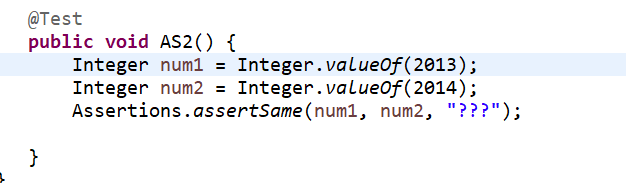
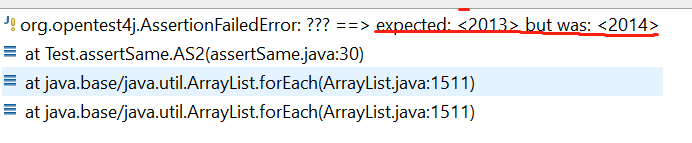
Hi,

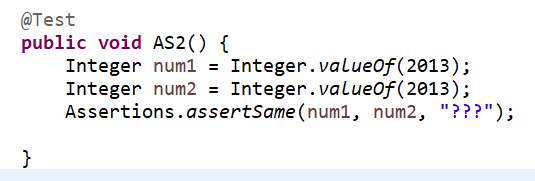
Thanks to the students asking questions about the assertSame() method, I realize that, in today’s class (Monday, 28th Nov), I confused the following code with one I planned to use before, and mistakenly said that the test would pass, without checking the detailed test result.



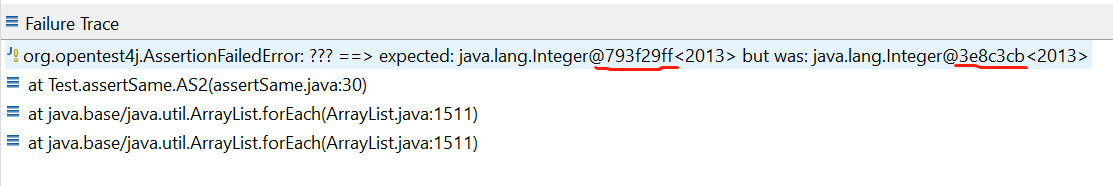
The correct answer should be: AS2() test would fail because num1 and num2 are obviously referring to two objects, see the following result: 

It indicates that we are expecting <2013>, which is an object (NOT a value), but we get object <2014>. So the assertSame() test fails, num1 and num2 are not referring to a same object.

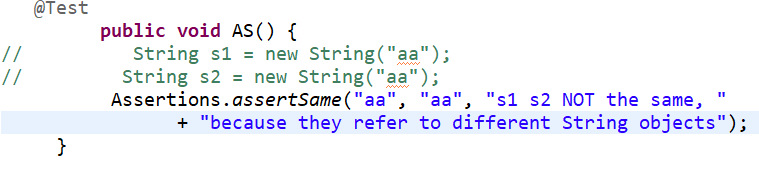
The test would still fail even if we change like the following way:



The following result indicate the two <2013> objects are actually stored in different address.

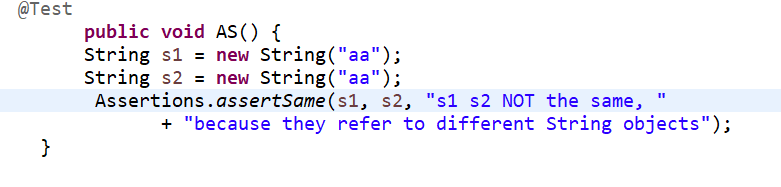


For those who wondering the following case:



The test will pass, because both “aa” are automatically treated as the same object (again, NOT value) in a same address.

However, in the following code:



The test will fail, because in this case, we actually define (force) that both “aa” refer to different objects using new String () method, and consequently they have two different memory addresses. So the test fails.

The key point here is that the assertSame() method deals with **objects having memory addresses,** not values. It is used to check if two objects refer to a same object. For values, we usually use assertEquals() to deal with them.

