Day 23

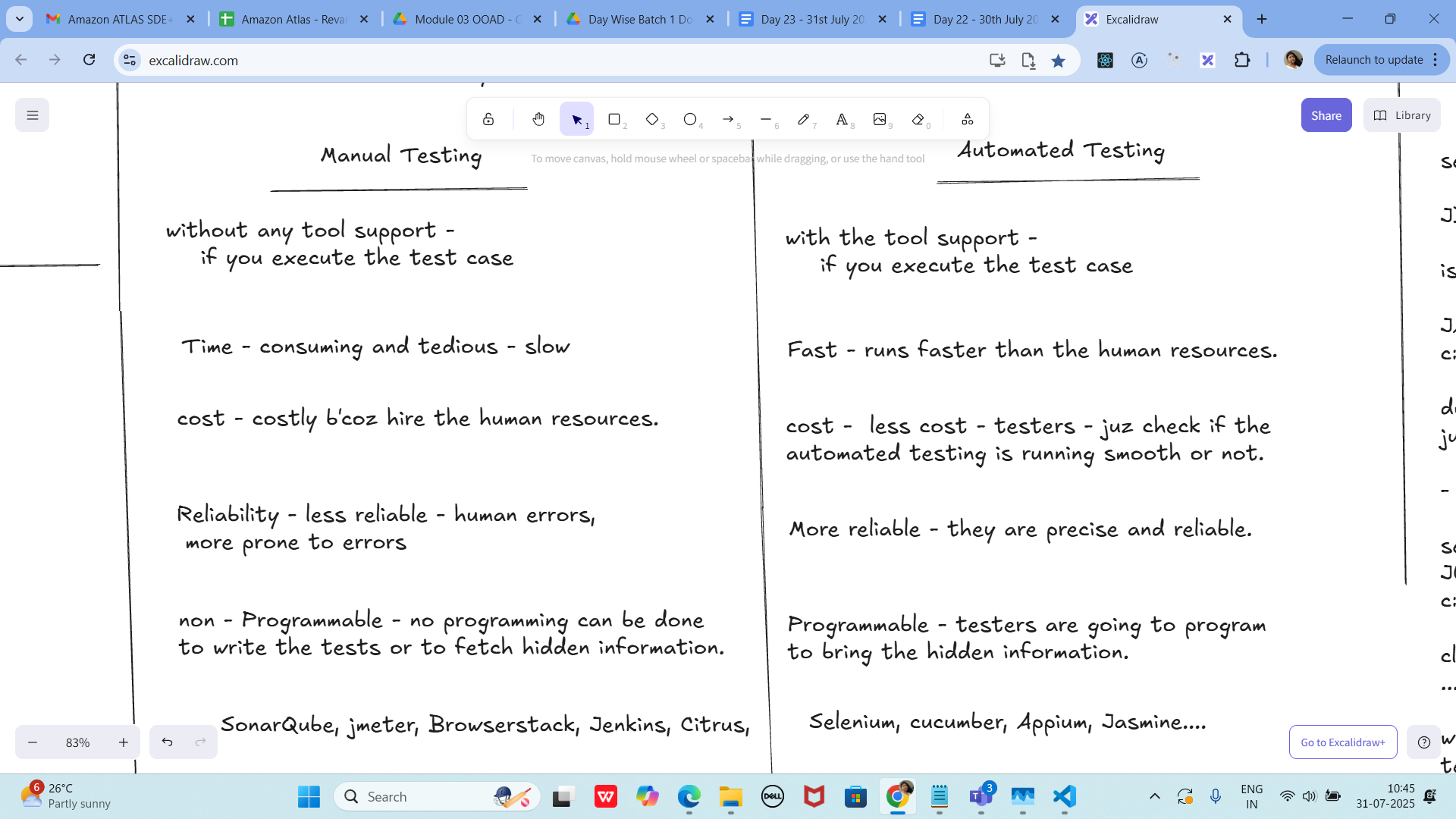
Employee ID: 201933938

Login ID: iamasif

Name: Shaik Asif

<https://excalidraw.com/#json=_Cup10oNwNyB6wT2cbCXc,y88IaceH_ptz9HkLCpIJrg>

Search Atlas Batch 2 31th july



Junit

setup - environmental

JDk 1.5 or above

is java installed in your system

JAVA\_HOME = "...path "

c:\programfiles\Java\jdk...

download JUnit archive

junit.org/junit5/

- .jar file ==> junit4.11.jar

set environmental variables

JUNIT\_HOME = "Path.."

c:\programfile\junit

classpath ==> junit jar

..... \junit4.11.jar

write the code: -

testcase write

see the result pass or fail

cmd:

javac TestJnuit.java TestRunner.java

java TestRunner

TestJunit --> @test...... cases

TestRunner --> psvm ( ==> driver class for Junittest

=========================================================

Task 01

TestJunit.java

import org.junit.Test;

import static org.junit.Assert.assertEquals;

public class TestJunit {

@Test

public void testSubtract() {

int num1 = 10;

