**JUNIT BASIC TESTING EXERCISES**

**Exercise 1: Setting Up Junit**

**Greet.java**

package com.example;

public class Greet {

public String sayHello() {

return "Hello, JUnit!";

}

}

**GreetTest.java**

package com.example;

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

import org.junit.jupiter.api.Test;

public class GreetTest {

*@Test*

public void testSayHello() {

Greet g = new Greet();

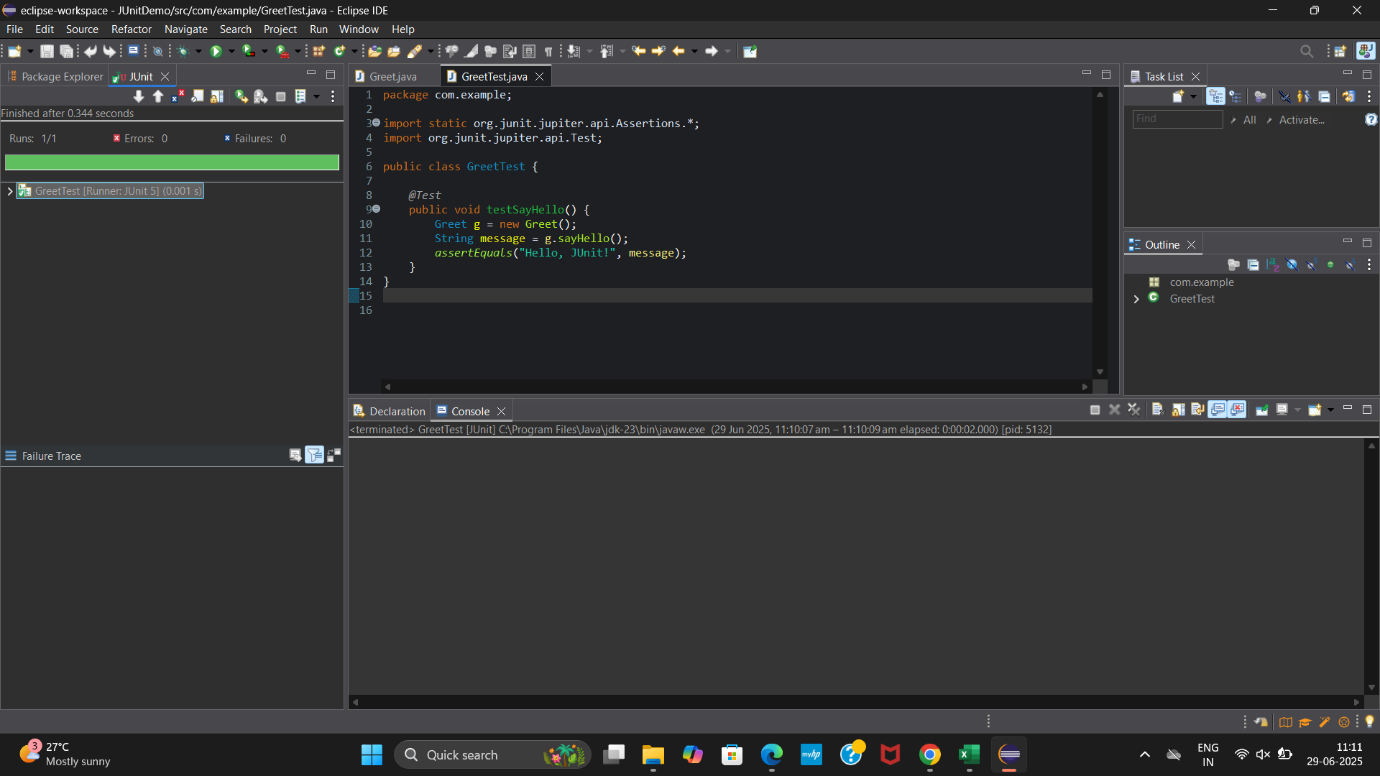
String message = g.sayHello();

*assertEquals*("Hello, JUnit!", message);

}

}

**OUTPUT**

****

**Exercise 2: Writing Basic JUnit Tests**

**Calculator.java**

package com.example;

public class Calculator {

public int add(int a, int b) {

return a+b;

}

public int subtract(int a, int b) {

return a-b;

}

public int multiply(int a, int b) {

return a\*b;

}

public int divide(int a, int b) {

if (b==0) {

throw new IllegalArgumentException("Cannot divide by zero");

