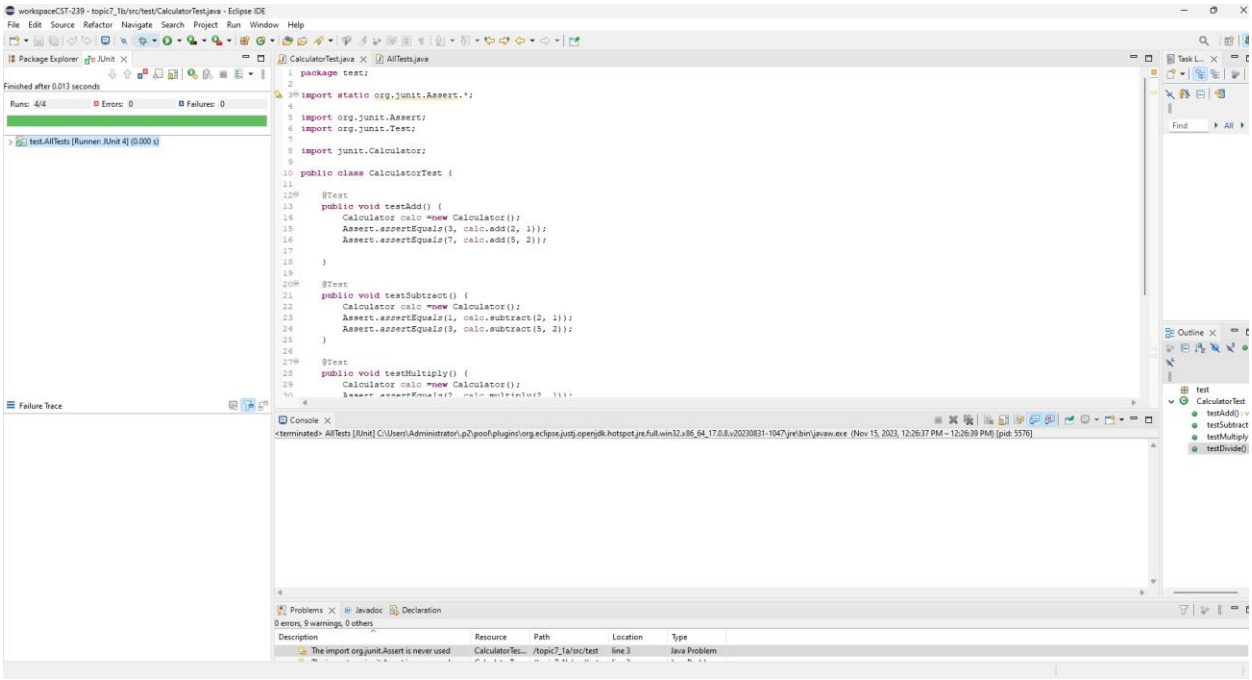
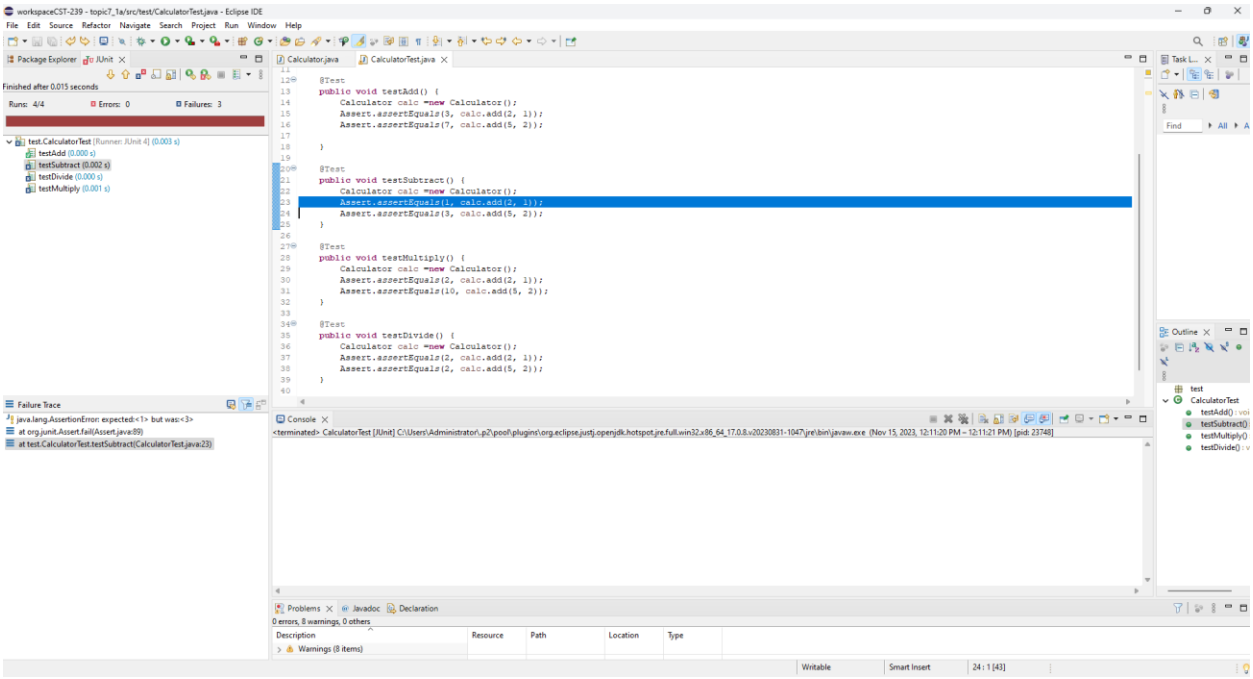
Activity 7