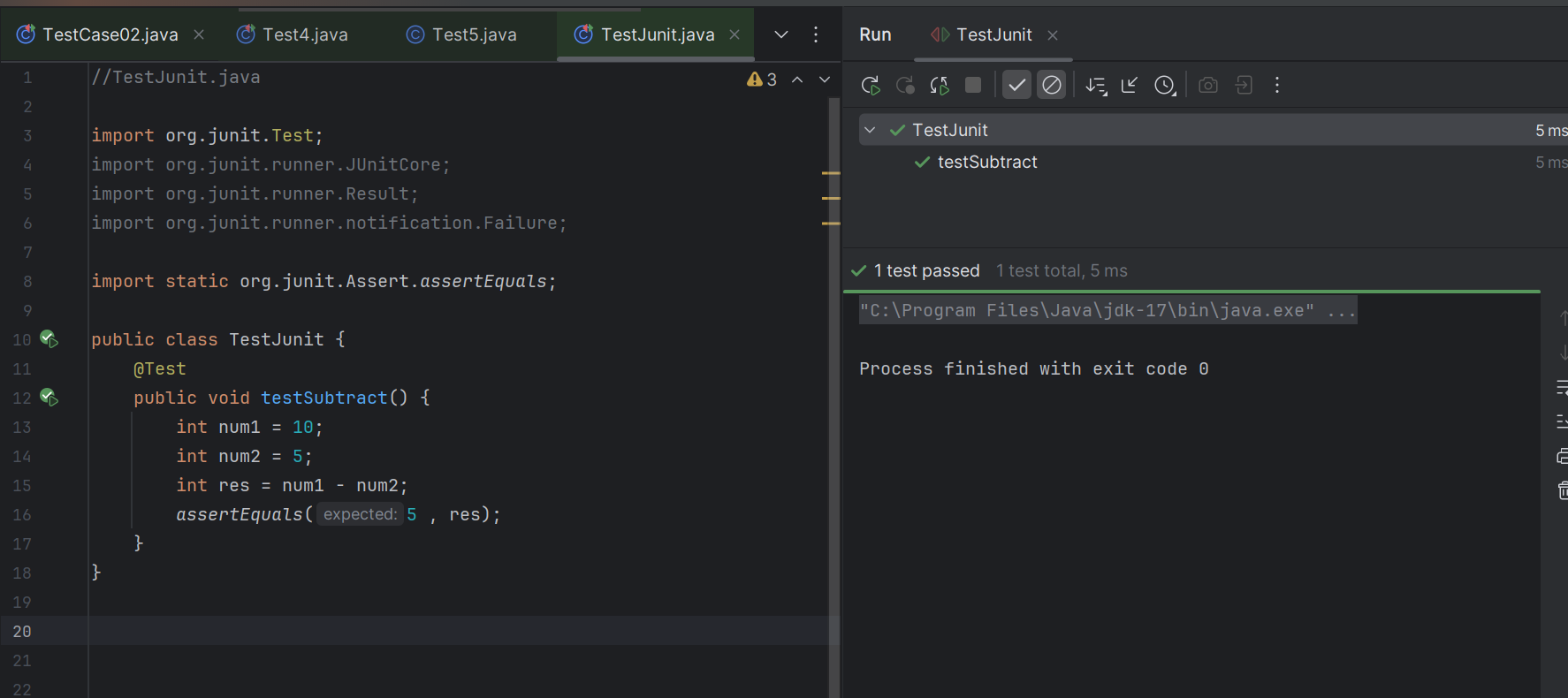
int num2 = 5;

int res = num1 - num2;

assertEquals(5 , res);

}

}



TestRunner.java

import org.junit.runner.JUnitCore;

import org.junit.runner.Result;

import org.junit.runner.notification.Failure;

public class TestRunner {

public static void main(String[] args) {

Result resobj = JUnitCore.runClasses(TestJunit.class);

for (Failure fobj : resobj.getFailures()) {

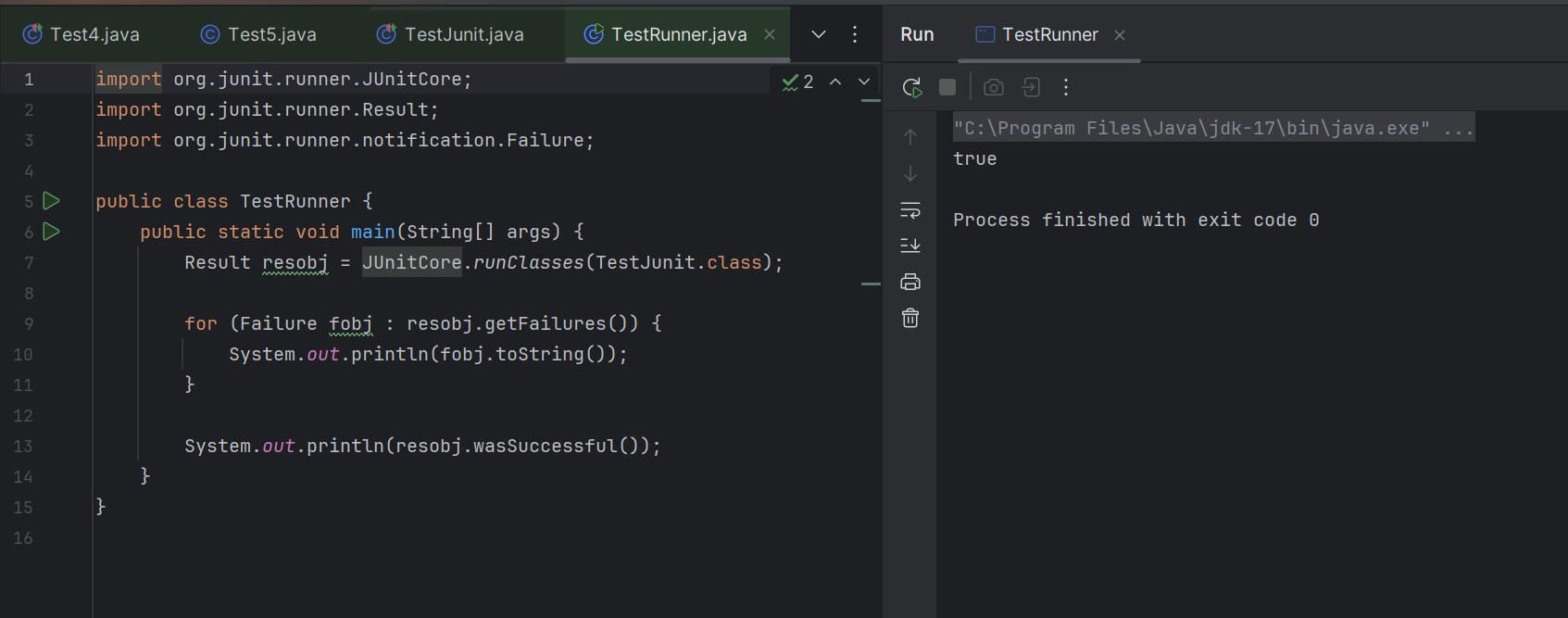
Sout(fobj.toString());

}

sout(resobj.wasSuccessful()); //--> true

}

}



===================================================================

Junit test Framework

Regression Testing framework.

- Junit - unit Testing in java

- Junit - it will increase the programming speed, quality of my code will increase.

Junit - integrated with

Maven

Eclipse

Ant

features of Junit Test

Framework

JUnit classes - TestResult

- Assert

- TestCase

Test runners

Test suites

Fixtures

Fixtures

-fixed state of set of objects

- use for baseline for running tests

- why ? they ensure that

- the test cases will run in

fixed environment

- repeatedly.. test cases.

few methods:

tearDown() : - if you want you

to run after each and every test method

setUp(): if you want you

to run before each and every test method

Fixtures sample code:

import junit.framework.\*;

public class FixturesDemo extends TestCase {

protected int num1,num2;

protected void setUp() {

num1 = 100;

num2 = 200;

}

//test method to multiply numbers

public void testMultiply() {

int res = num1 \* num2;

assertTrue(res == 20000);

}

}

Test Suite:

- its a bundle of unit test cases and run them

all together.