}

return a/b;

}

}

**CalculatorTest.java**

package com.example;

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

import org.junit.jupiter.api.Test;

public class CalculatorTest {

*@Test*

public void testAdd() {

Calculator calc = new Calculator();

*assertEquals*(5, calc.add(2, 3));

}

*@Test*

public void testSubtract() {

Calculator calc = new Calculator();

*assertEquals*(2, calc.subtract(5, 3));

}

*@Test*

public void testMultiply() {

Calculator calc = new Calculator();

*assertEquals*(15, calc.multiply(3, 5));

}

*@Test*

public void testDivide() {

Calculator calc = new Calculator();

*assertEquals*(2, calc.divide(10, 5));

}

*@Test*

public void testDivideByZero() {

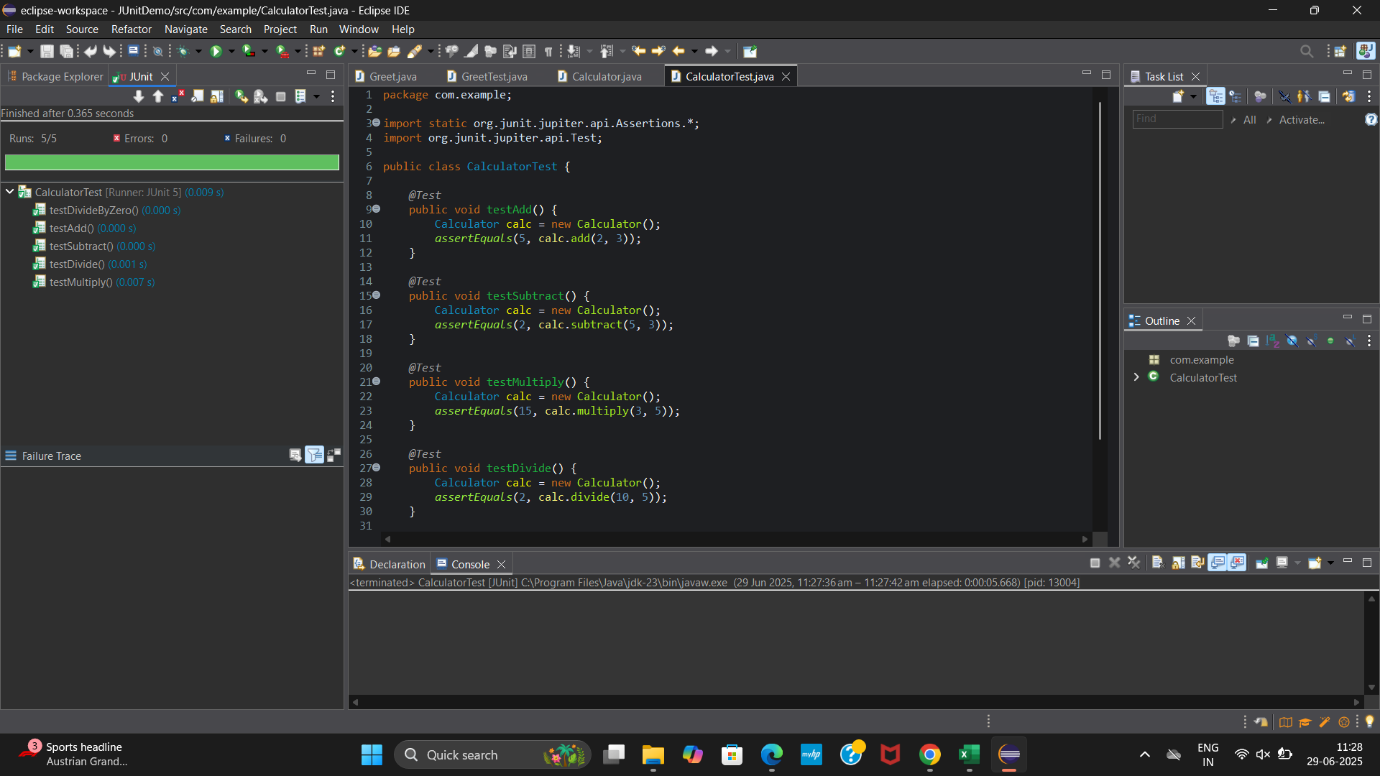
Calculator calc = new Calculator();

*assertThrows*(IllegalArgumentException.class, () -> calc.divide(10, 0));

