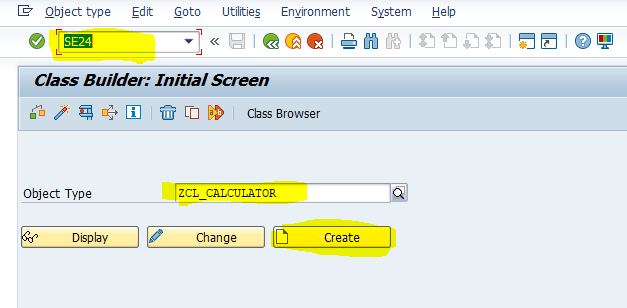
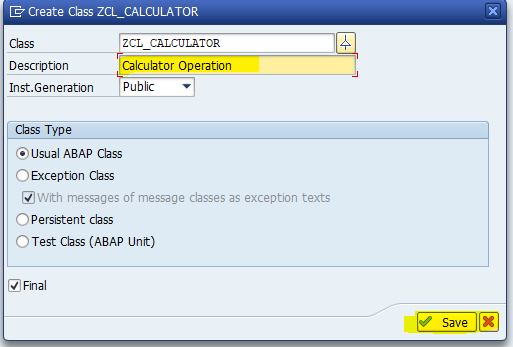
**ABAP Unit Testing**

**Top-Down Approach**

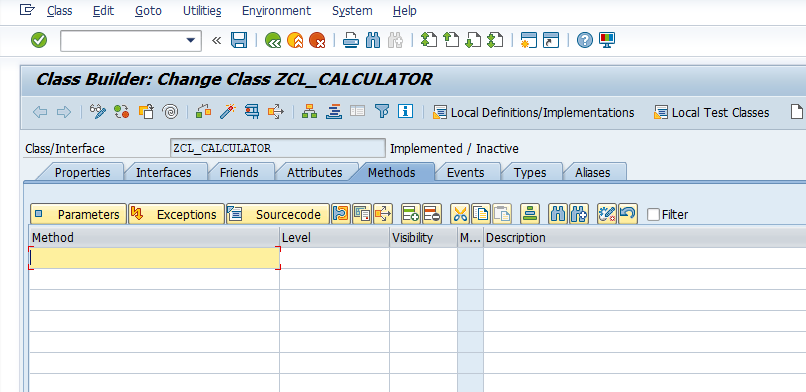
Execute transaction code **SE24** and Enter global class name ‘**ZCL\_CALCULATOR’** and click on create button as below screen



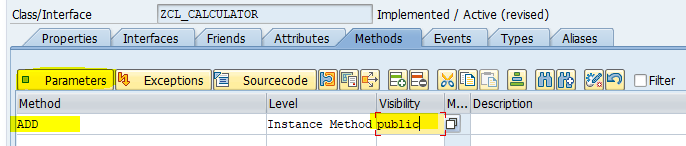
Enter description and click on save button



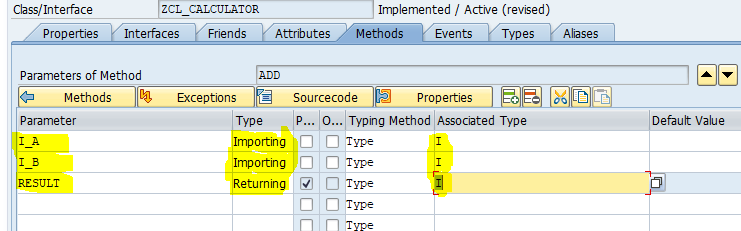
Then class look like as below screen



Enter Method ‘ADD’ as below

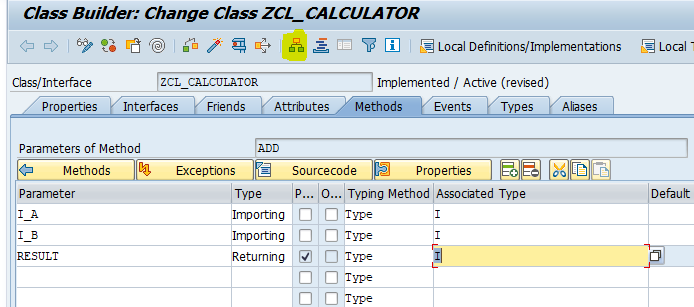


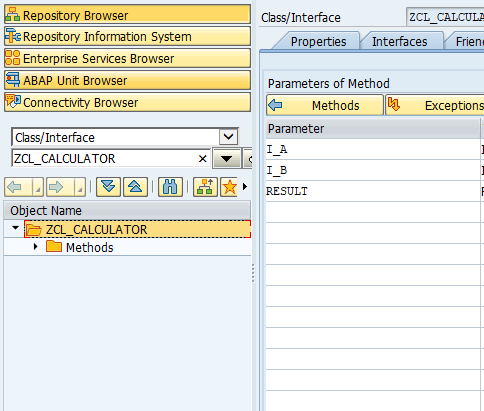
Click on parameter tab to enter importing and exporting parameter to method as below



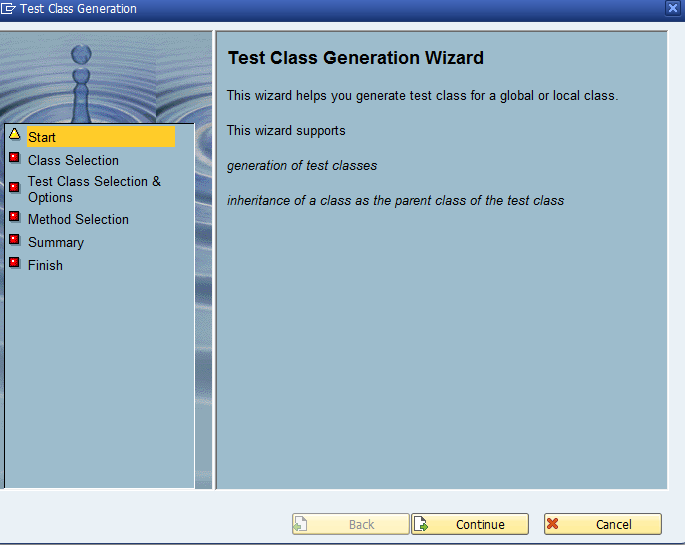
Click on active to activate class

Click on display object list button as below screen

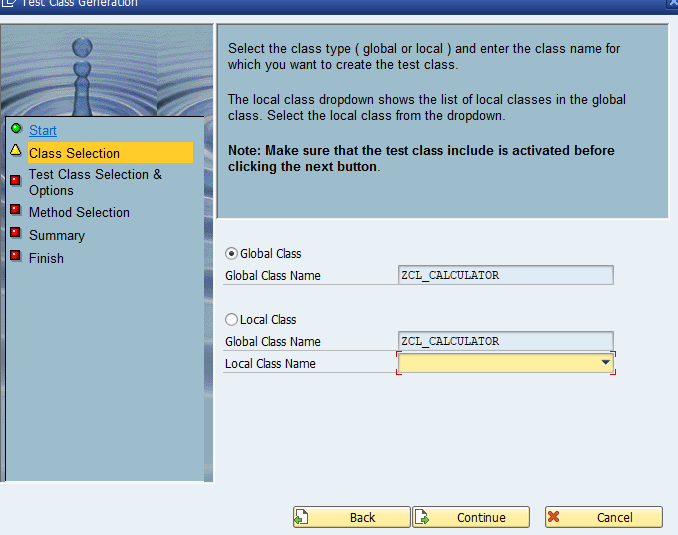




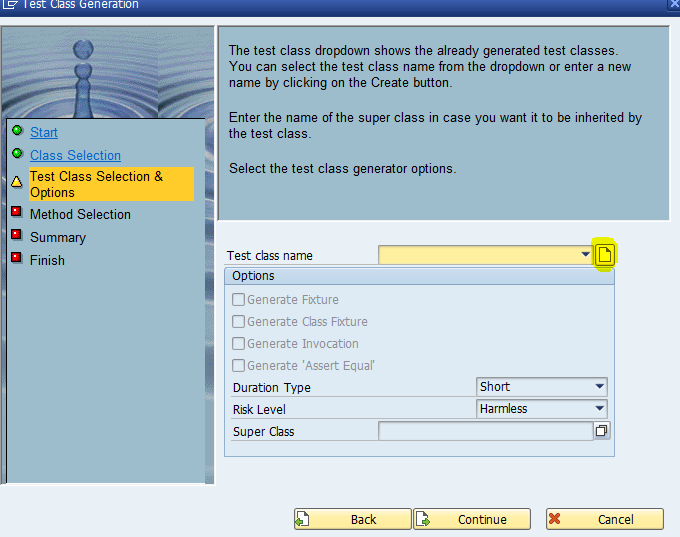
Right click on class ‘**ZCL\_CALCULATOR’** and choose path Create->Generate Test class to create test class then below screen will appear



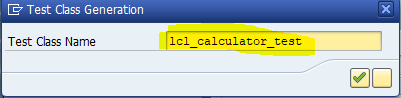
Click on continue button



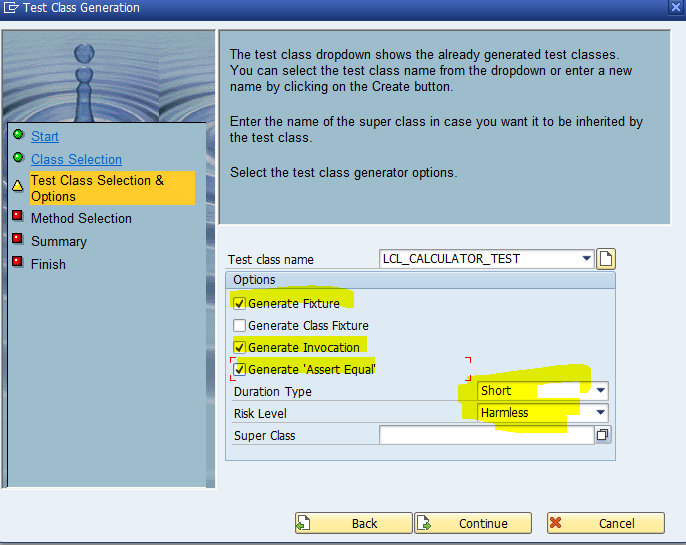
Don’t change anything and click on continue button



