**WEEK-2**

**JUnit Testing Exercises**

**Exercise 1: Setting up JUnit**

**Code:**

**NumberUtils.java**

package com.example.Junit\_example;

public class NumberUtils {

public boolean isEven(int number) {

return number % 2 == 0;

}

}

**NumberUtilsTest.java**

package com.example.Junit\_example;

import org.junit.Test;

import static org.junit.Assert.\*;

public class NumberUtilsTest {

@Test

public void testIsEvenTrue() {

NumberUtils n = new NumberUtils();

assertTrue(n.isEven(8));

}

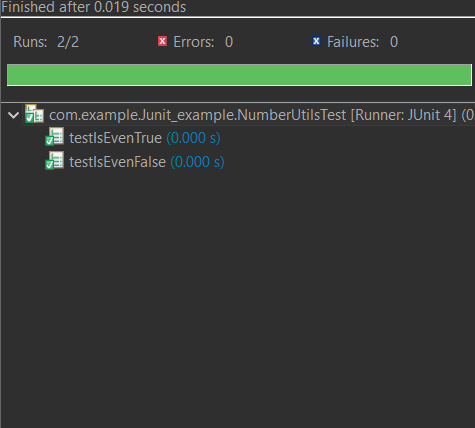
@Test

public void testIsEvenFalse() {

NumberUtils n = new NumberUtils();

assertFalse(n.isEven(7)); }}

**OUTPUT:**

****

**Exercise 3: Assertions in JUnit**

**Code:**

**Assertions.java**

package com.example.Junit\_example;

import org.junit.Test;

import static org.junit.Assert.\*;

public class AssertionsTest {

@Test

public void testAssertions() {

// Assert equals

assertEquals(5, 2 + 3);

// Assert true

assertTrue(5 > 3);

// Assert false

assertFalse(5 < 3);

// Assert null

assertNull(null);

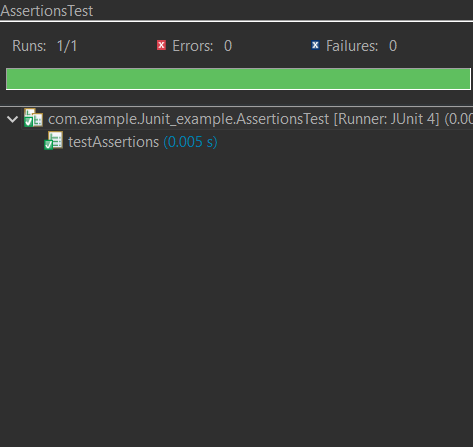
// Assert not null

assertNotNull(new Object());

}

}

**OUTPUT:**

****

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

**Code:**

**BankAccount.java**

package com.example.Junit\_example;

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**

package com.example.Junit\_example;

import org.junit.Before;

import org.junit.After;

import org.junit.Test;

import static org.junit.Assert.\*;

public class BankAccountTest {

BankAccount account;

@Before

public void setUp() {

// Arrange

account = new BankAccount(100);

System.out.println("Account setup");

}

@After

public void tearDown() {

account = null;

System.out.println("Account cleaned up");

}

@Test

public void testDeposit() {

// Act

account.deposit(50);

// Assert

assertEquals(150, account.getBalance());

}

@Test

public void testWithdraw() {

// Act

account.withdraw(30);

