**Junit\_Basic Testing Exercises:**

**Exercise 1: Setting Up JUnit**

**Calculator.java**

public class Calculator {

public int add(int a, int b) {

return a + b;

}

}

**CalculatorTest.java**

import static org.junit.Assert.\*;

import org.junit.Test;

public class CalculatorTest {

@Test

public void testAdd() {

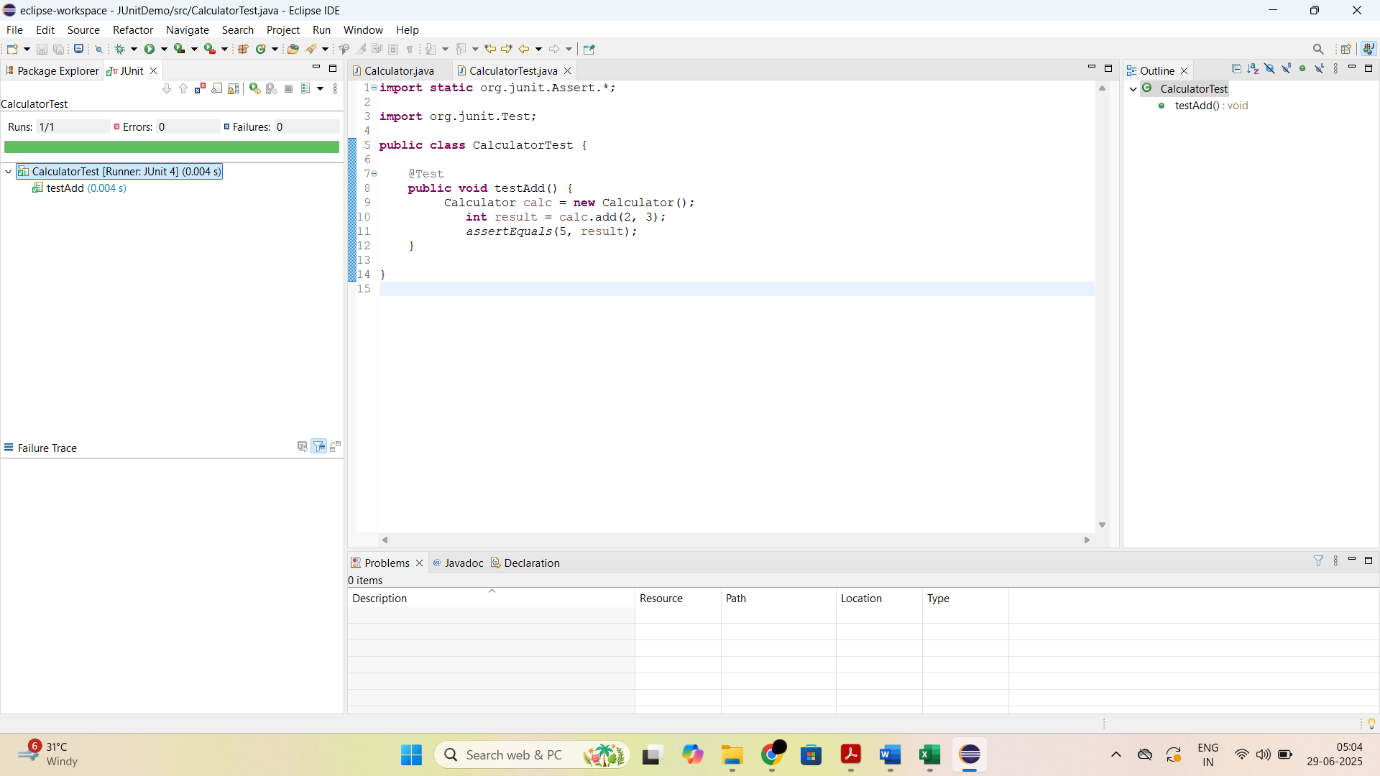
Calculator calc = new Calculator();

int result = calc.add(2, 3);

assertEquals(5, result);

}

}



**Exercise 3 : Assertions in JUnit**

**AssertionsTest.java**

import static org.junit.Assert.\*;

import org.junit.Test;

public class AssertionsTest {

@Test

public void testAssertions() {

assertEquals(5, 2 + 3);

assertTrue(5 > 3);