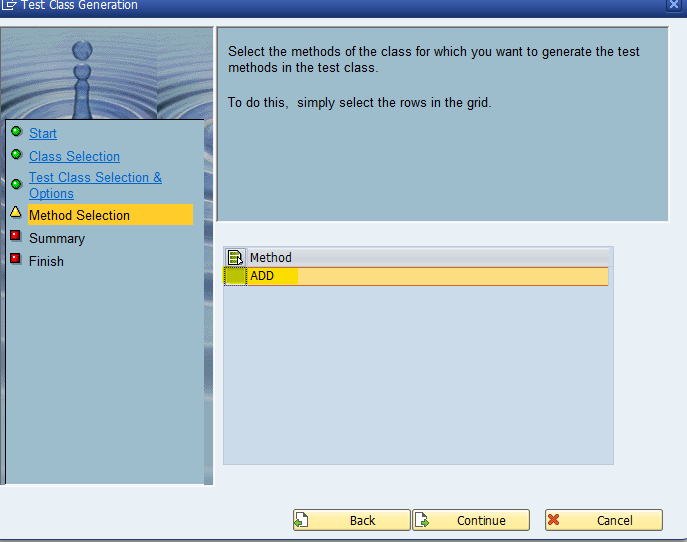
Click on create button to enter local test class name



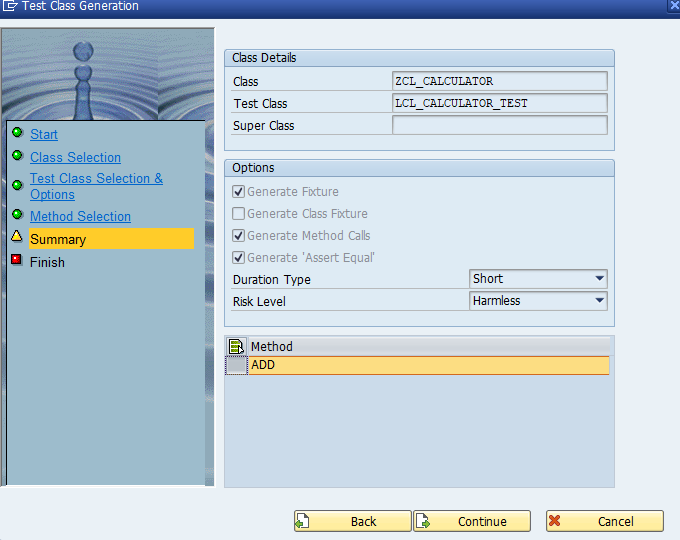
Click on continue button



Select Generate Fixture, innovation and ‘**Assert\_Equal’** checkbox and choose ‘**short**’ as duration type and ‘**Harmless**’ as Risk level and click on continue button



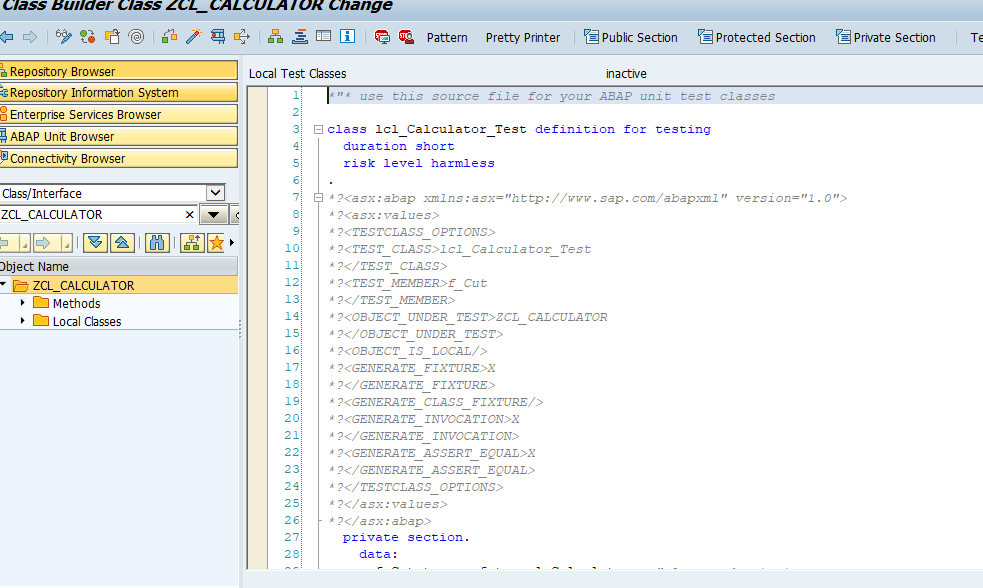
Select method ‘**ADD**’ and click on continue button then next screen will show as below with Global class, Local class, fixture method, duration and risk level and click on continue button



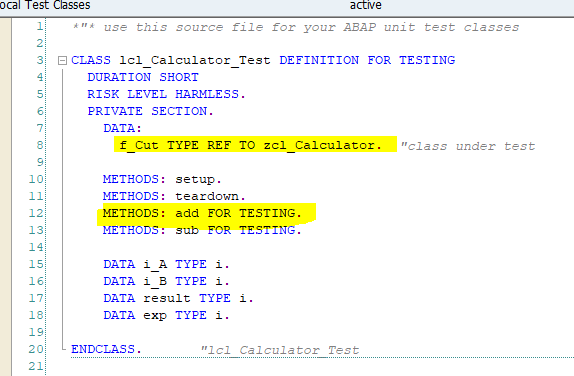
Click on complete button



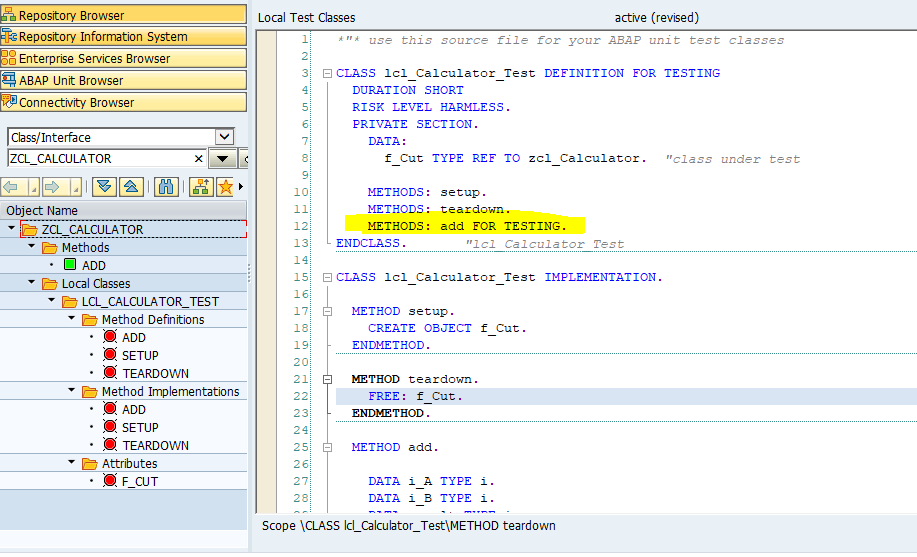
Then local test class will generate with required method as below screen



System generated test method ‘**ADD**’ in test class as below screenshot



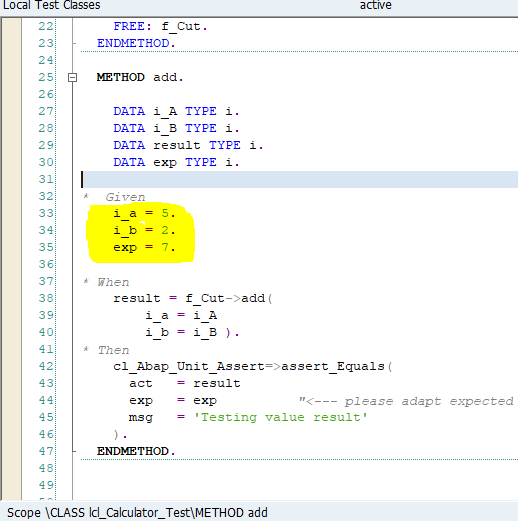
Object “**F\_Cut**” is used to access method of product code



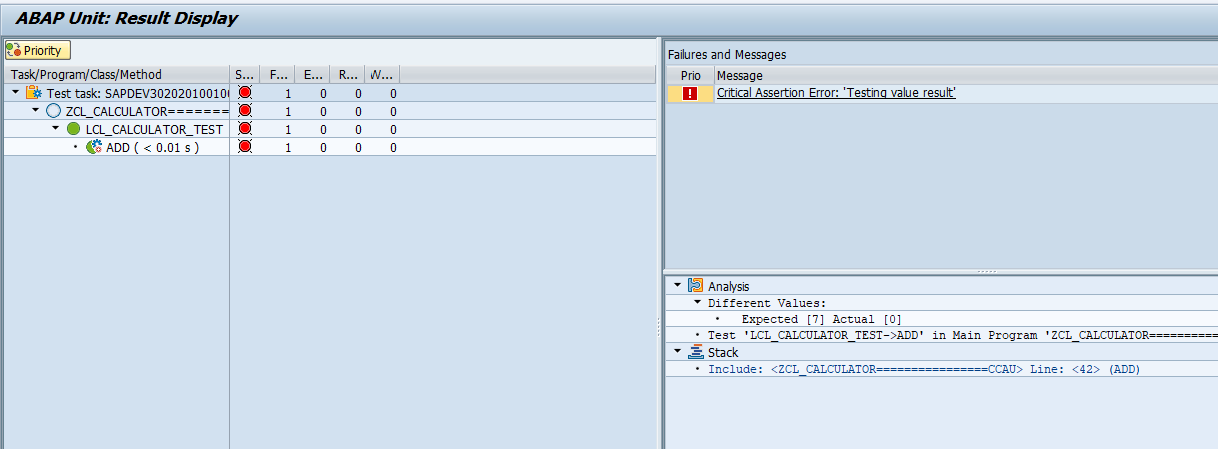
Pass the Given value as

I\_A = 5, I\_B = 2

Expected Result = 7

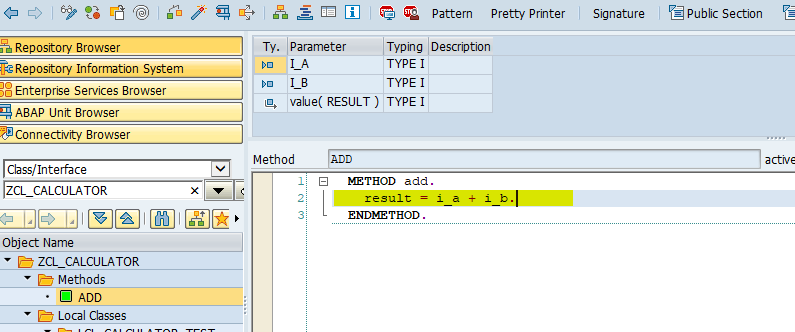


Execute test by shortcut ‘**Ctrl+Shift+F10**’ or Menu path ‘**Local test class->Execute->Unit Tests’**