@Suite

@RunWith

TestSuite Demo

import org.junit.runner.RunWith;

import org.junit.runner.Suite;

@RunWith(Suite.class)

@Suite.SuiteClasses( {

JunitTest01.class,

JunitTest02.class,

JunitTest03.class

})

public class TestSuiteDemo {

}

JunitTest01.java

import org,junit.Test;

import static org.junit.Assert.assetEquals;

import org.junit.Igore;

public class junitTest01 {

String msg = "running test01 ";

MessageUtil msgUtilobj = new MessageUtil(msg);

//MessageUtil is a utility class

//store a message

@Test

public void msgTest() {

sout("we are inside the msgtest()");

assertEquals(msg, msgUtilobj.printMessage());

}

}

junitTest02.java

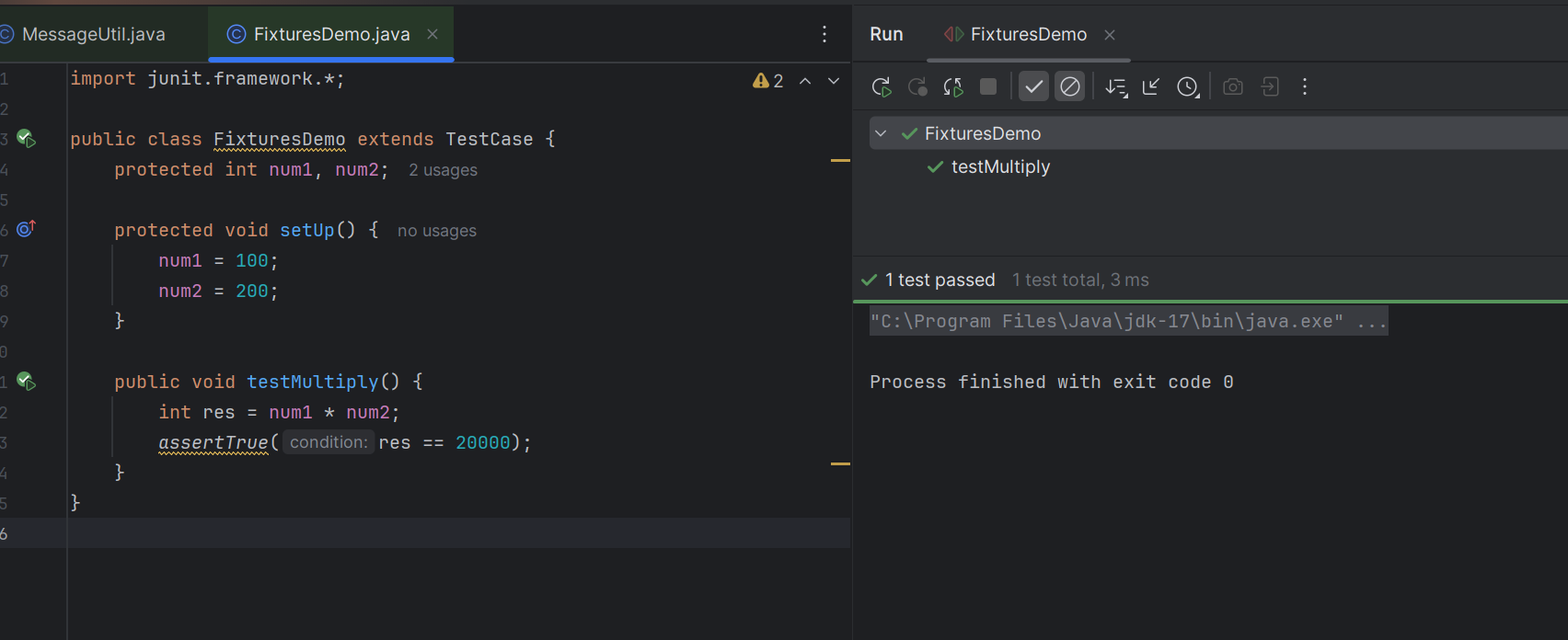
test Runner

Test 02:

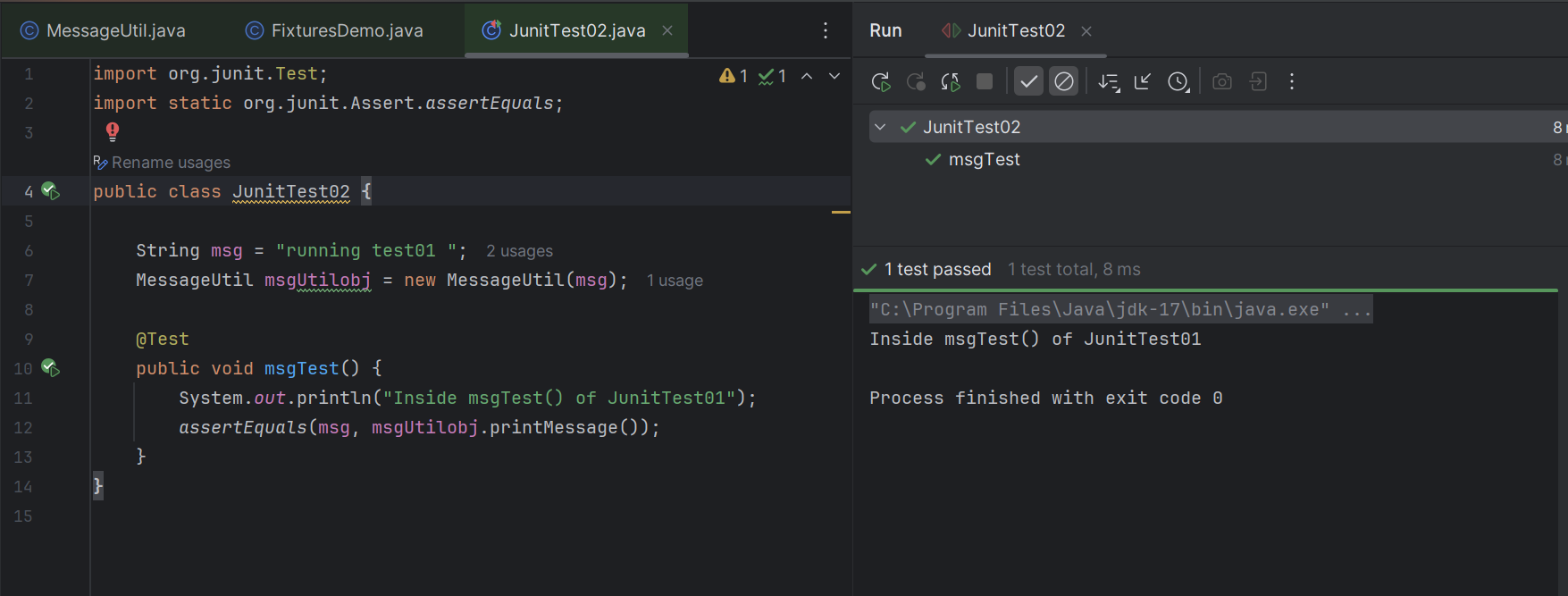
Implement the above code also add one more test case and test runner.. And execute the test suite.