Topic 7-1a

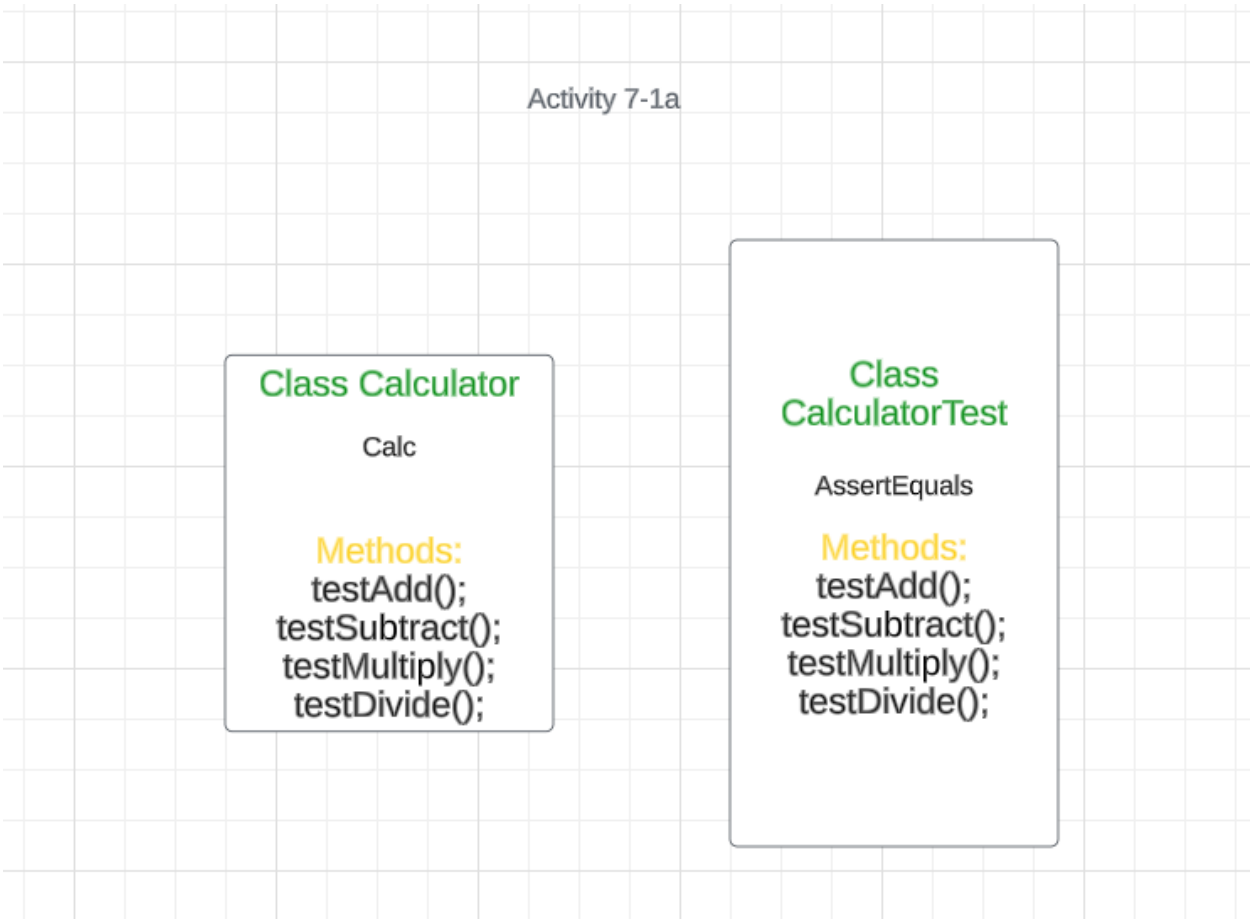
Passed cases:



Failed cases:

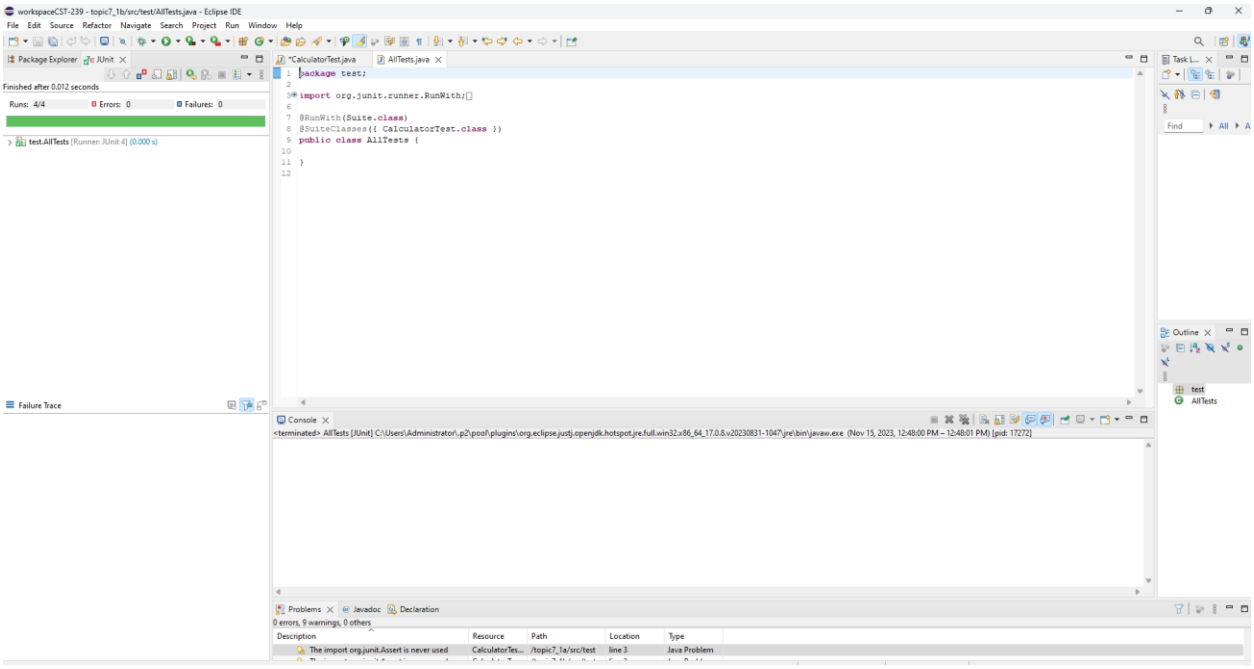


UML:


Activity 7-1a
Class Calculator
Calc
Methods:
testAdd();
testSubtract();
testMultiply();
testDivide();
Class CalculatorTest
AssertEquals
Methods:
testAdd();
testSubtract();
testMultiply();
testDivide();

Topic 7-1b

Screenshot:



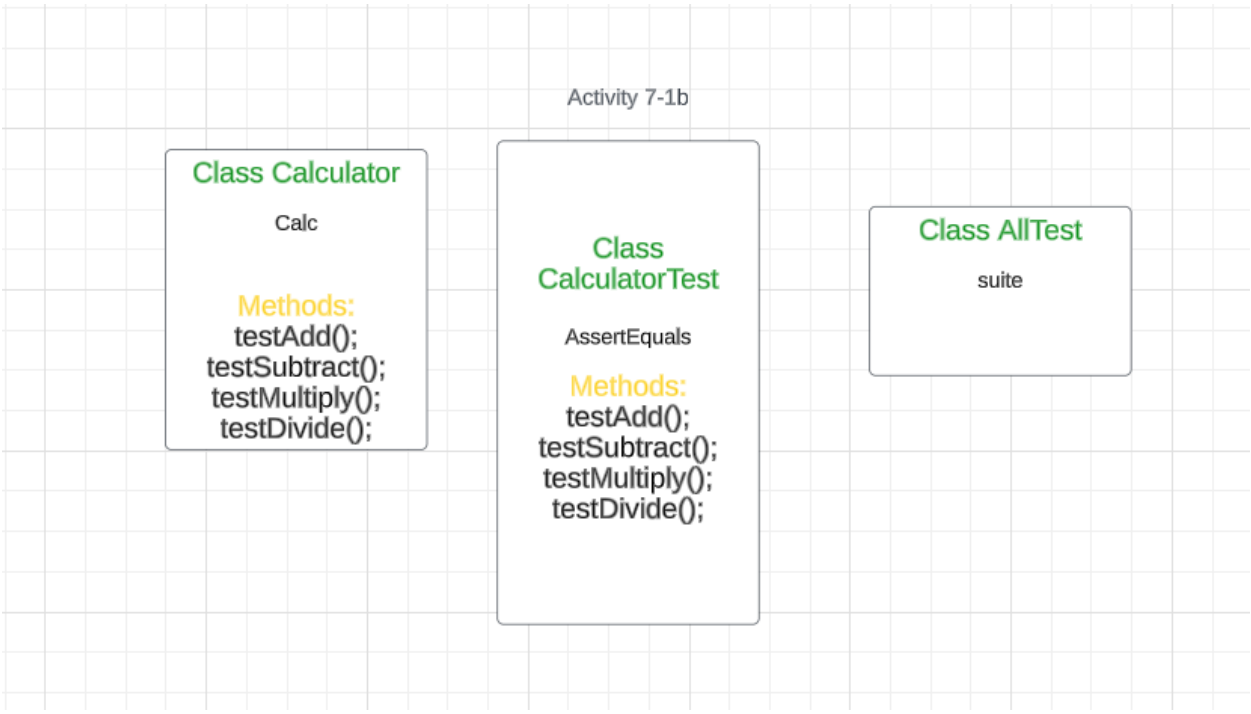
Write up:

5a. There was 4 test cases that we wrote in the test calculator. The error conditions are add, subtract, divide and multiply where assert cases were used. Assert being a good test case already, another method would be assert.same. There are so many methods that java provides that can be used in order to test cases.

5b. White testing box is basically the testing of the internal code and internal structure. While black box is the opposite, where testing is happening without the access of the internal code.