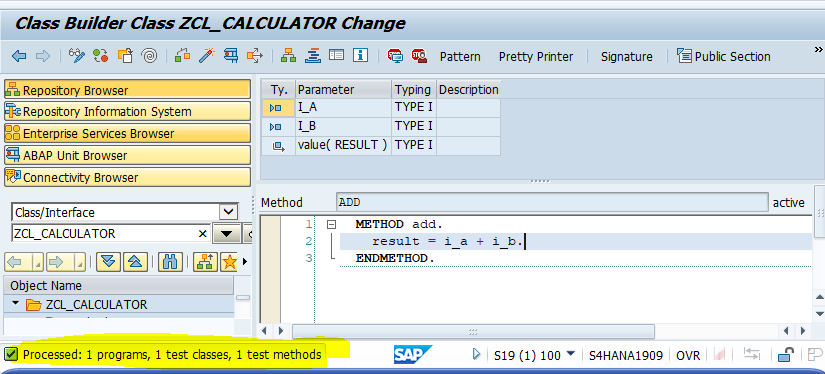


Unit test is fail because no logic inside the product method ‘**ADD**’

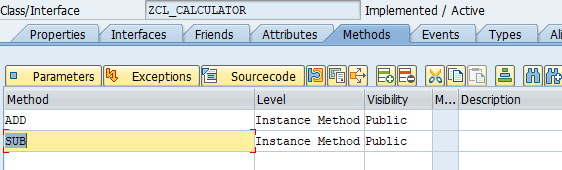
Refactor product method code to Pass test case as below

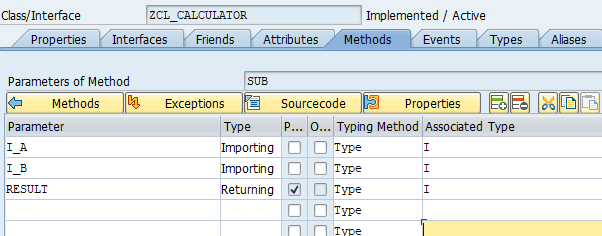


Activate class and run unit test then unit test case will pass

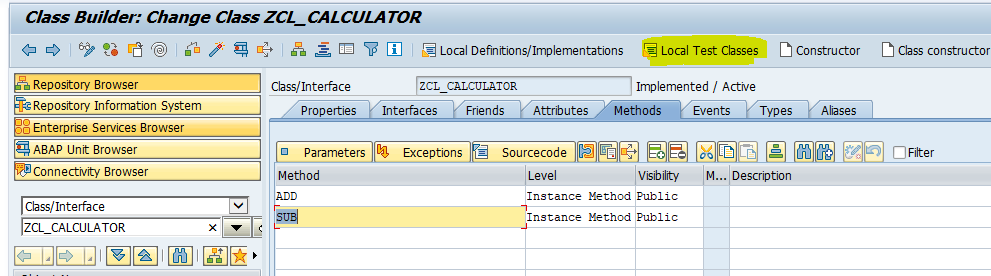


Add method ‘**SUB**’ for subtraction in product code class as below

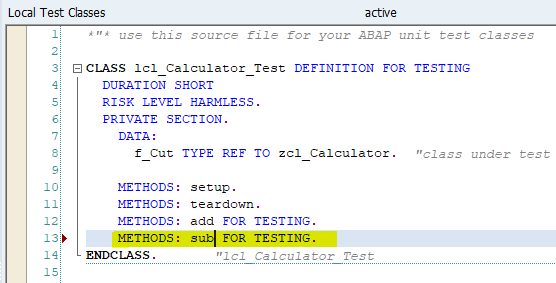




Click on ‘Local test class’ menu button to goes to test class



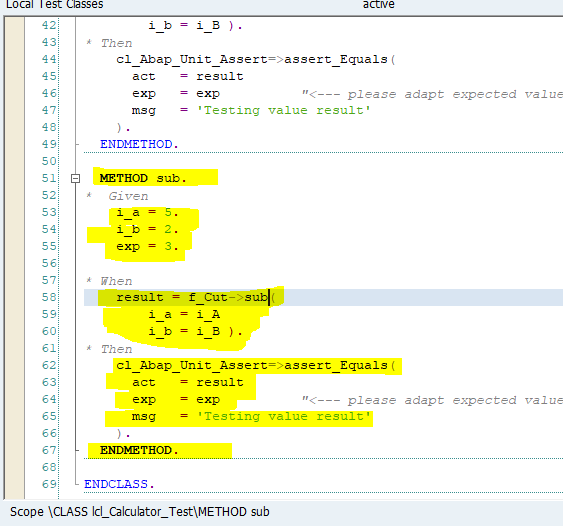
Add “**SUB**” test method in test class definition as below



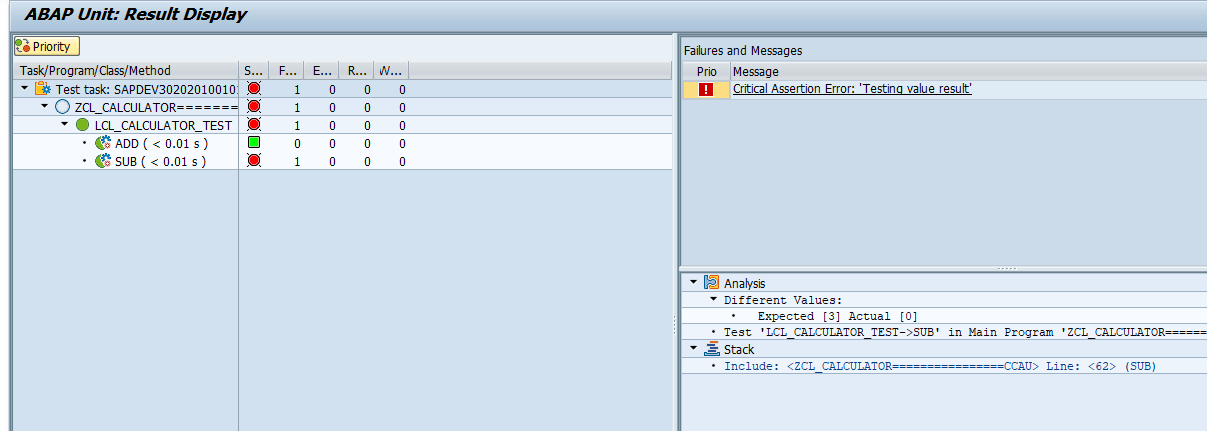
Implement ‘SUB’ method as below and pass Given value

I\_A = 5, I\_B = 2

Expected Result = 3

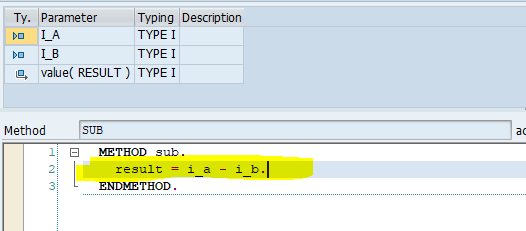


Execute test by shortcut ‘**Ctrl+Shift+F10’** or Menu path ‘**Local test class->Execute->Unit Tests’**

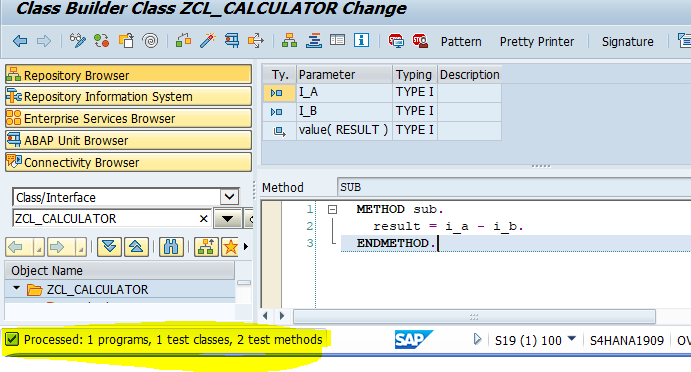


Test method ‘**SUB**’ fail because no product code written inside product method ‘**SUB**’

Refactor product method ‘**SUB**’ code to pass test case as below

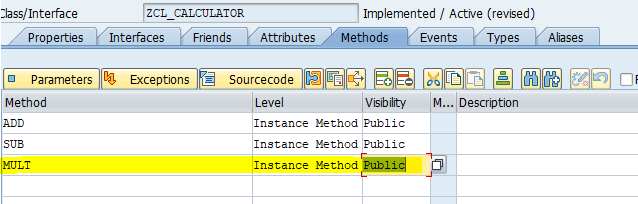


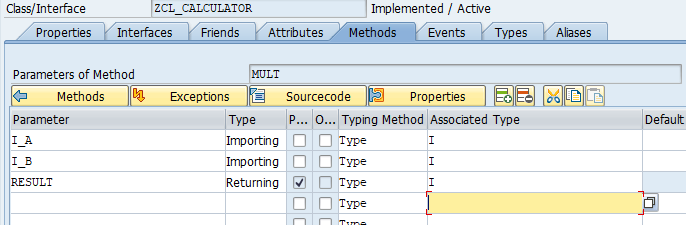
Execute unit test once again



Now two test methods passed

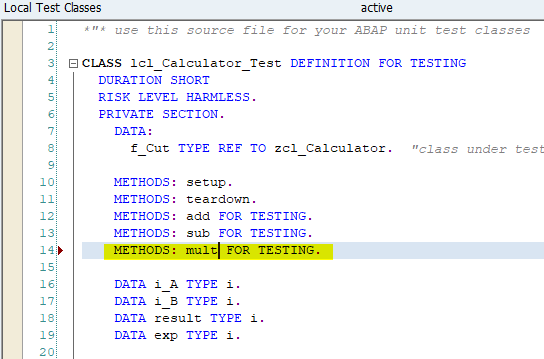
Add method ‘**MULT**’ in product code for multiplication operation as below