14.38 to 14.46 8 min

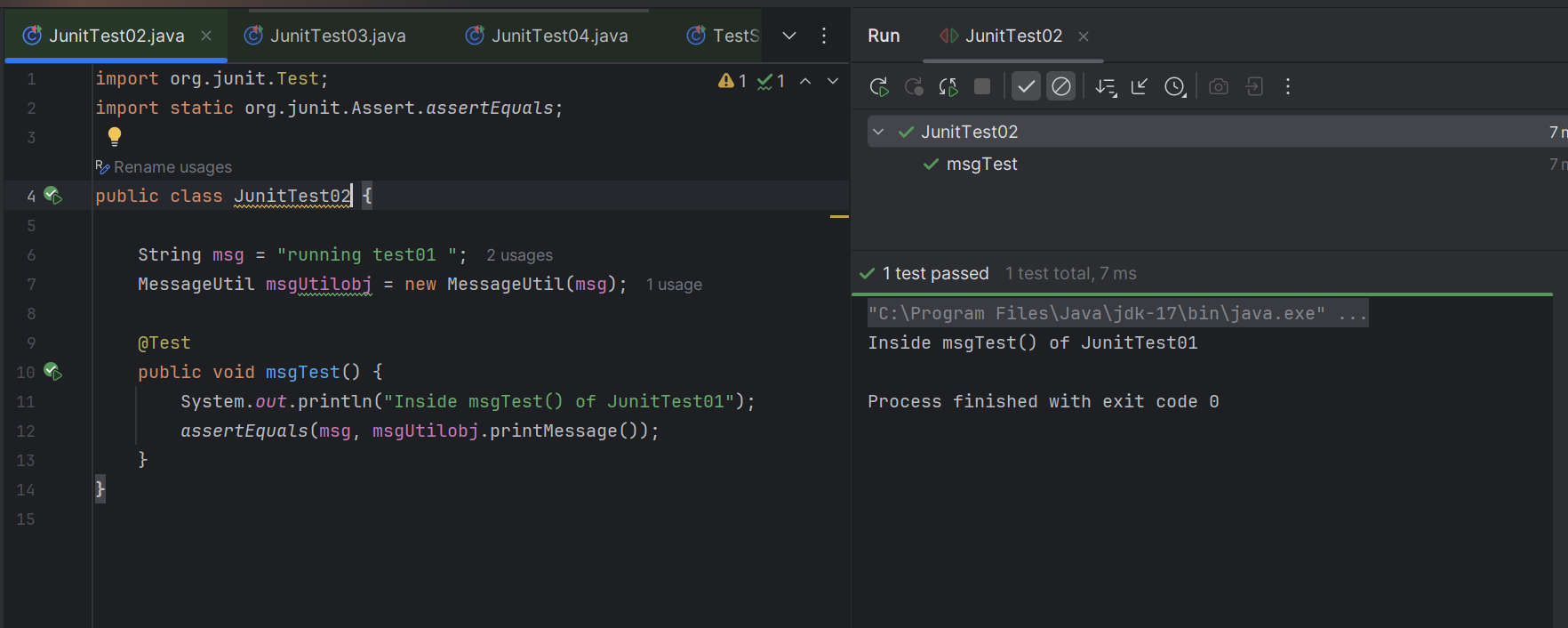
**MessageUtil.java:**



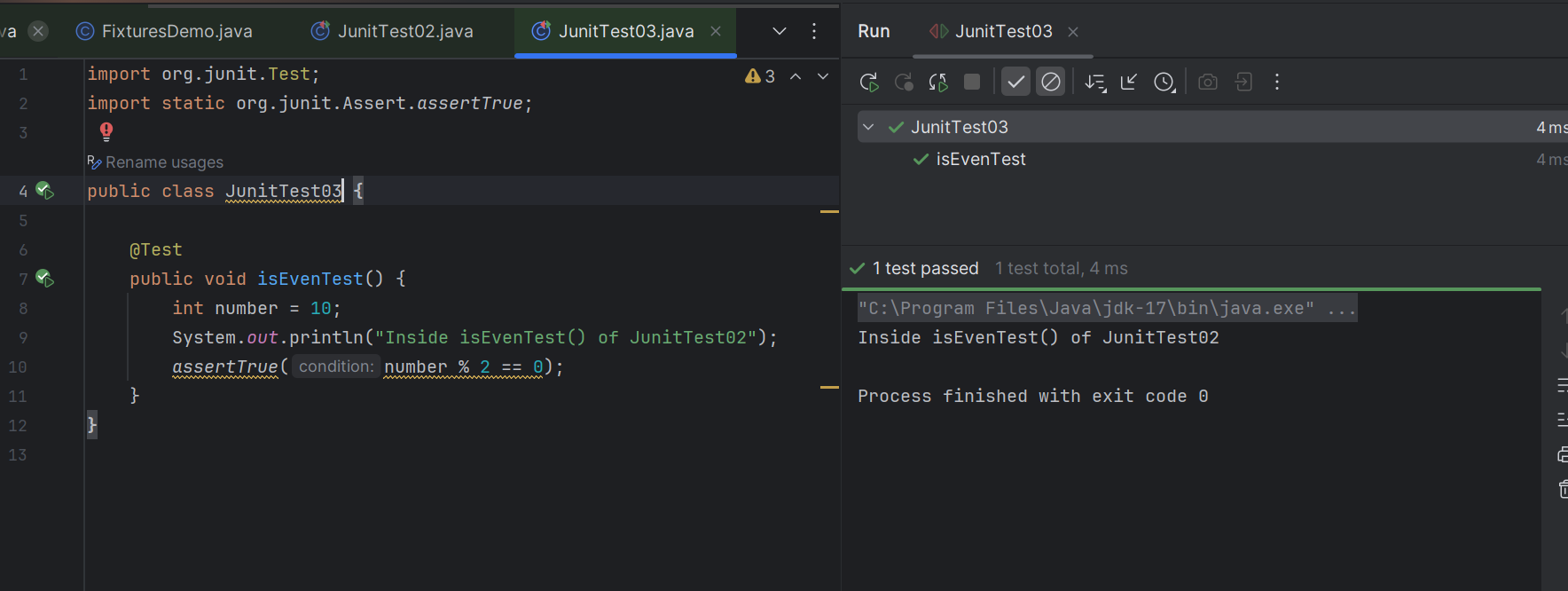
