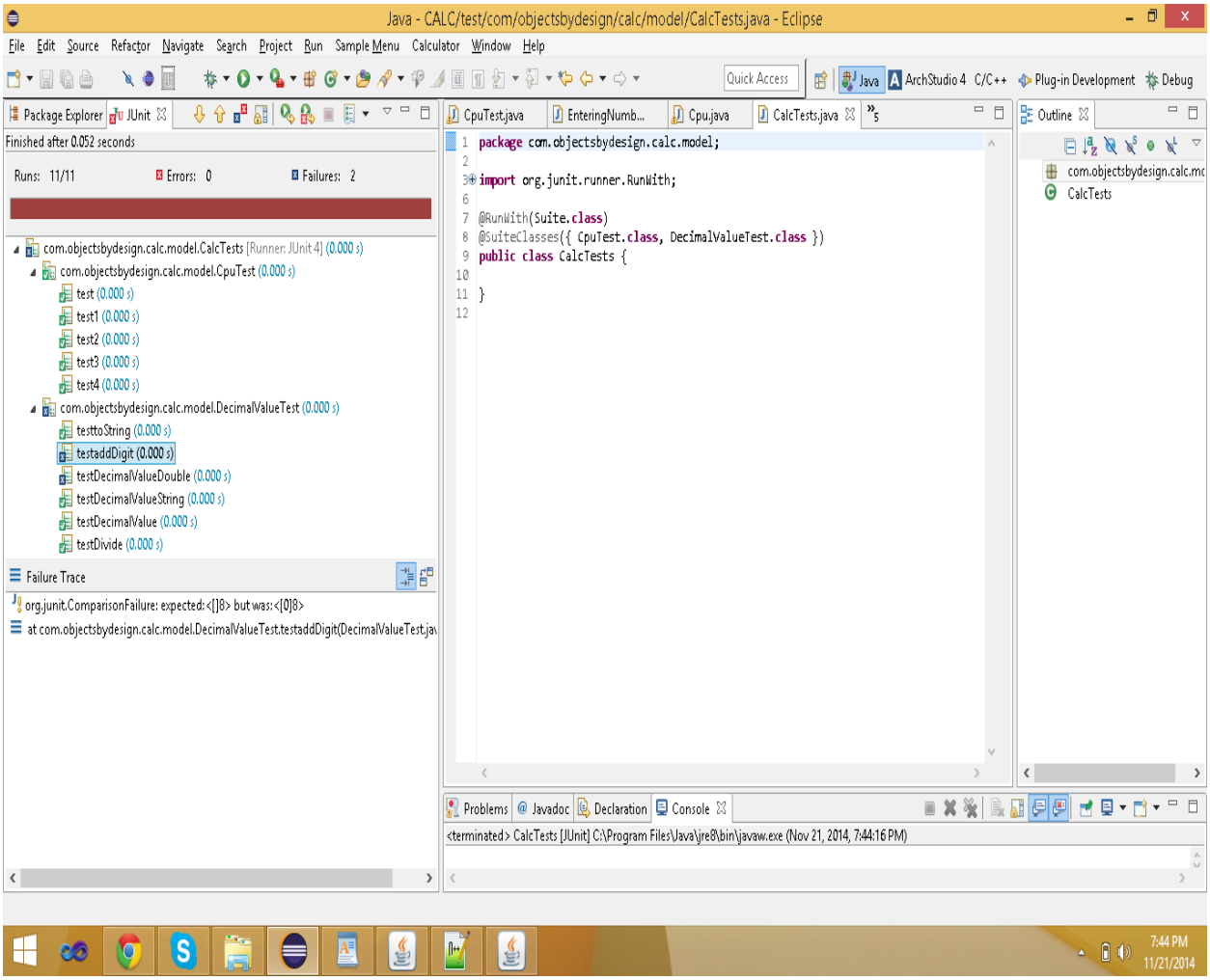
Java - CALC/test/com/objectsbydesign/calc/model/CalcTests.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Sample Menu Calculator Window Help
Quick Access
Java ArchStudio 4 C/C++ Plug-in Development Debug
Package Explorer JUnit
Finished after 0.052 seconds
Runs: 11/11
Errors: 0
Failures: 2
com.objectsbydesign.calc.model.CalcTests [Runner: JUnit 4] (0.000 s)
com.objectsbydesign.calc.model.CpuTest (0.000 s)
test (0.000 s)
test1 (0.000 s)
test2 (0.000 s)
test3 (0.000 s)
test4 (0.000 s)
com.objectsbydesign.calc.model.DecimalValueTest (0.000 s)
testtoString (0.000 s)
testaddDigit (0.000 s)
testDecimalValueDouble (0.000 s)
testDecimalValueString (0.000 s)
testDecimalValue (0.000 s)
testDivide (0.000 s)
Failure Trace
org.junit.ComparisonFailure: expected:<[]8> but was:<[0]8>
at com.objectsbydesign.calc.model.DecimalValueTest.testaddDigit(DecimalValueTest.ja
CpuTest.java
EnteringNumb...
Cpu.java
CalcTests.java
1 package com.objectsbydesign.calc.model;
2
3 import org.junit.runner.RunWith;
6
7 @RunWith(Suite.class)
8 @SuiteClasses({ CpuTest.class, DecimalValueTest.class })
9 public class CalcTests {
10
11 }
12
Outline
com.objectsbydesign.calc.mc
CalcTests
Problems Javadoc Declaration Console
<terminated> CalcTests [JUnit] C:\Program Files\Java\jre8\bin\javaw.exe (Nov 21, 2014, 7:44:16 PM)
7:44 PM
11/21/2014