And activate class

Go to local test class and add ‘**MULT**’ test method in test class definition as below

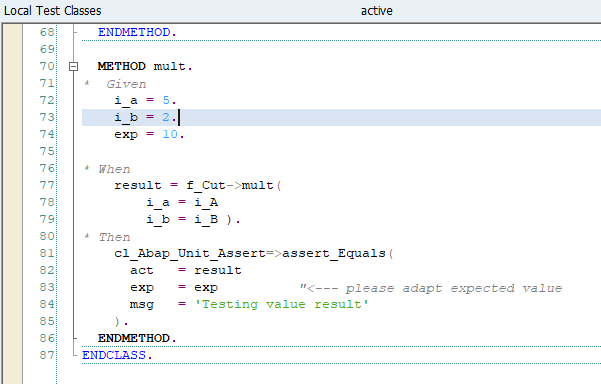


Implement test method ‘**MULT**’ as below

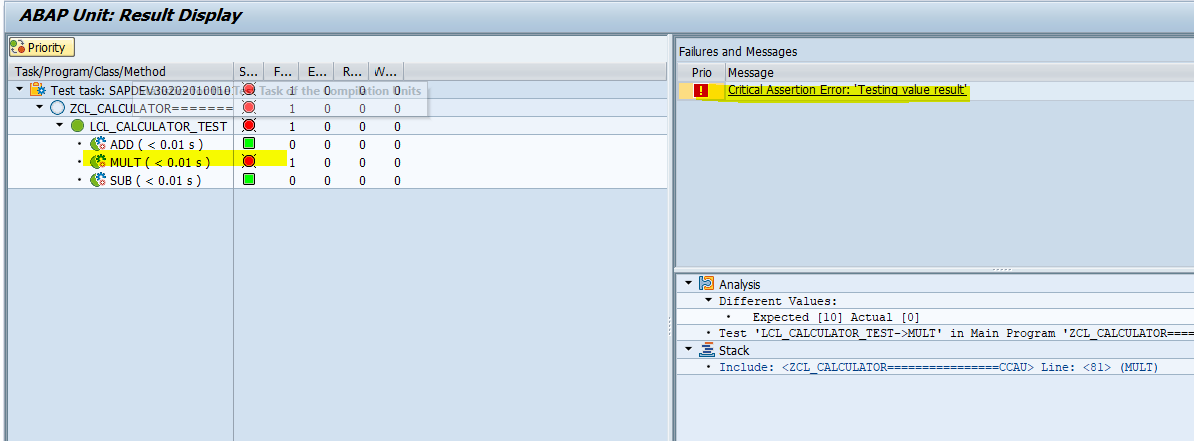
pass Given value

I\_A = 5, I\_B = 2

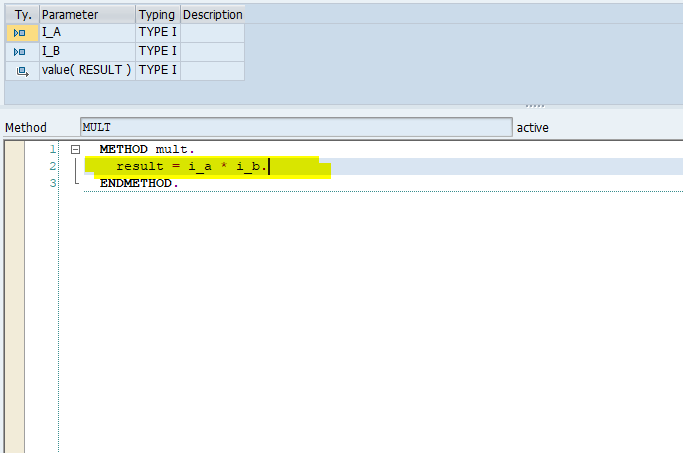
Expected Result = 10



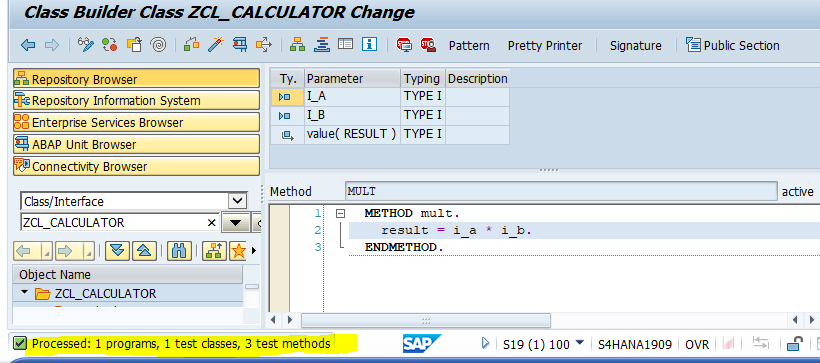
Execute test by shortcut ‘Ctrl+Shift+F10’ or Menu path ‘Local test class->Execute->Unit Tests’



Test method **‘MULT’** is fail because no product code exist. Refactor product code method to pass test method **‘MULT’** as below

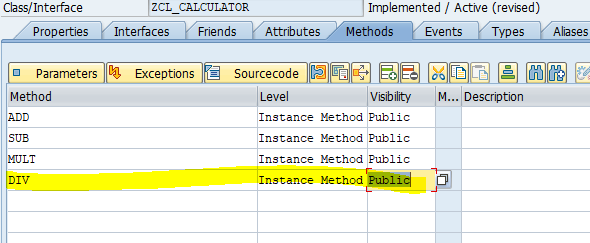


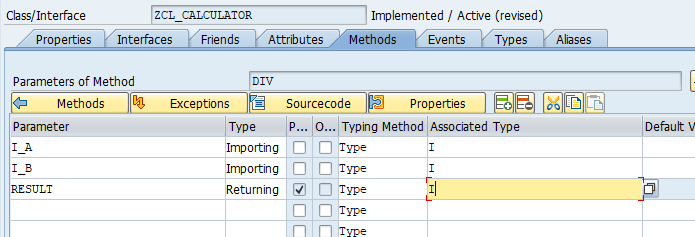
Execute unit test once again



Now test method ‘MULT’ is passed

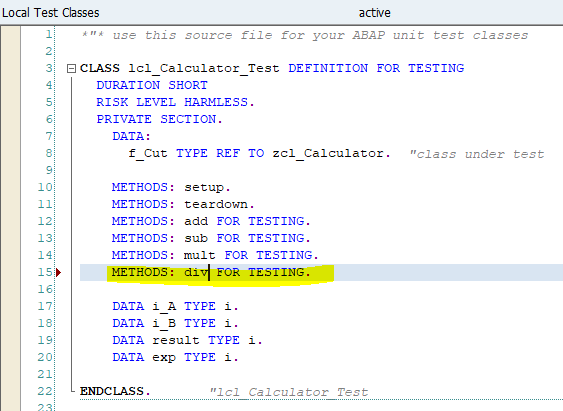
Add method **‘DIV’** in product code for multiplication operation as below





And activate class

Go to local test class and add ‘DIV’ test method in test class definition as below