**FixturesDemo.java:**



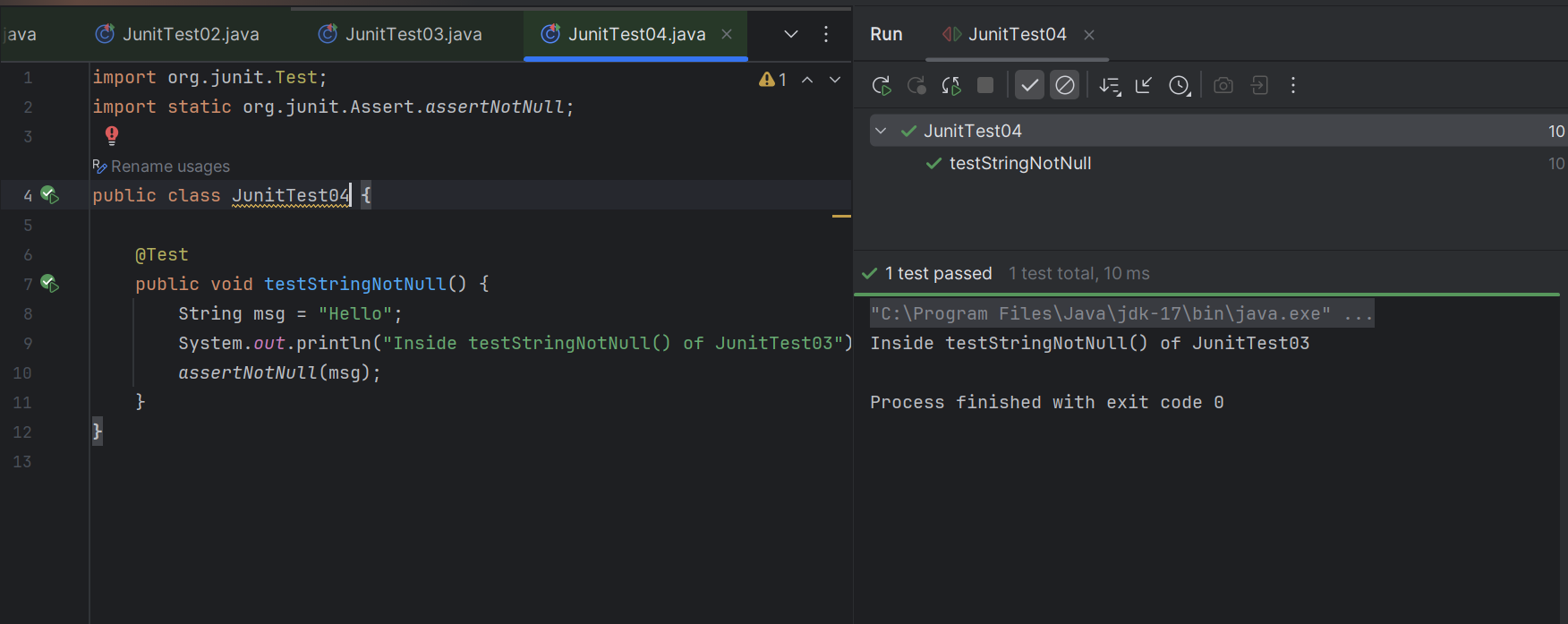
**JunitTest03.java:**



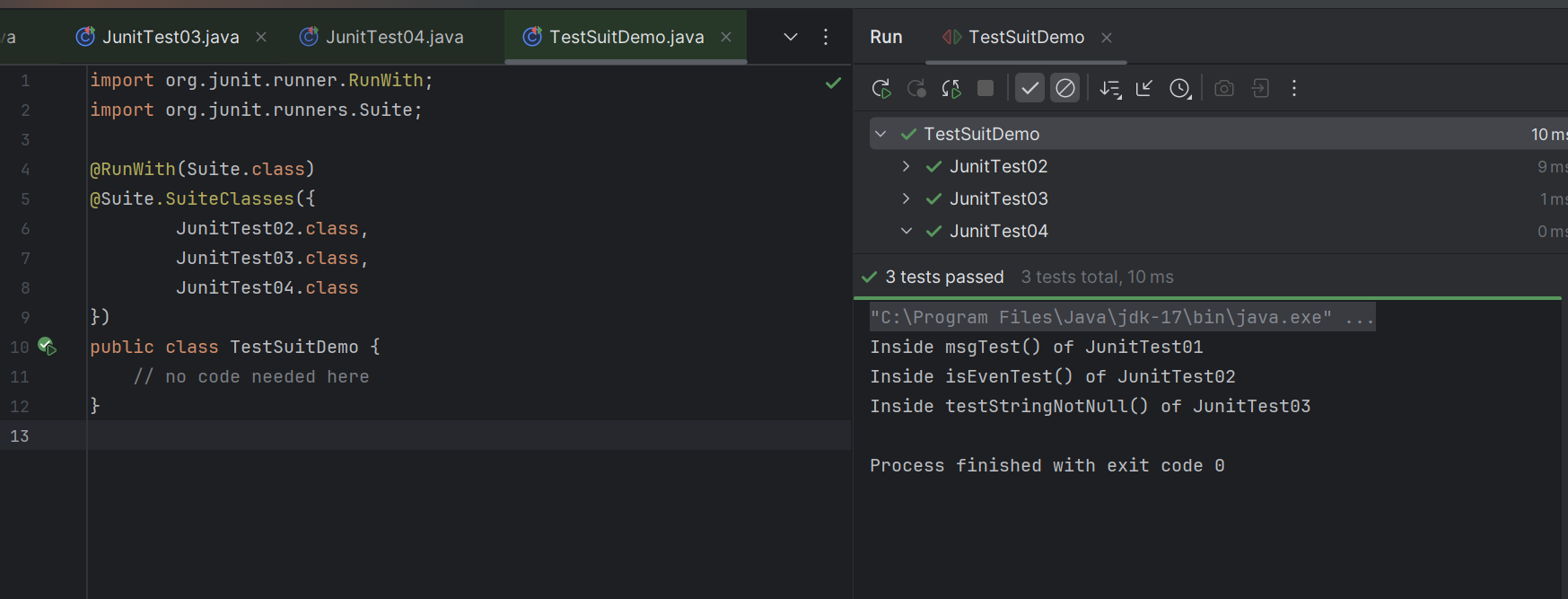
**JunitTest03.java:**