}

}

**OUTPUT**



**Exercise 3: Assertions in Junit**

**AssertionsTest.java**

package com.example;

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

import org.junit.jupiter.api.Test;

public class AssertionsTest {

*@Test*

public void testAssertions() {

*assertEquals*(5, 2 + 3);

*assertTrue*(5 > 3);

*assertFalse*(5 < 3);

Object obj = null;

*assertNull*(obj);

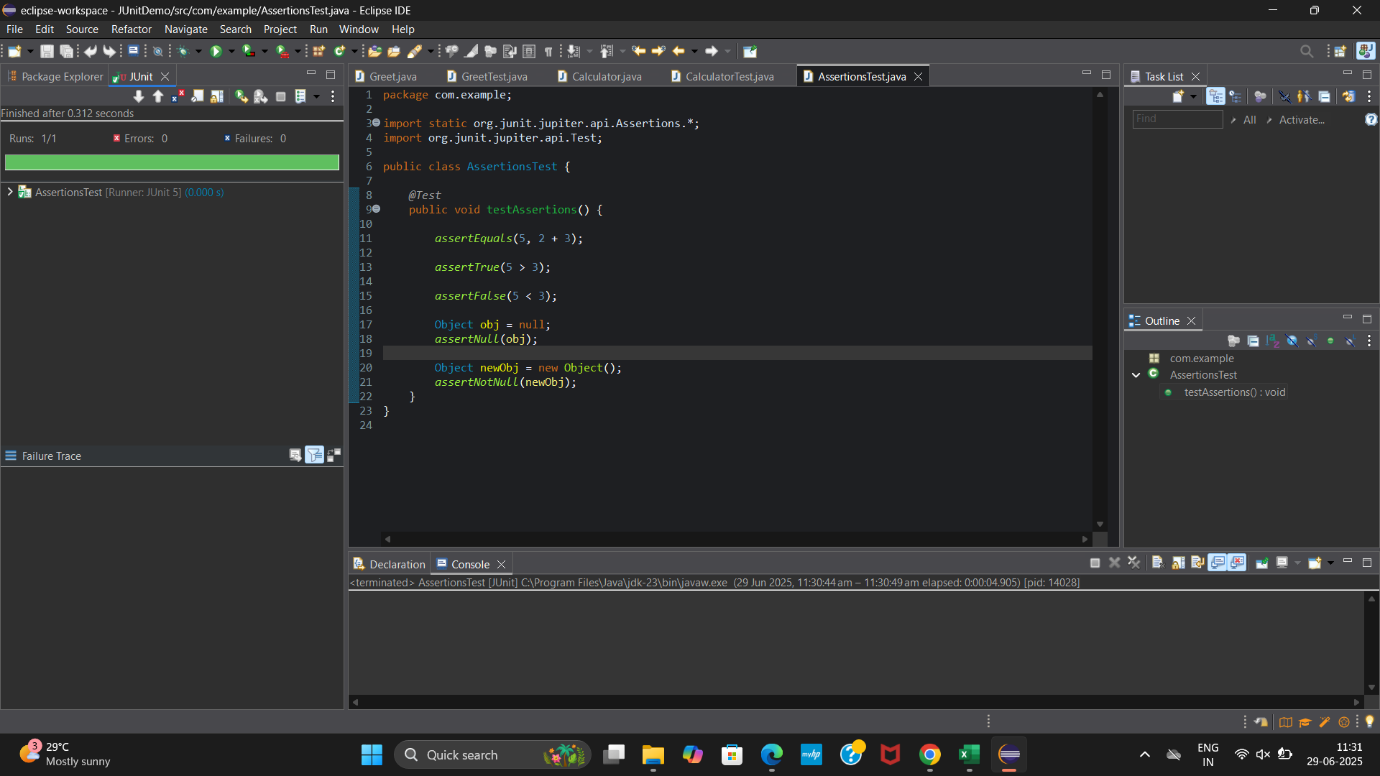
Object newObj = new Object();

*assertNotNull*(newObj);

}

}

**OUTPUT**

****

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

**BankAccount.java**

package com.example;

public class BankAccount {

private int balance;

public void deposit(int amount) {

balance += amount;

}

public void withdraw(int amount) {

if (amount > balance) {

throw new IllegalArgumentException("Insufficient balance");

}

balance -= amount;

}

public int getBalance() {

return balance;

}

}

**BankAccountTest.java**

package com.example;