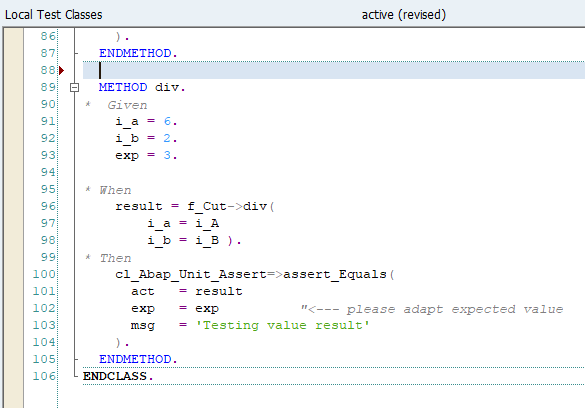


Implement test method ‘**DIV**’ as below

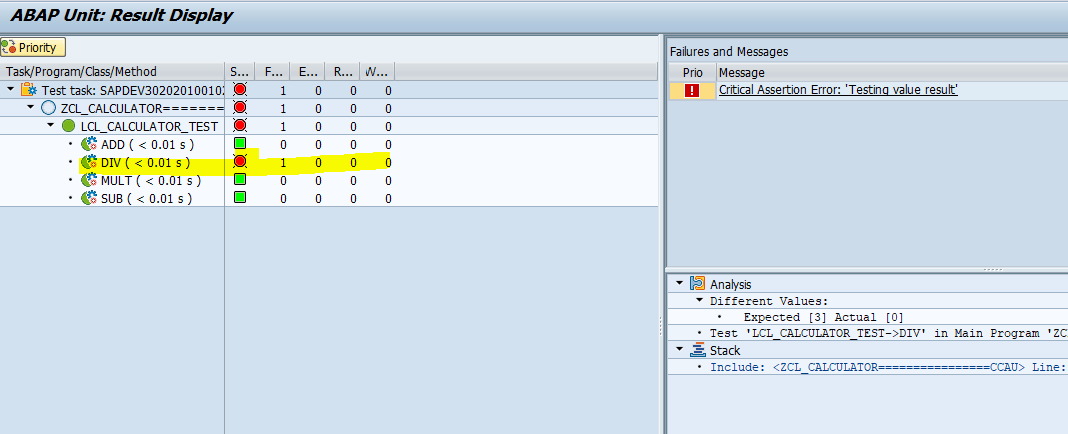
pass Given value

I\_A = 6, I\_B = 2

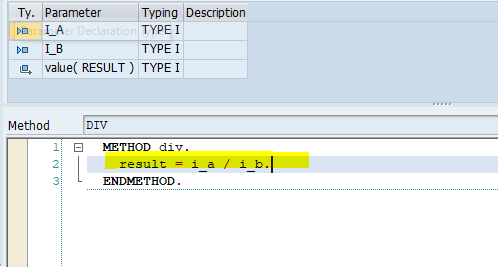
Expected Result = 3



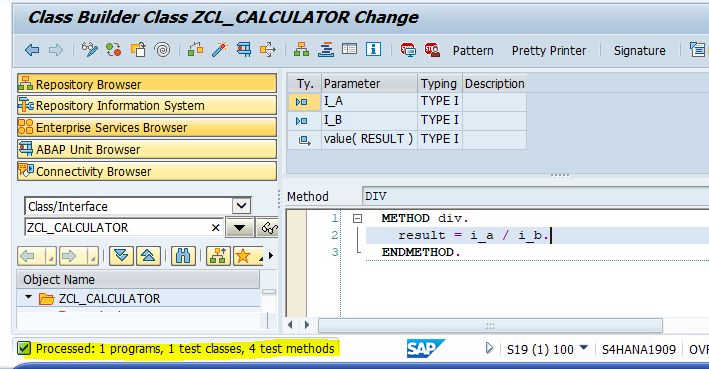
Execute test by shortcut ‘**Ctrl+Shift+F10’** or Menu path ‘**Local test class->Execute->Unit Tests’**



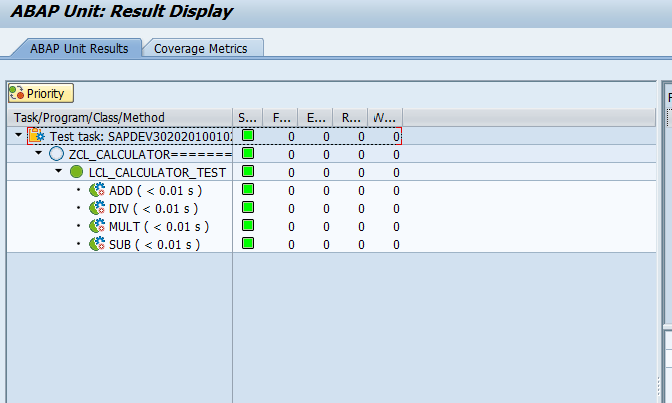
Test method ‘**DIV’** is fail because no product code exist. Refactor product code method to pass test method ‘**DIV**’ as below



Execute unit test once again

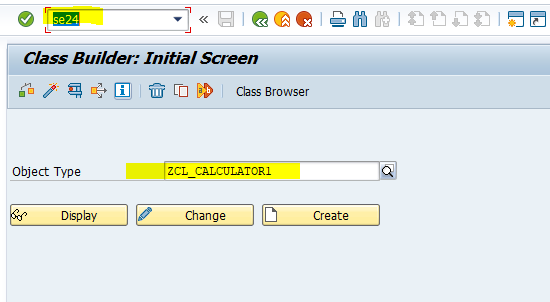


Now test method ‘**DIV**’ is passed

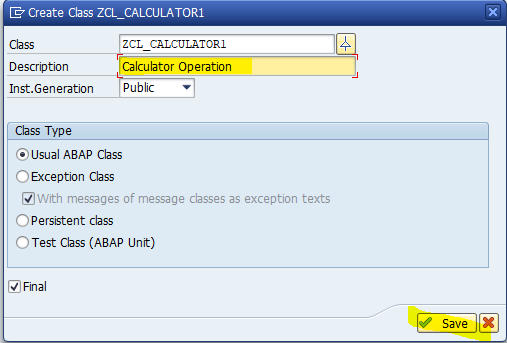


**Bottom-up Approach**

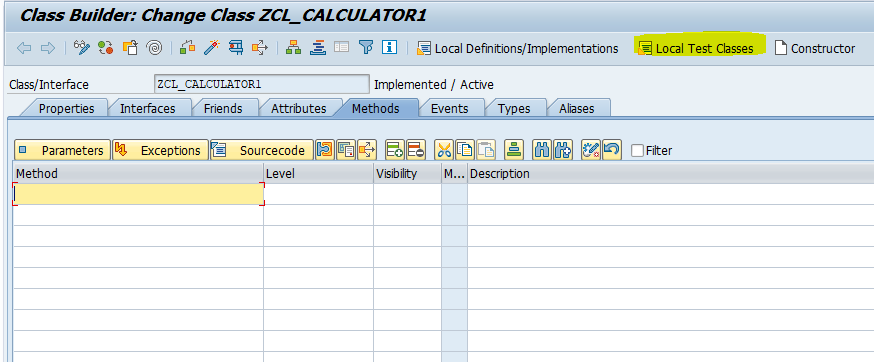
Execute transaction code SE24 and Enter class name ‘**ZCL\_CALCULATOR1**’ and click on create button to create global class



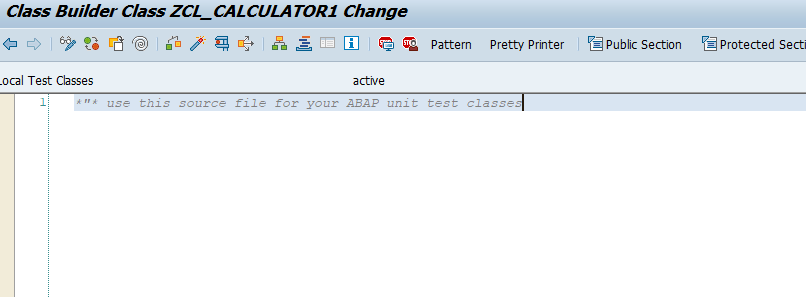
Enter class description and click on save button



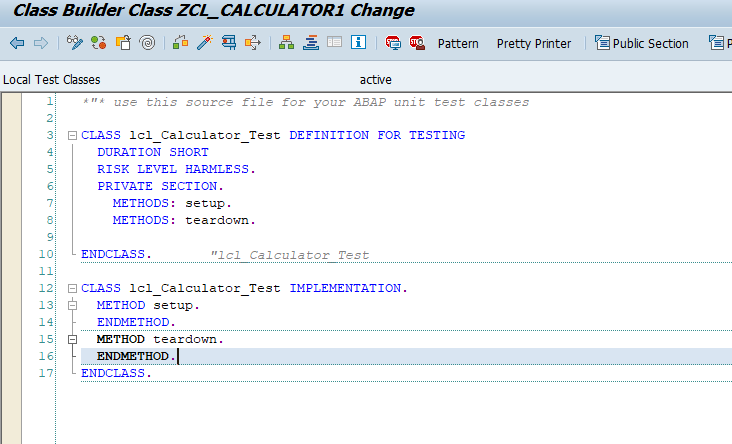
Then class look like as below



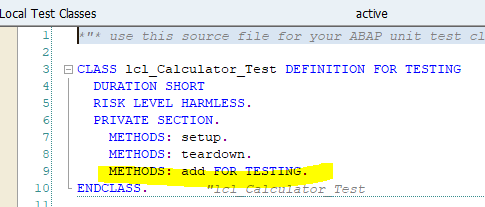
Click on Local Test Classes Menu button to write local test class

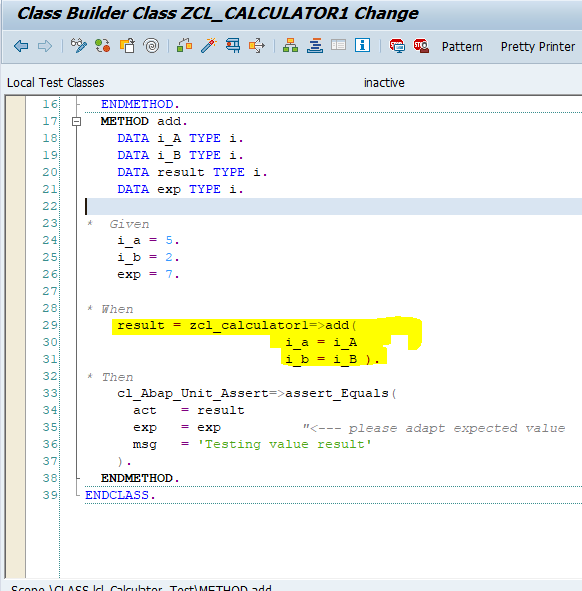


Write Local test class definition and implementation with fixture method as below

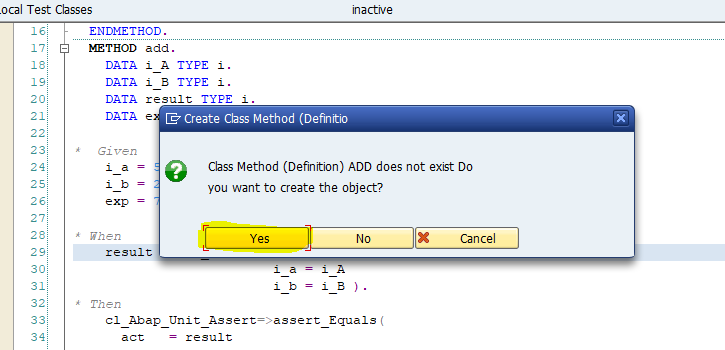


Add Test method ‘**ADD**’ in test class definition, implementation and pass Given data

