 5, Failures: 0, Errors: 0, Skipped: 0

Exercise 4: Arrange-Act-Assert (AAA) Pattern, Test Fixtures, Setup and Teardown Methods in JUnit

Arrange-Act-Assert (AAA) Pattern

```
@Test
void testAdd() {
  // Arrange
  Calculator calc = new Calculator();

  // Act
  int result = calc.add(3, 4);

  // Assert
  assertEquals(7, result);
}
```

Test Fixtures

A **test fixture** is the set of objects and data used in multiple tests. Instead of creating them in every test, we define them **once**, and reuse them.

Setup and Teardown Methods (JUnit 5)

Full Example Using @BeforeEach and @AfterEach

```
import org.junit.jupiter.api.*;

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

public class CalculatorTest {

  Calculator calc;

  @BeforeEach
  void setUp() {
```

```
    calc = new Calculator();
    System.out.println("Setting up Calculator...");
  }

  @AfterEach
  void tearDown() {
    // This runs after every test
    System.out.println("Tearing down after test...\n");
  }

  @Test
  void testAdd() {
    int result = calc.add(10, 5);
    assertEquals(15, result);
  }

  @Test
  void testMultiply() {
    int result = calc.multiply(2, 4);
    assertEquals(8, result);
  }

  @Test
  void testDivideByZero() {
    assertThrows(ArithmeticException.class, () -> {
      calc.divide(10, 0);
    });
  }

  @Test
  void testNotNull() {
    assertNotNull(calc);
  }
}
```

@BeforeAll and @AfterAll

```
@BeforeAll
static void initAll() {
  System.out.println("Start of all tests");
}

@AfterAll
static void tearDownAll() {
  System.out.println("End of all tests");
}
```