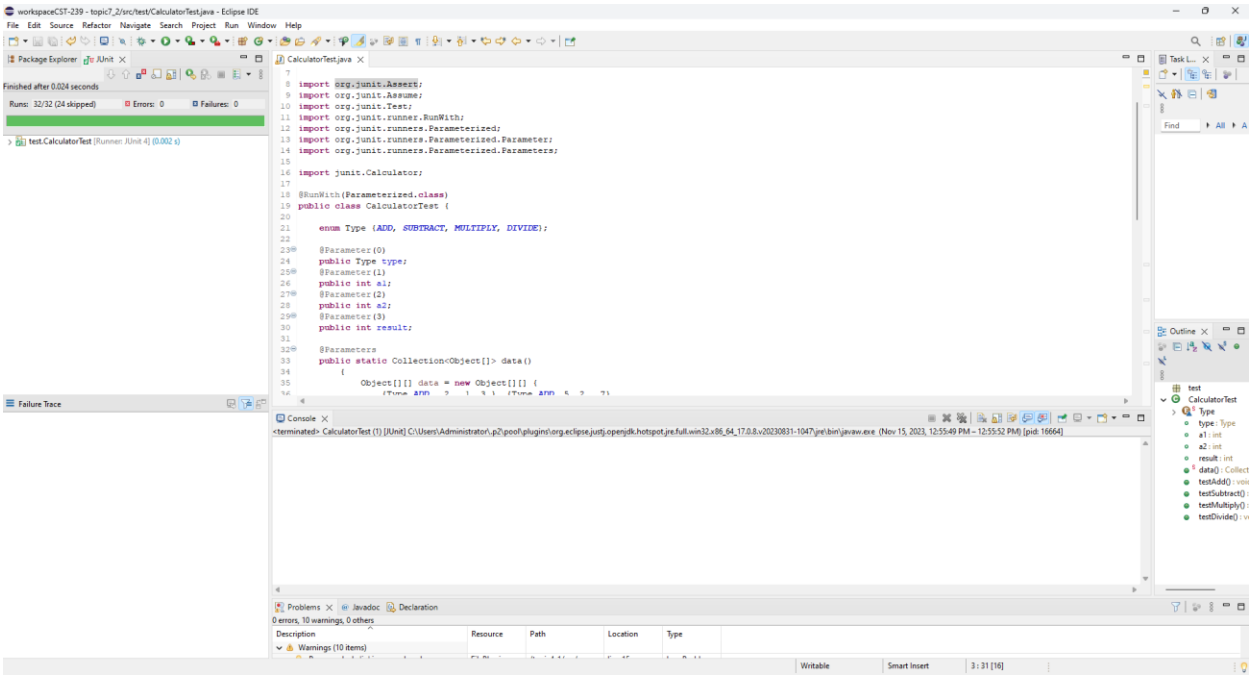
5c. Test suites are used to organize and do a certain amount of tests to make sure everything works and passes. We use it when we need to manage, excite and report the test results in an efficient way.

UML:


Activity 7-1b
Class Calculator
Calc
Methods:
testAdd();
testSubtract();
testMultiply();
testDivide();
Class CalculatorTest
AssertEquals
Methods:
testAdd();
testSubtract();
testMultiply();
testDivide();
Class AllTest
suite

Topic 7-2

Screenshot:

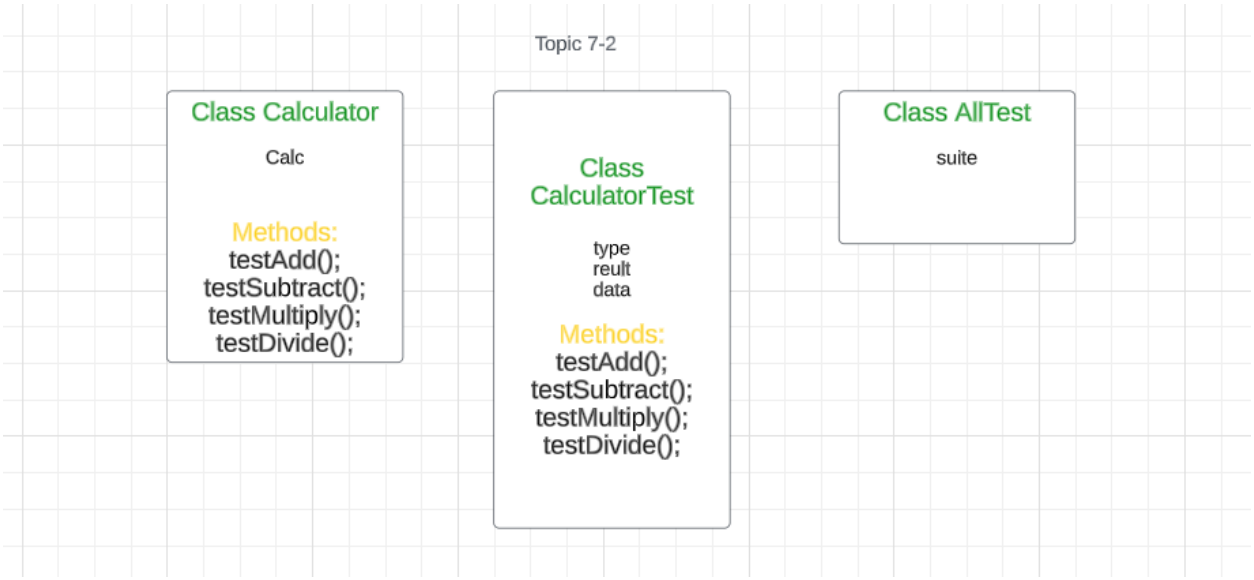


Write up:

2a. Parameterization gives us the ability to test the code in a systematically way, where we could make a single test case execute multiple times with multiple different data. So it is way more efficient to do it that way. In a nut shell, we can do testing with different parameters multiple time using the same test cases.

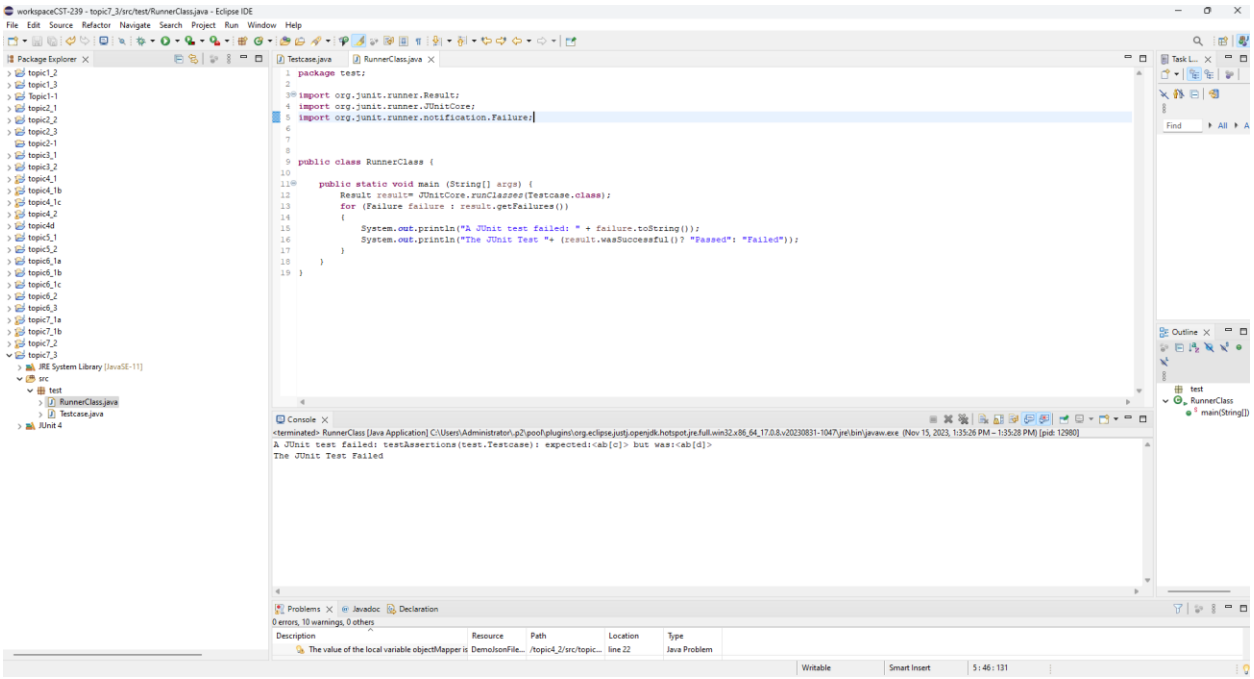
UML:

Topic 7-2


Class Calculator
Calc
Methods:
testAdd();
testSubtract();
testMultiply();
testDivide();
Class CalculatorTest
type
reult
data
Methods:
testAdd();
testSubtract();
testMultiply();
testDivide();
Class AllTest
suite

Topic 7-3

Screenshot:



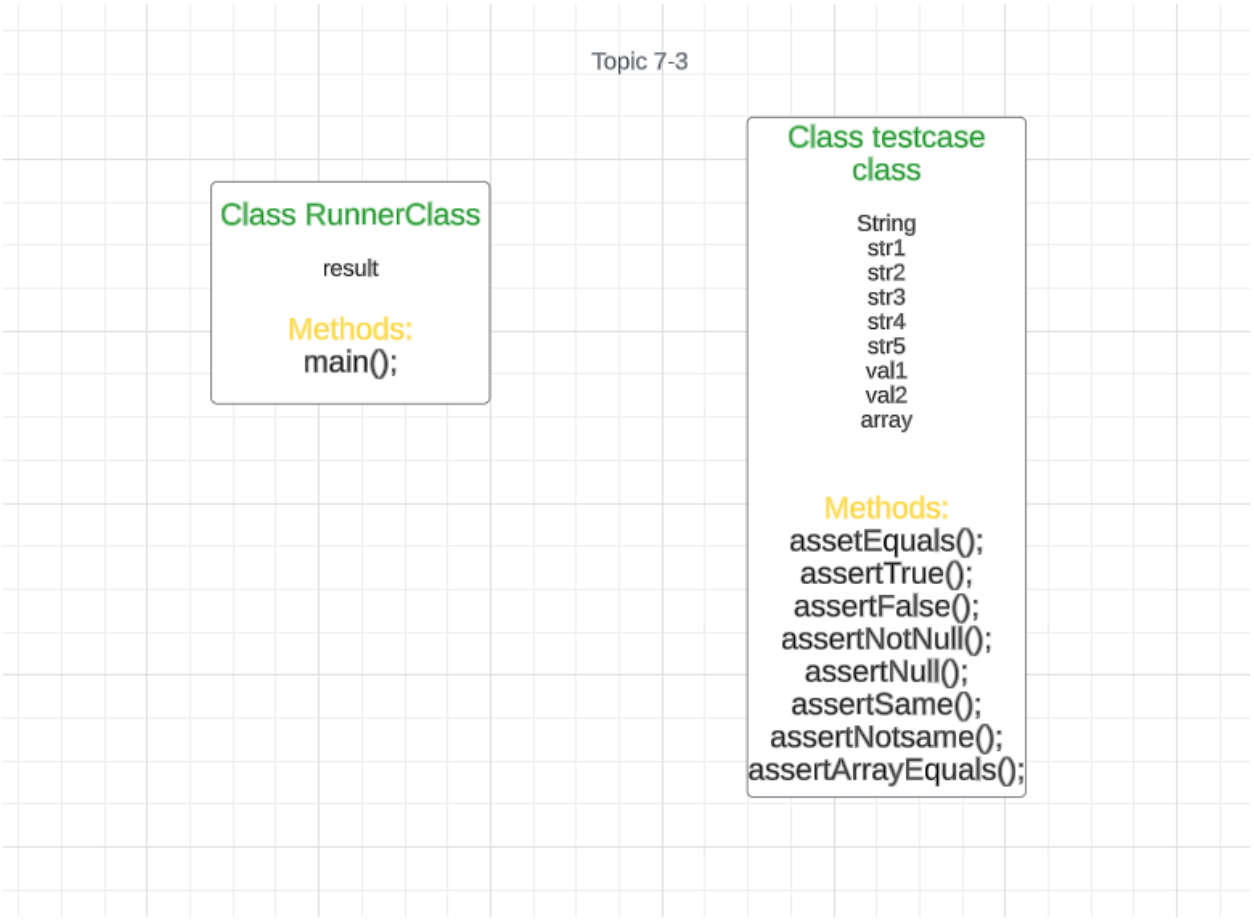
Write up:

6a. In order for us to test for the exceptions failures, we have to set up the expected failure case first, we then call on the code that throws the exception. We finally call on the assert that exception was thrown.

6b. There are many challenges to test for all the error conditions and exceptions. An example may include some sort of code that runs many times and gives many solutions so testing for that is one of the hardest tests.

UML:

Topic 7-3


Class RunnerClass
result
Methods:
main();
Class testcase class
String
str1
str2
str3
str4
str5
val1
val2
array
Methods:
assetEquals();
assertTrue();
assertFalse();
assertNotNull();
assertNull();
assertSame();
assertNotsame();
assertArrayEquals();