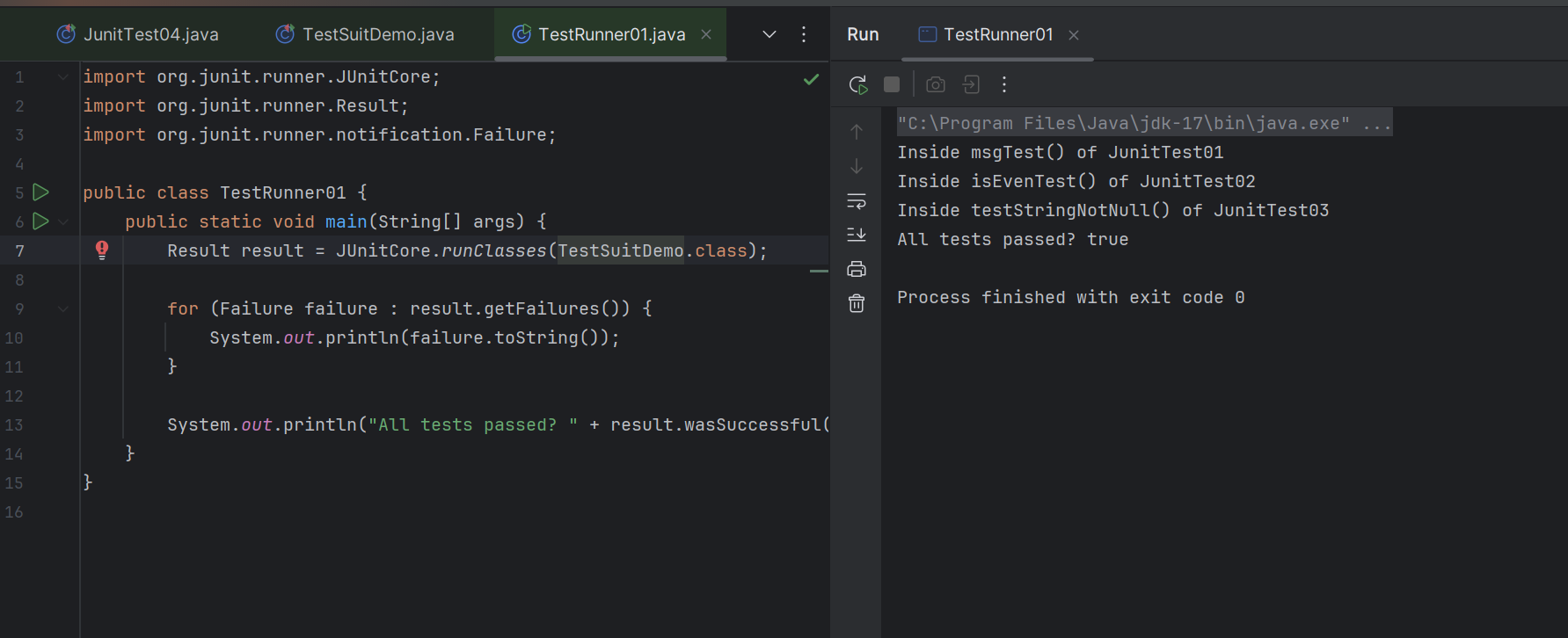
**JunitTest04.java:**



**TestSuit.java:**

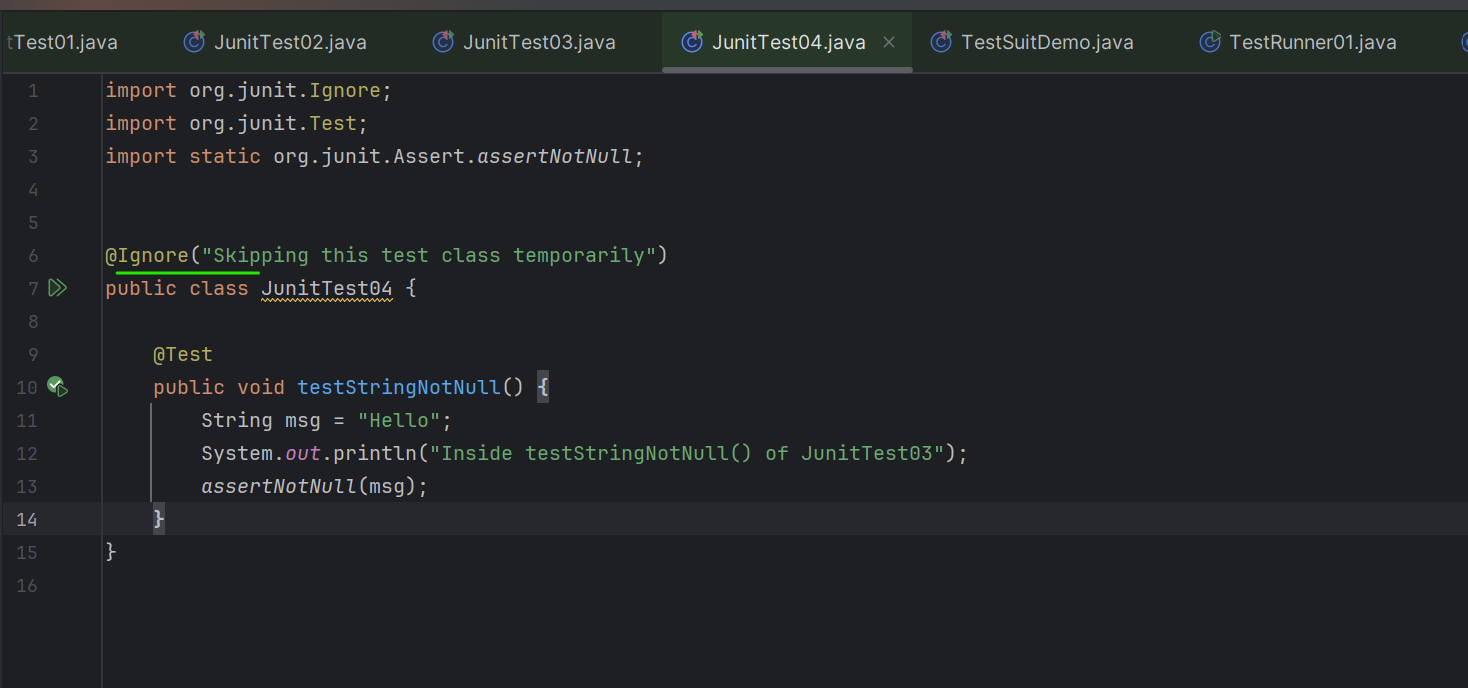