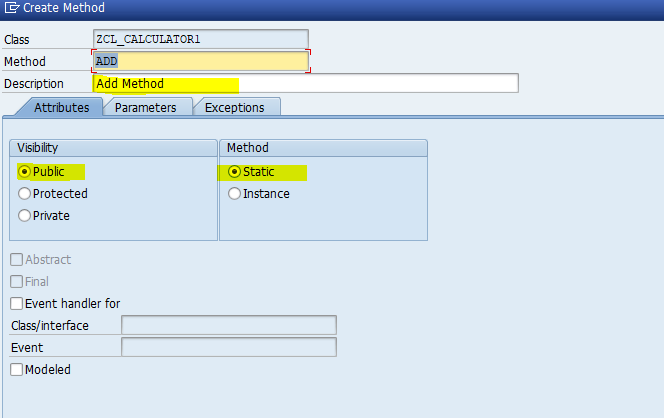




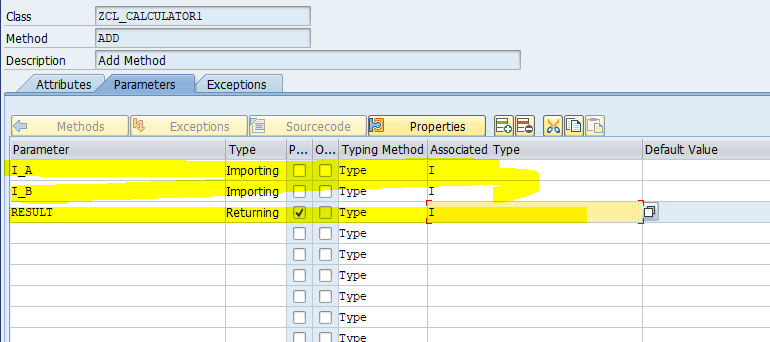
Double click on **ADD** method to create this method in product class



Click on Yes button

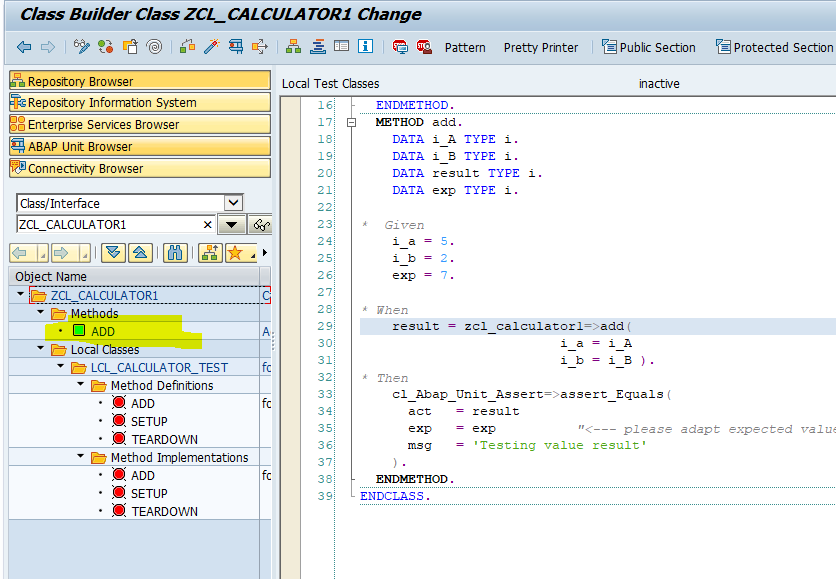
Enter Method description, Visibility as public and method as static

Enter parameter in parameter tab



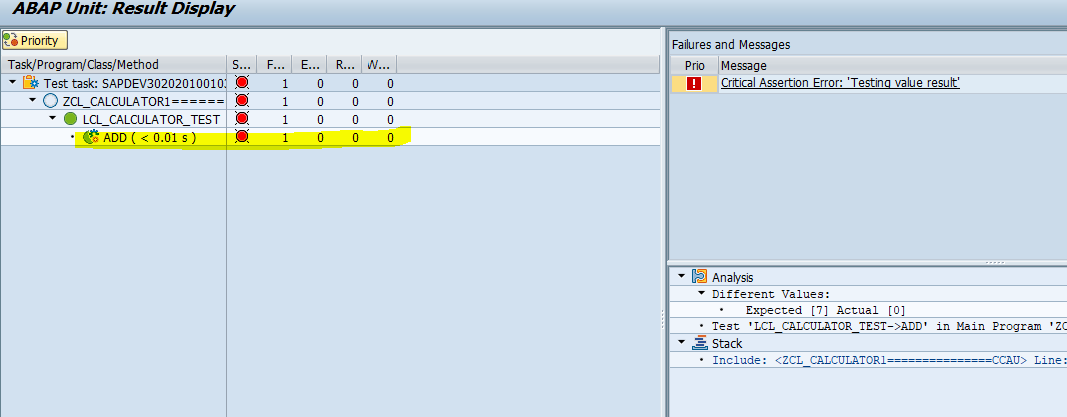
And click on create button

Now method “**ADD**” is added in product class

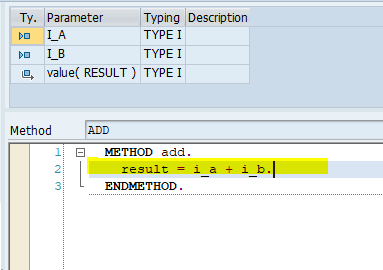


Activate class and Execute test by shortcut ‘**Ctrl+Shift+F10’** or Menu path ‘**Local test class->Execute->Unit Tests’**

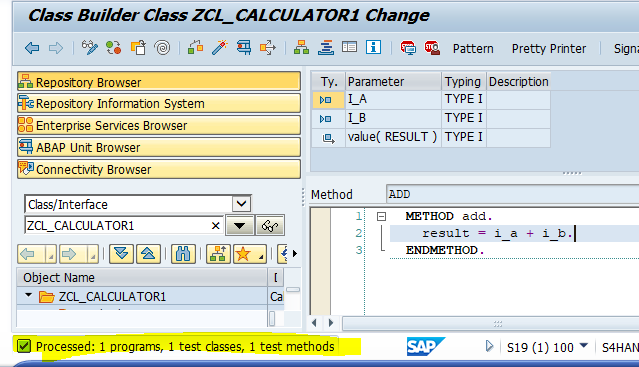
Unit testing fail for **ADD** method because no product code exist in product method in product class



Refactor product method code to pass test case as below



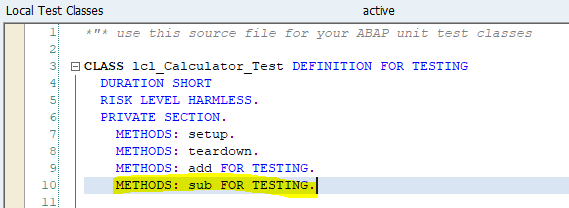
Execute unit test once again



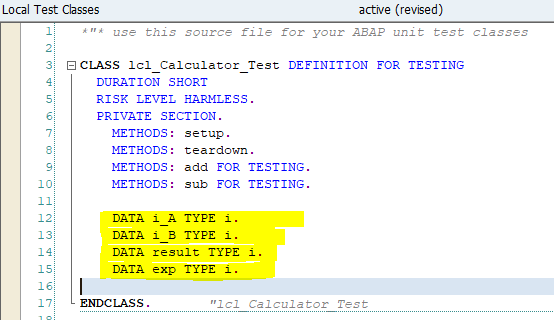
Now test case is passed

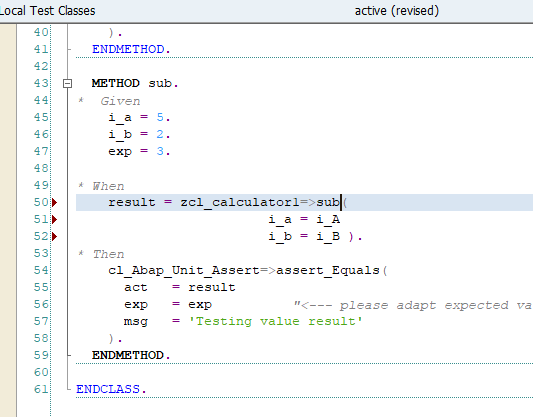
**Subtraction Operation**

Add Test method **‘SUB’** in test class definition, implementation and pass Given data

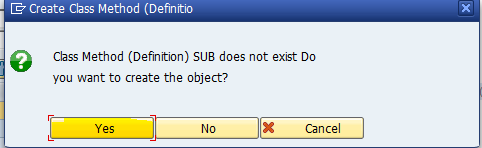


Add input and export into test class definition as below

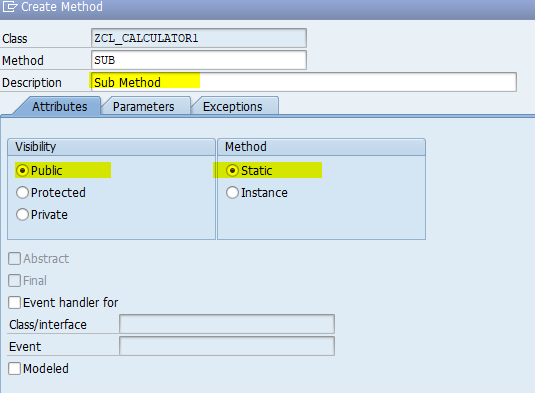




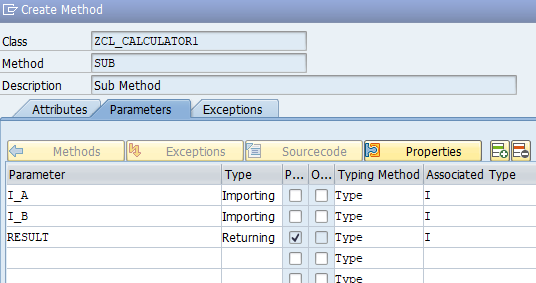
Double click on **SUB** method to create this method in product class