1)

Bugs:

1. In Add digit it should remove leading zeros, in the case number 0 and digit 8, the output expected is 8, but actual is 08.

   As, it is said in the function it should remove all leading zeros, according to this it is failing.

   Here in the adddigit code

```
if (!decimalSet && number.equals("0")) {
        // only one zero allowed
        if (digit.equals("0"))
           return number;
        // trim leading zero
        else
           number = "0";
              }
```

   When not decimal and number equals 0 and digit is not 0 in that case number="" should be done to trim leading zero. But here it is left as number="0". This should be done to fix the bug.

2. In Decimal Value constructor, the Decimal Value constructor taking double value as input is failing, because when I pass BigDecimal 100.0 (done scaling as 10 and roundup for the Big Decimal Value) but the result in the Decimal Value Object value is not matching with the sent value. Here expected is coming as 100. 0000000000and actual is 100.0000000000. But during comparison passed value(100. 0000000000) is changing to 100.0 and because of precision loss it is failing.
   Actually, both should be same, because in DecimalValue code also it is scaled to 10, in testing code also it is scaled to 10. But not working. As, it is not coming same, so considered as bug.

2) Desiging of Test Cases:

CpuTest:

In this tested the state logic of Cpu.

We have 4 states here:

1. WaitingForInputState
2. WaitingForNumberState
3. WaitingForOperationState
4. EnteringNumberState
   Logic here is initially cpu will be in WaitingForInputState, when some number is entered it will move to EnteringNumberState. If some operation is entered it does not have lookup then it enters into WaitingForInputState and if It has lookup it will enter into WaitingForNumberState.
   Similarly analyzed for remaining states.

   Next after analyzing transitions, started with for each state done like this:
   Giving enter digit/ enter operation and how it is transitioning. And checked that state with the expected state after analyzing.
   Ex: CPU is in WaitingForInputState, and entered number 2 then it will be moved to Entering Number State.
   Similarly like this given different inputs and tested. Continued same with all other test cases.

   In this tested in first 4 methods from each state to all other states, the logic is working fine or not.
   That means for example WaitingForInputState. From this state all possible states are tested by giving different inputs.

   2nd: Tested the complex expression and testing after each input whether it is in correct state or not.
   Ex: 22-1*4=
   And also tested:
   Entered 2 then done MemoryRecall, this invokes the state WaitingForOperationState,as MemoeryRecall is treated as Value here and changing to that state.

   In this all test cases passed.

Decimal ValueTest:

For Constructors tested whether object is created successfully or not using assertNotNull.

Function	Input	Expected	Actual	Test Case Result
testDecimalValueBigDecimal	BigDecimal(0)	BigDecimal(0)	BigDecimal(0)	Pass
	BigDecimal(0)	BigDecimal(0)	BigDecimal(0)	Pass
	BigDecimal(0)	BigDecimal(0)	BigDecimal(0)	Pass
	BigDecimal(0)	BigDecimal(0)	BigDecimal(0)	Pass
testDecimalValueDouble	100.0	100.0000000000	100.0000000000	Fai(Here even input is given same as 100.0000000000 but when comparing it is changed to 100.0 and the test case is failing.
testDecimalValueString	“100.00”	BigDecimal(“100.00”)	BigDecimal(“100.00”)	
	“Bharat”	Exception should occur	Expection coming	Pass
testDivide	Passed “5” and “10”	2.0000000000	2.0000000000	Pass
	Passed 2 and 1	0.5000000000	0.5000000000	Pass
	Passed 1 and 0	Exception	Exception	Pass
testAddDigit	0,0	0	0	Pass
	3.2,6	3.26	3.26	Pass
	2.0,0	2.00	2.00	Pass
	0,8	8	08	Fail(Leading zeros not

				removed) Above top solution is proposed.
testtoString	10	10.	10.	Pass
	10.00	10.	10.	Pass

Note: As DecimalValue constructor taking double value failed, due to precission loss , the same when divide happens also will fail.

So, in divide test cases, the expected is tested with creating BigDecimal through string.