**TestRunner.java:**

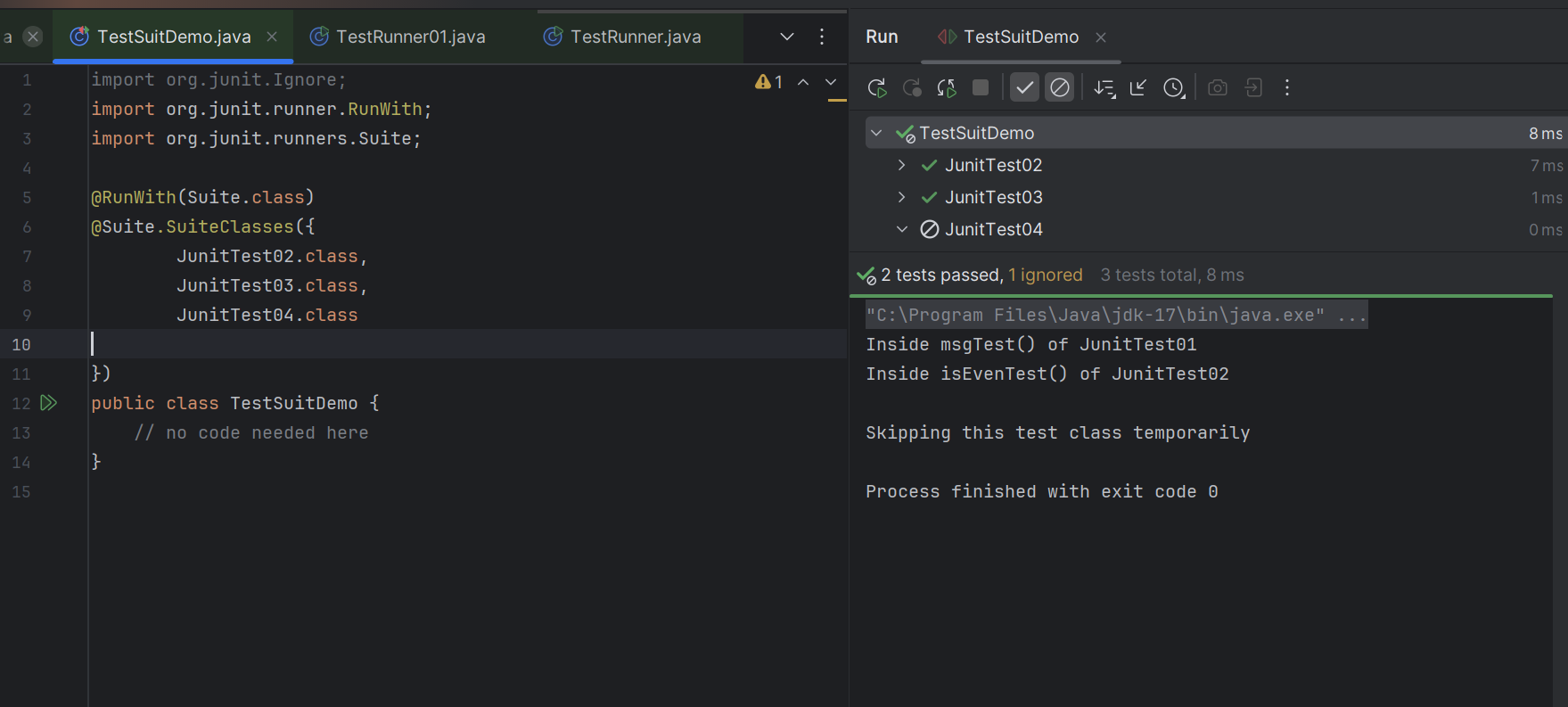


Task 03:

Add @Ignore to your one of the test case and check if thet case is ignored or not