Click on Yes button

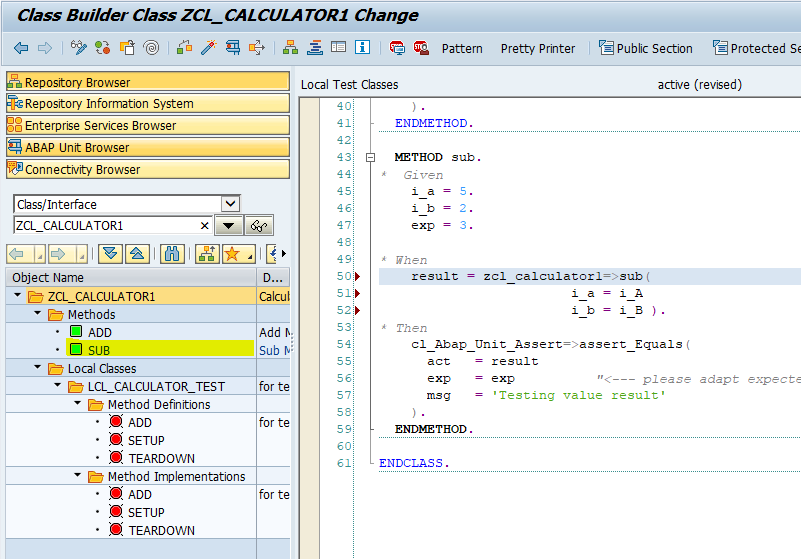
Enter Method description, Visibility as public and method as static 

Enter parameter in parameter tab



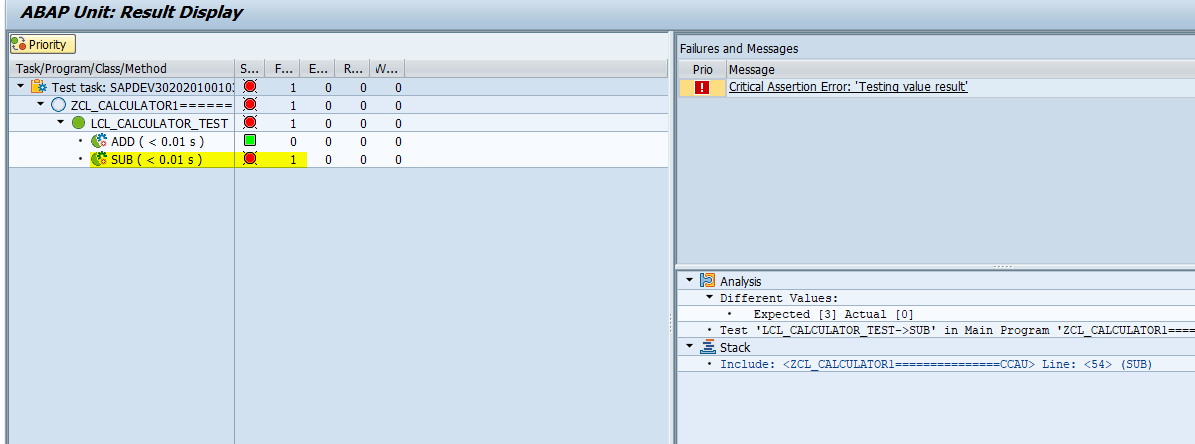
And click on create button

Now **SUB** method is added in product class

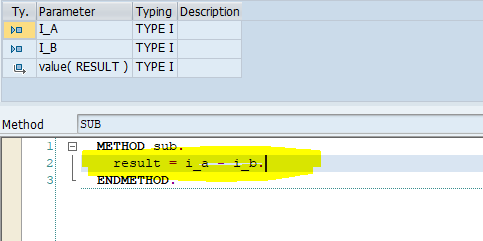


Activate class and Execute test by shortcut ‘**Ctrl+Shift+F10’** or Menu path ‘**Local test class->Execute->Unit Tests’**

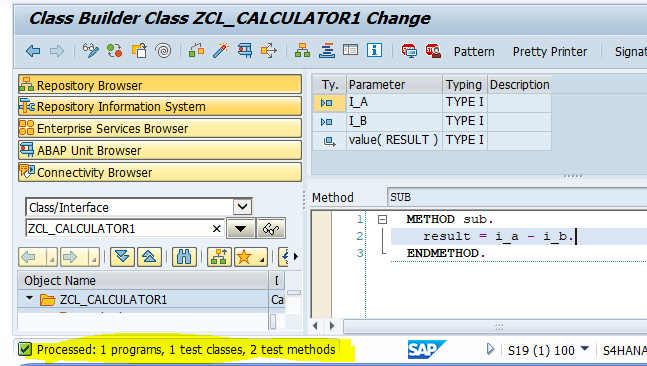
Unit testing fail for **SUB** method because no product code exist in product method in product class



Refactor product method code to pass test case as below



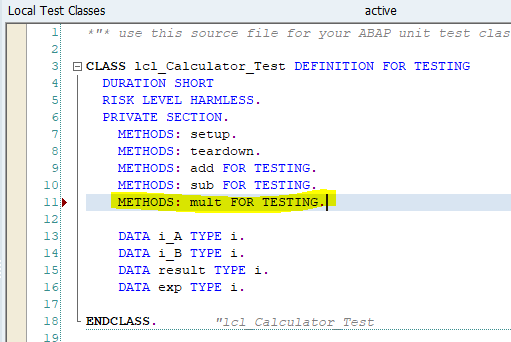
Execute unit test once again

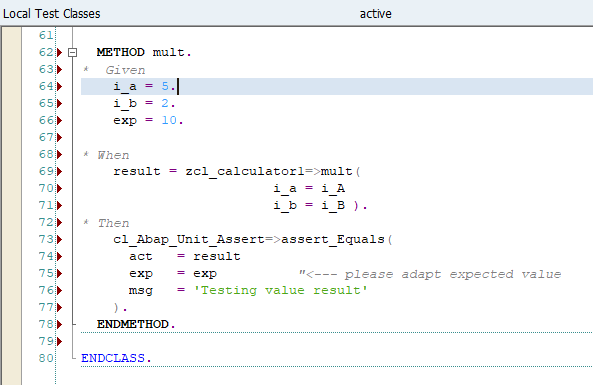


Now test case is passed

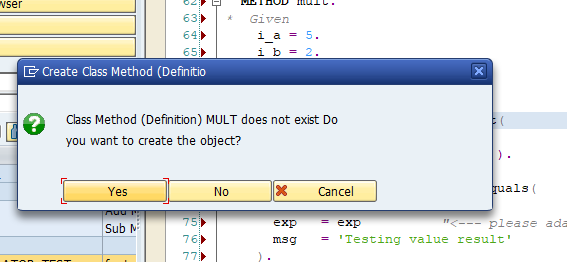
**Multiplication Operation**

Add Test method ‘**MULT**’ in test class definition, implementation and pass Given data

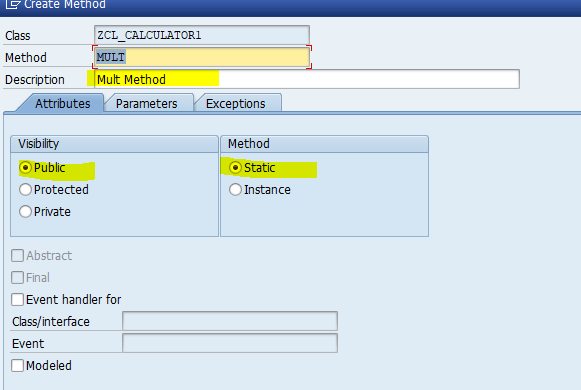




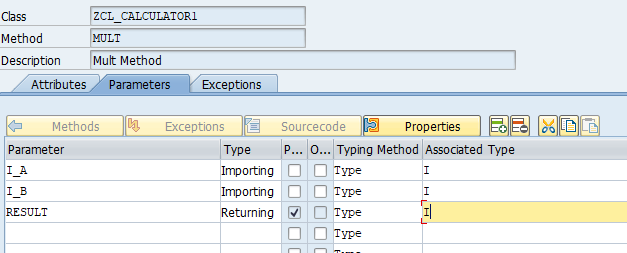
Double click on **MULT** method to create this method in product class



Click on Yes button

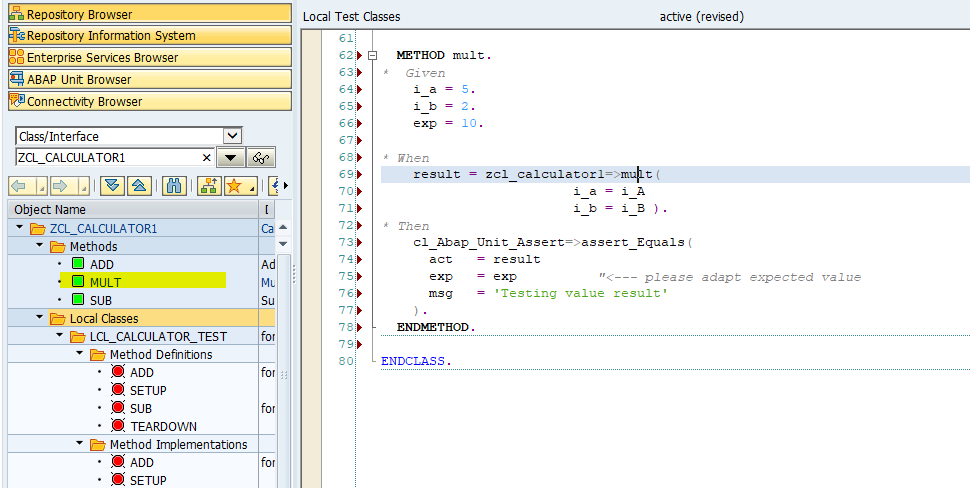
Enter Method description, Visibility as public and method as static 