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

import org.junit.jupiter.api.BeforeEach;

import org.junit.jupiter.api.AfterEach;

import org.junit.jupiter.api.Test;

public class BankAccountTest {

private BankAccount account;

*@BeforeEach*

public void setUp() {

account = new BankAccount();

account.deposit(100);

}

*@AfterEach*

public void tearDown() {

account = null;

}

*@Test*

public void testDeposit() {

account.deposit(50);

*assertEquals*(150, account.getBalance());

}

*@Test*

public void testWithdraw() {

account.withdraw(30);

*assertEquals*(70, account.getBalance());

}

*@Test*

public void testWithdrawInsufficientBalance() {

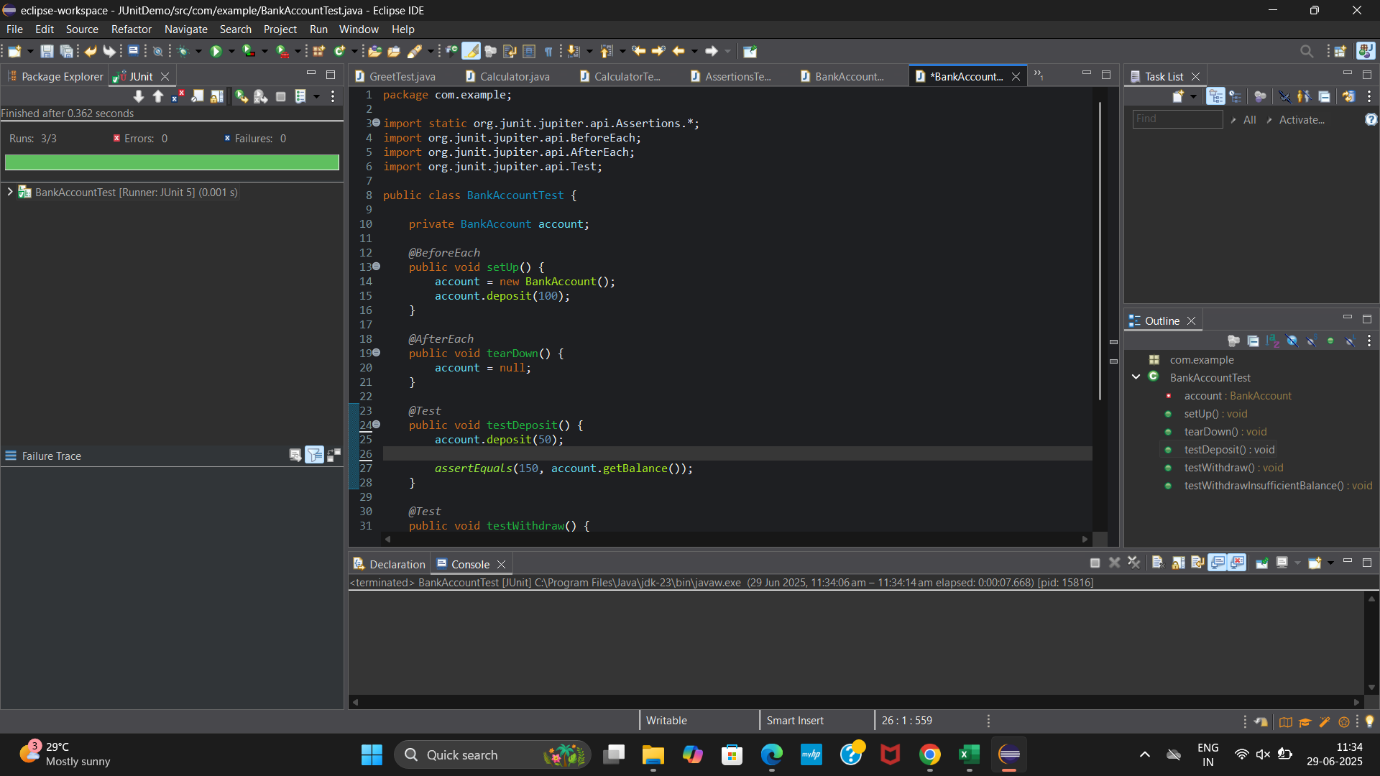
Exception e = *assertThrows*(IllegalArgumentException.class, () -> account.withdraw(200));

*assertEquals*("Insufficient balance", e.getMessage());

}

}

**OUTPUT**

****