**JunitTest04 Ignored:**

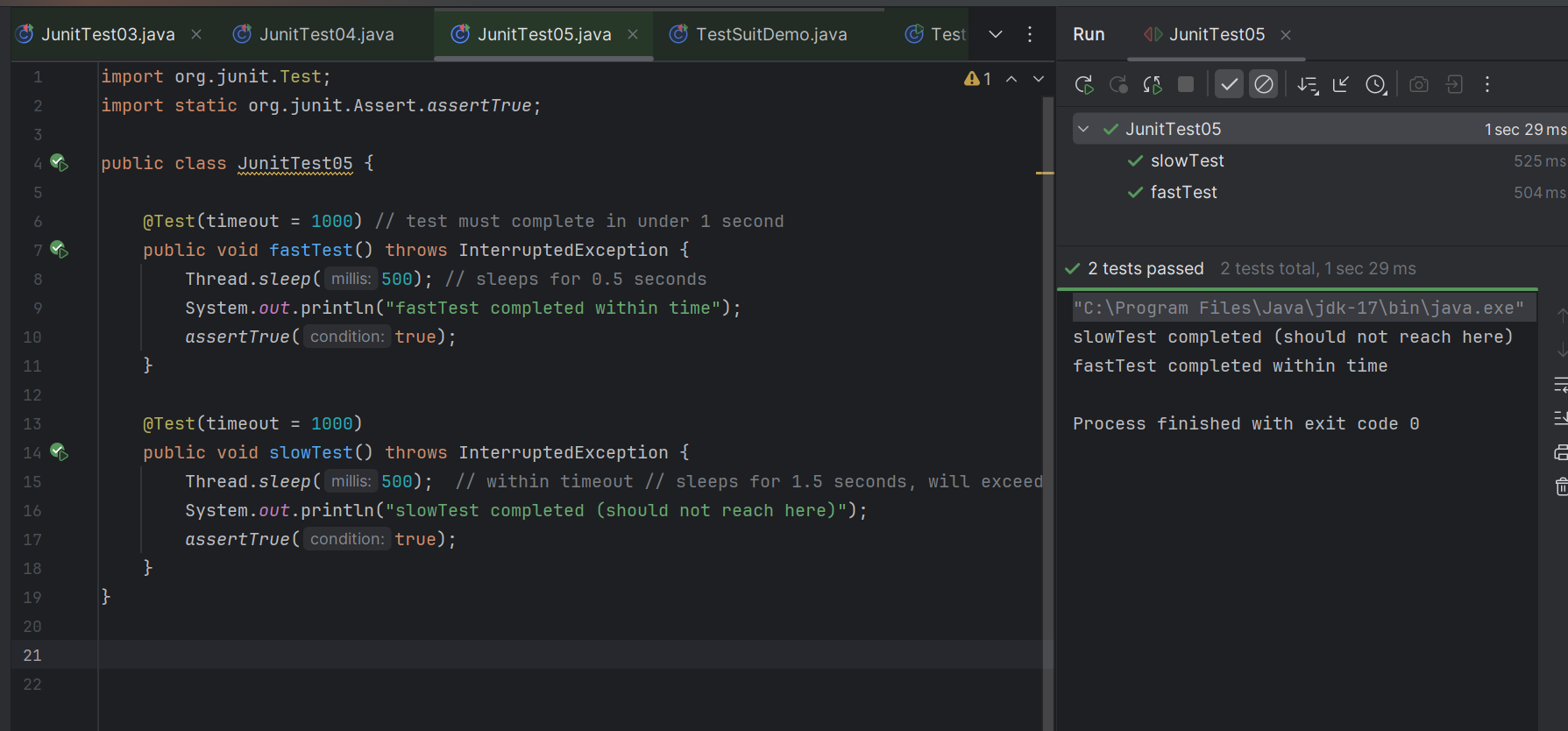
****

Task 04:

Time Test

In one the tests pass the timeout parameter with value and check if the test case fails if the time is out

@Test(timeout = 1000)

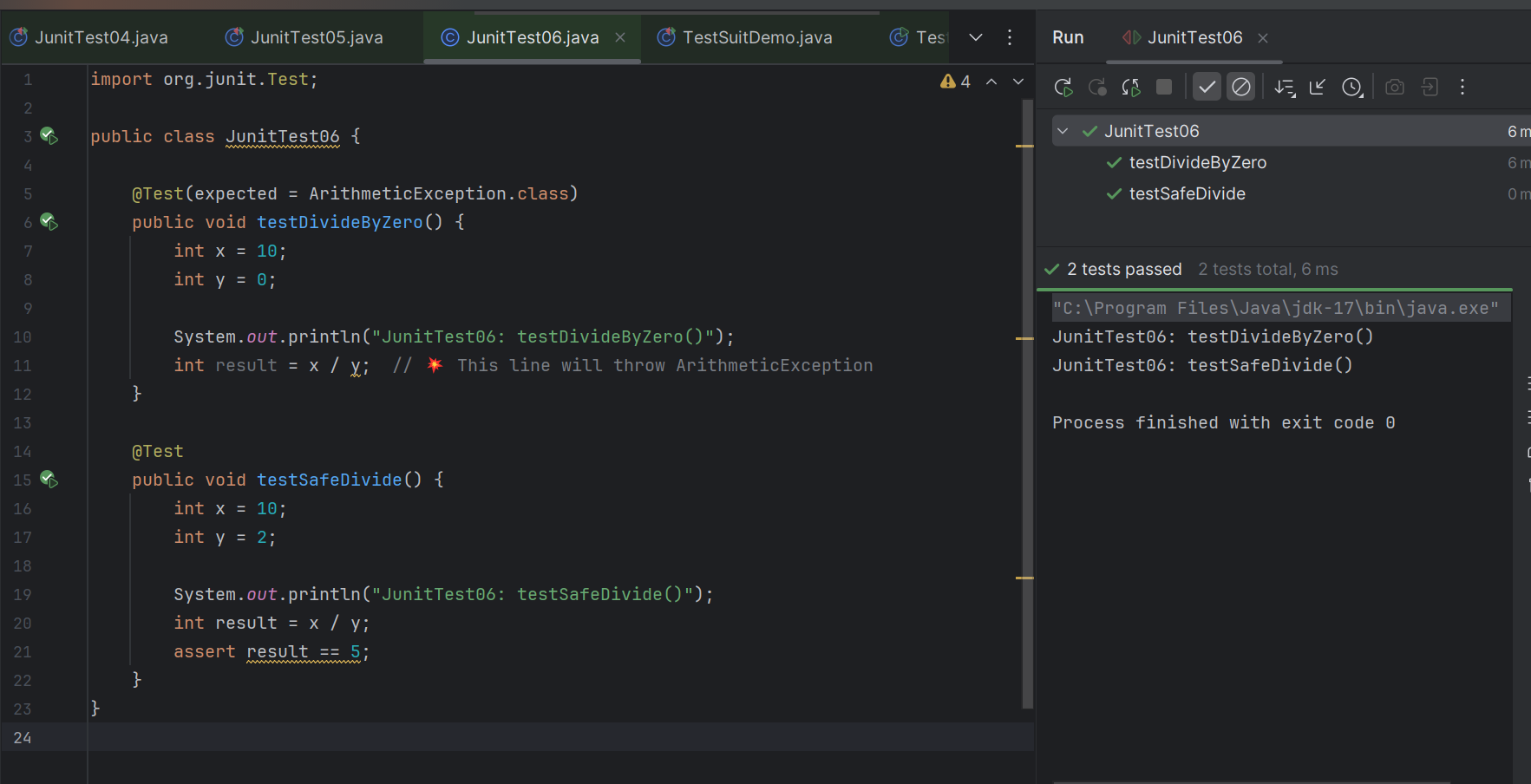


Task 05:

Exception test

In one of your tests check for arithmetic exceptions and see if it works..

@Test(expected = ArithmeticException.class)



Info box:

<https://junit.org/junit4/>

Junit4 archive.

<https://excalidraw.com/#json=5q1SN8_NqZBXvnbebkWY1,RyZBHBZmLIcetkhuIxVImg>

Updated canvas link as of 13.13 IST.

Codes for reference link → for uml codes

<https://drive.google.com/drive/folders/1LwhNov1s1-vHzF9GPAObLSnP9kAvipmw?usp=sharing>

Plz go through uml diagrams in page nos:

Docs 04 OOAD UML in doc to study

8 to 16

12

18 to 39

57 to 58

97

118

123

136

154 to 160

Plantuml site juz found .. plz check for reference

**done**

<https://plantuml.com/use-case-diagram>