Enter parameter in parameter tab



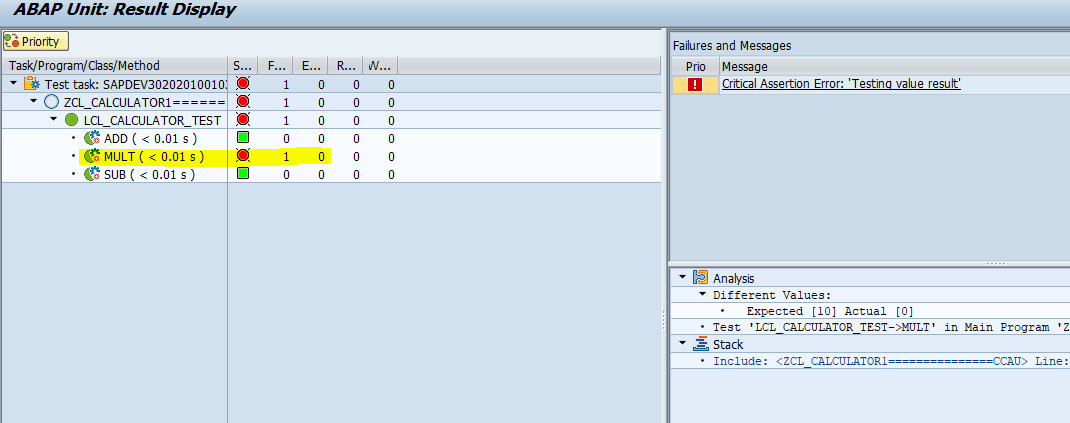
And click on create button

Now MULT method is added in product class

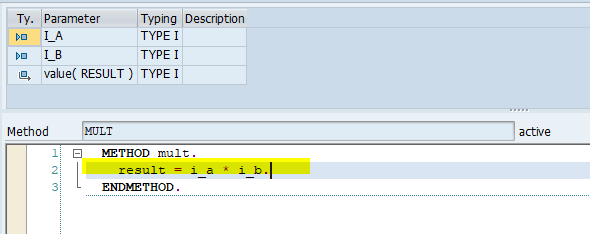


Activate class and Execute test by shortcut ‘**Ctrl+Shift+F10’** or Menu path ‘**Local test class->Execute->Unit Tests**’

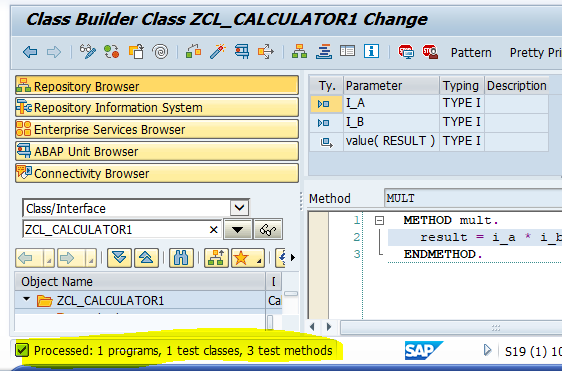
Unit testing fail for **MULT** method because no product code exist in product method in product class



Refactor product method code to pass test case as below



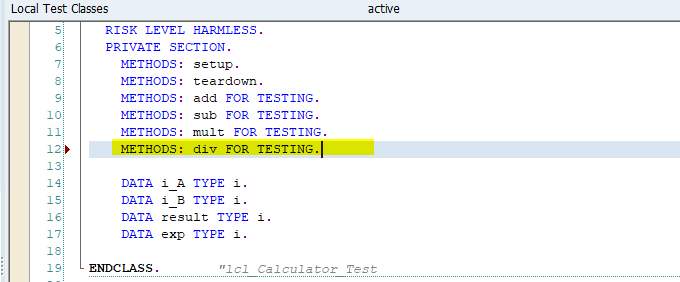
Execute unit test once again

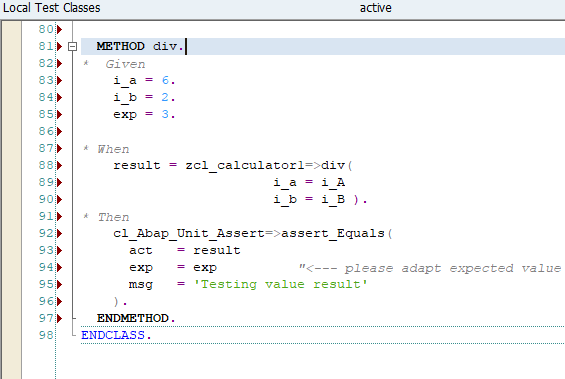


Now test case is passed

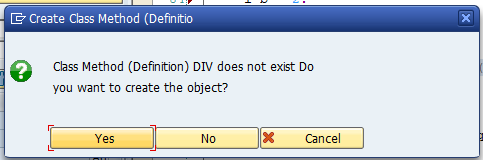
**Division Operation**

Add Test method ‘DIV’ in test class definition, implementation and pass Given data

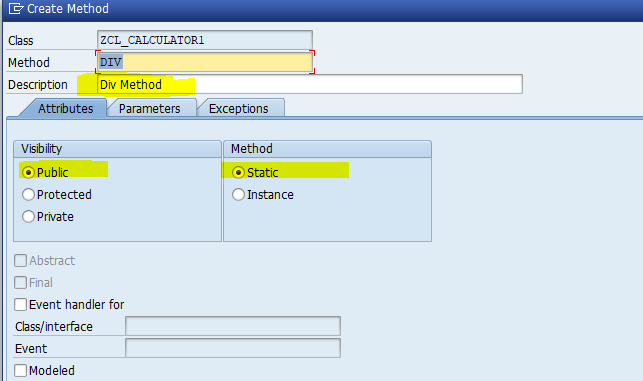




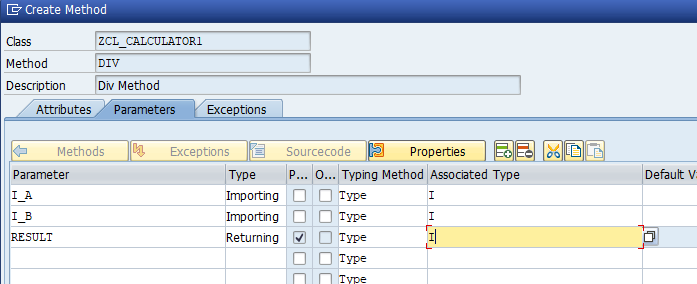
Double click on **DIV** method to create this method in product class



Click on Yes button

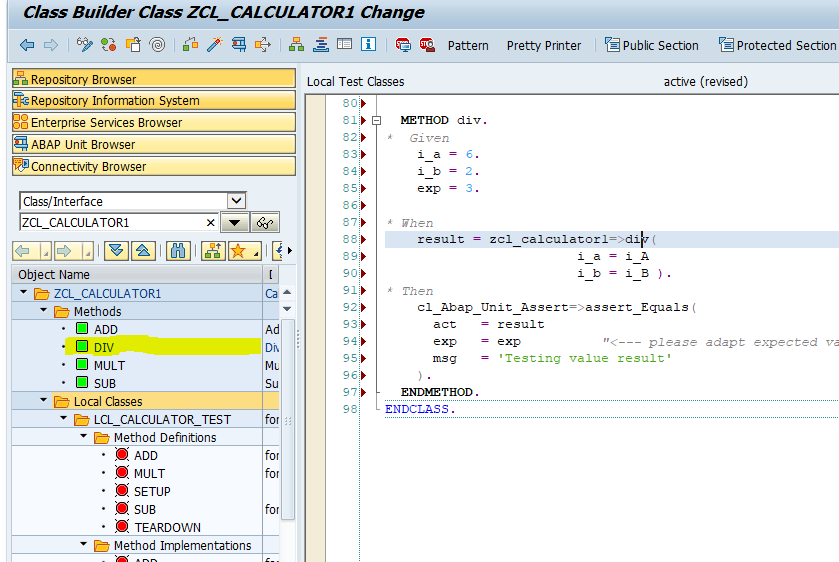
Enter Method description, Visibility as public and method as static 

Enter parameter in parameter tab



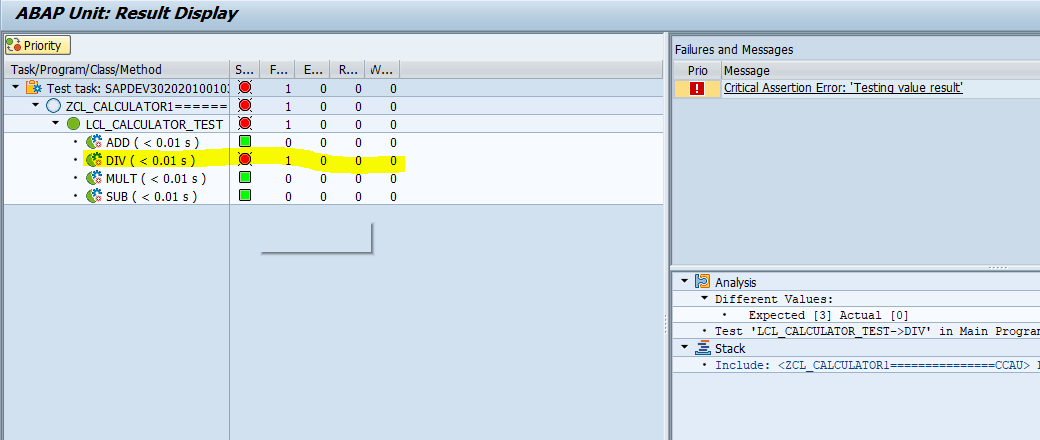
And click on create button

Now **DIV** method is added in product class

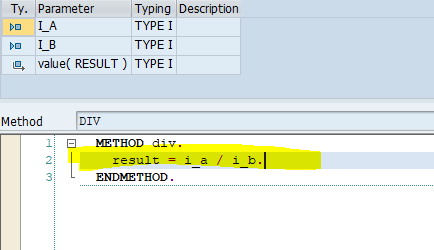


Activate class and Execute test by shortcut ‘**Ctrl+Shift+F10’** or Menu path ‘**Local test class->Execute->Unit Tests**’

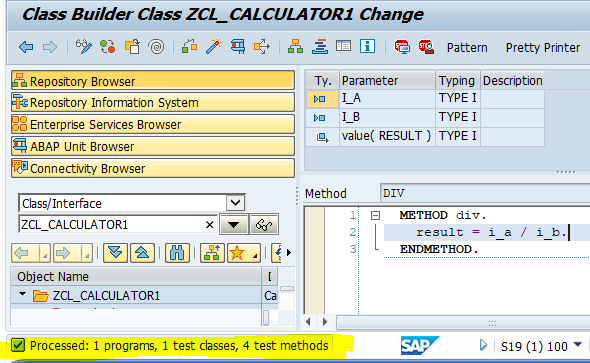
Unit testing fail for **DIV** method because no product code exist in product method in product class



Refactor product method code to pass test case as below



Execute unit test once again



Now test case is passed