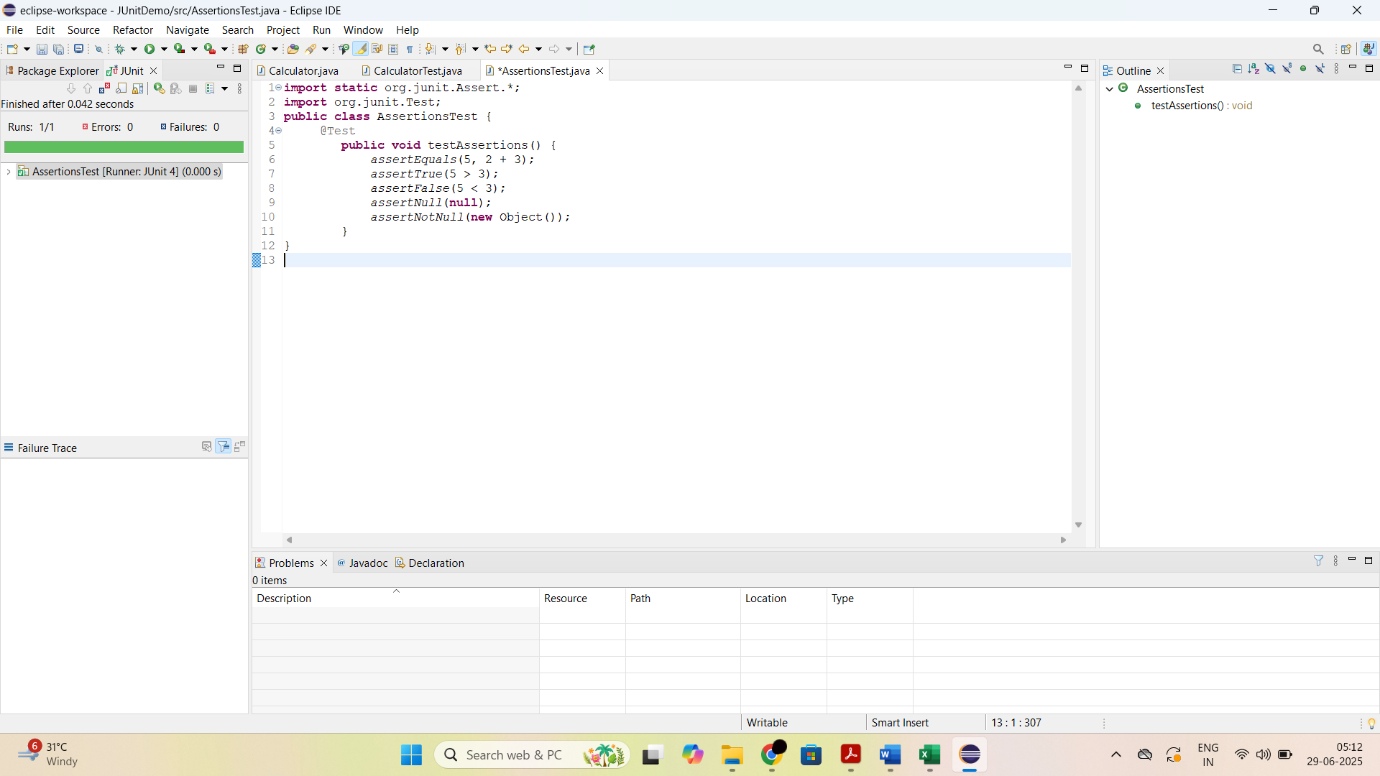
assertFalse(5 < 3);

assertNull(null);

assertNotNull(new Object());

}

}



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

**BankAccount.java**

public class BankAccount {

private int balance;

public BankAccount(int initialBalance) {

this.balance = initialBalance;

}

public void deposit(int amount) {

balance += amount;

}

public void withdraw(int amount) {

balance -= amount;

}

public int getBalance() {

return balance;

}

}

**BankAccountTest.java**

import static org.junit.Assert.\*;

import org.junit.Before;

import org.junit.After;

import org.junit.Test;

public class BankAccountTest {

private BankAccount account;

@Before

public void setUp() {

account = new BankAccount(1000);

}

@After

public void tearDown() {

account = null;

}

@Test

public void testDeposit() {

account.deposit(500);

assertEquals(1500, account.getBalance());

}

@Test

public void testWithdraw() {

account.withdraw(300);

assertEquals(700, account.getBalance());

}

}