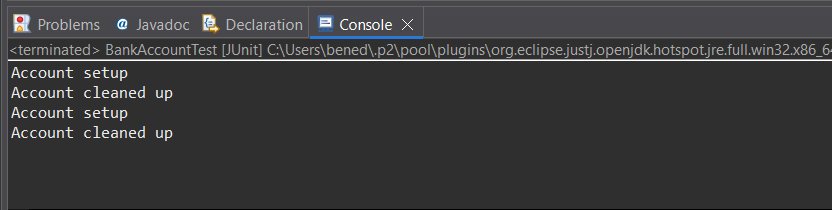
// Assert

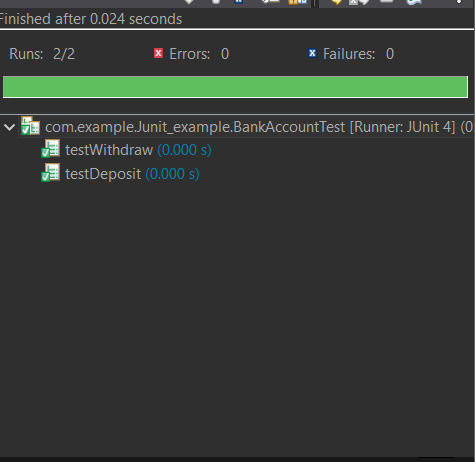
assertEquals(70, account.getBalance());

}

}

**OUTPUT:**

****

****

**Mockito Exercises**

**Exercise 1: Mocking and Stubbing**

**Code:**

**ExternalApi.java**

package com.example.Junit\_example;

public class ExternalApi {

String getData();

}

**MyService.java**

package com.example.Junit\_example;

public class MyService {

private ExternalApi api;

public MyService(ExternalApi api) {

this.api = api;

}

public String fetchData() {

return api.getData();

}

}

**MyServiceTest.java**

package com.example.Junit\_example;

import static org.junit.Assert.assertEquals;

import static org.mockito.Mockito.\*;

import org.junit.jupiter.api.Test;

import org.mockito.Mockito;

public class MyServiceTest {

@Test

public void testExternalApi() {

ExternalApi mockApi = Mockito.mock(ExternalApi.class);

when(mockApi.getData()).thenReturn("Mock Data");

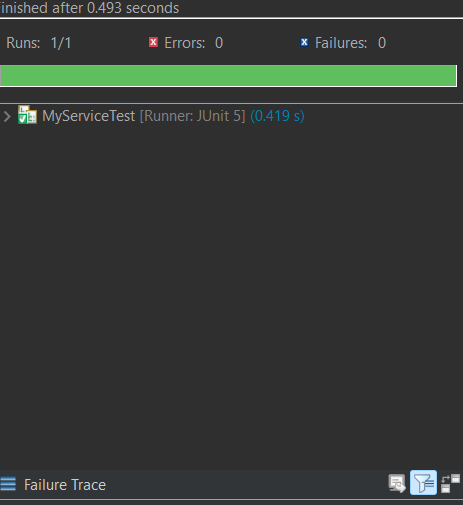
MyService service = new MyService(mockApi);

String result = service.fetchData();

assertEquals("Mock Data", result);

} }

**OUTPUT:**



**Exercise 2: Verifying Interactions**

**Code:**

**ExternalApi.java**

package com.example.Junit\_example;

public class ExternalApi {

String getData();

}

**MyService.java**

package com.example.Junit\_example;

public class MyService {

private ExternalApi api;

public MyService(ExternalApi api) {

this.api = api;

}

public String fetchData() {

return api.getData();

}

}

**MyServiceTest.java**

package com.example.Junit\_example;

import static org.mockito.Mockito.\*;

import org.junit.jupiter.api.Test;

import org.mockito.Mockito;

public class MyServiceTest {

@Test

public void testVerifyInteraction() {

ExternalApi mockApi = Mockito.mock(ExternalApi.class);

MyService service = new MyService(mockApi);

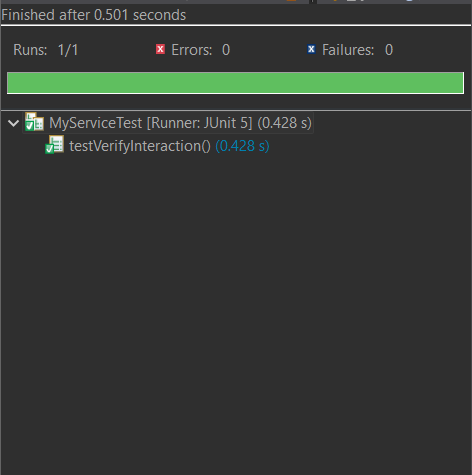
service.fetchData();

verify(mockApi).getData();

}

}

**OUTPUT:**

****