

The screenshot shows an IDE with two main panels. The left panel, titled 'JUnit', displays the test results for 'Demo2'. It indicates that the test 'Finished after 0.108 seconds', with 'Runs: 1/1', 'Errors: 0', and 'Failures: 0'. Below this, a green bar represents the successful test run. The right panel shows the source code of 'Demo2.java' in the 'com.simplilearn.JUnitDemo' package. The code defines a 'Demo2' class with a '@Test' method 'testAssertions()'. This method performs several assertions: it checks that 'str1' equals 'str2', 'val1' is less than 'val2', 'val1' is greater than 'val2' (which fails, but the test still passes due to the 'assertFalse' wrapper), 'str1' is not null, 'str3' is null, 'str4' is the same as 'str5', 'str1' is not the same as 'str3', and that an array of strings equals another array of strings.

Package Explorer JUnit X
Finished after 0.108 seconds
Runs: 1/1 Errors: 0 Failures: 0
> Demo2 [Runner: JUnit 5] (0.020 s)
Failure Trace

```
1 package com.simplilearn.JUnitDemo;
2
3 import static org.junit.jupiter.api.Assertions.assertEquals;
4 import static org.junit.jupiter.api.Assertions.assertEquals;
5 import static org.junit.jupiter.api.Assertions.assertFalse;
6 import static org.junit.jupiter.api.Assertions.assertNotNull;
7 import static org.junit.jupiter.api.Assertions.assertNotSame;
8 import static org.junit.jupiter.api.Assertions.assertNull;
9 import static org.junit.jupiter.api.Assertions.assertSame;
10 import static org.junit.jupiter.api.Assertions.assertTrue;
11
12 import org.junit.jupiter.api.Test;
13
14 public class Demo2 {
15
16     @Test
17     public void testAssertions() {
18         String str1 = new String("abc");
19         String str2 = new String("abc");
20         String str3 = null;
21         String str4 = "abc";
22         String str5 = "abc";
23
24         int val1 = 5;
25         int val2 = 7;
26
27         String[] expectedArr = {"one", "two"};
28         String[] actualArr = {"one", "two"};
29
30         assertEquals(str1, str2);
31         assertTrue(val1 < val2);
32         assertFalse(val1 > val2);
33         assertNotNull(str1);
34         assertNull(str3);
35         assertSame(str4, str5);
36         assertNotSame(str1, str3);
37         assertEquals(expectedArr, actualArr);
38     }
39
40 }
41
```