

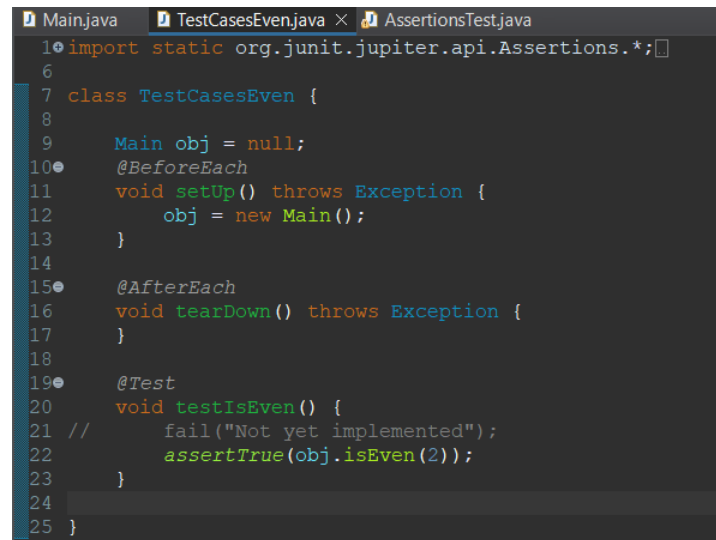
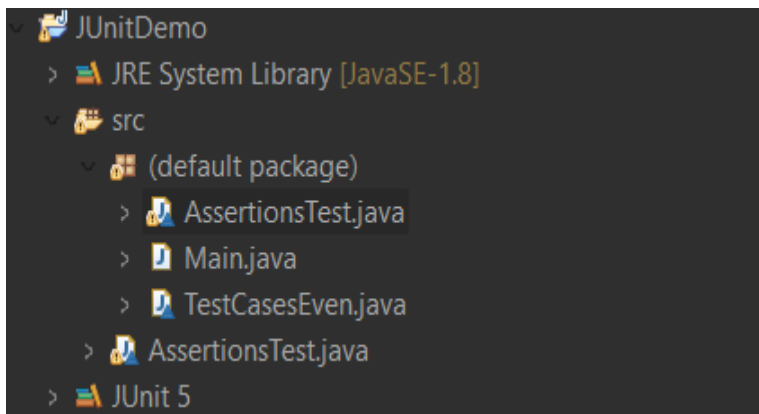
# JUnit Testing Exercises

## Exercise 1: Setting Up JUnit

Scenario: You need to set up JUnit in your Java project to start writing unit tests.

Steps:

1. Create a new Java project in your IDE (e.g., IntelliJ IDEA, Eclipse).
2. Add JUnit dependency to your project.
3. Create a new test class in your project.



## Exercise 2: Writing Basic Junit

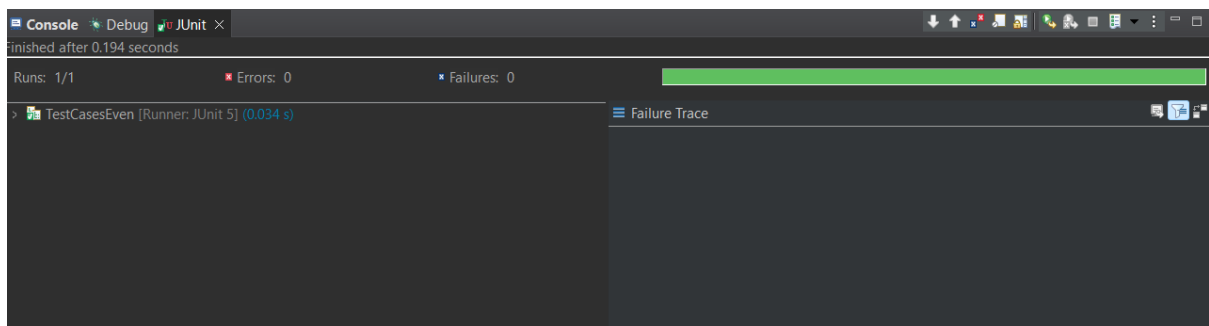
Tests Scenario:

You need to write basic JUnit tests for a simple Java class.

Steps:

1. Create a new Java class with some methods to test.
2. Write JUnit tests for these methods.

```
1 import static org.junit.jupiter.api.Assertions.*;
6
7 class TestCasesEven {
8
9     Main obj = null;
10    @BeforeEach
11    void setUp() throws Exception {
12        obj = new Main();
13    }
14
15    @AfterEach
16    void tearDown() throws Exception {
17    }
18
19    @Test
20    void testIsEven() {
21        // fail("Not yet implemented");
22        assertTrue(obj.isEven(2));
23    }
24
25 }
```



### Exercise 3: Assertions in JUnit

Scenario:

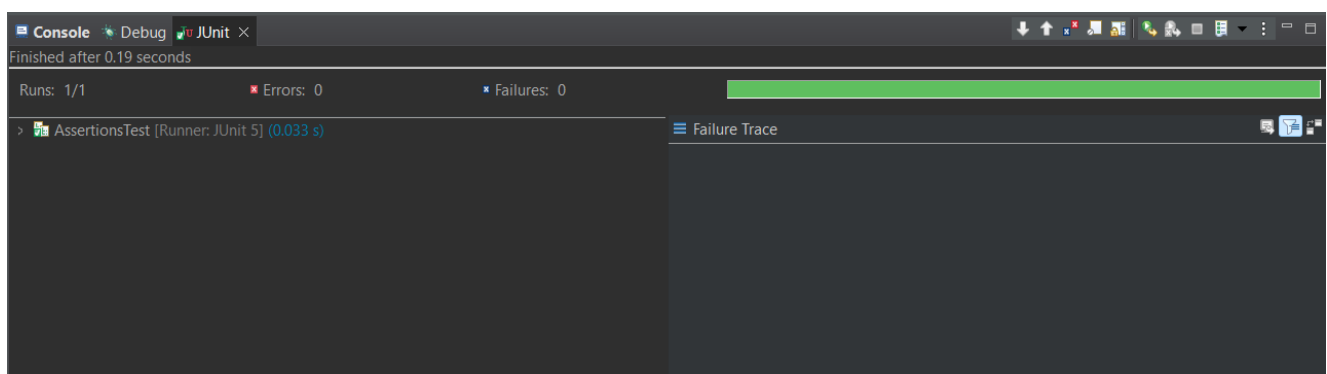
You need to use different assertions in JUnit to validate your test results.

Steps:

1. Write tests using various JUnit assertions.

```
Solution Code: public class AssertionsTest {
    @Test
    public void testAssertions() {
        // Assert equals assertEquals(5, 2 + 3);
        // Assert true assertTrue(5 > 3);
        // Assert false assertFalse(5 < 3);
        // Assert null assertNull(null);
        // Assert not null assertNotNull(new Object());
    }
}
```

```
Main.java  TestCasesEven.java  AssertionsTest.java ×
1 import static org.junit.jupiter.api.Assertions.*;
2
3
4
5
6
7 class AssertionsTest {
8
9     @BeforeEach
10    void setUp() throws Exception {
11    }
12
13    @AfterEach
14    void tearDown() throws Exception {}
15 }
16
17    @Test
18    void test() {
19        assertEquals(5, 2 + 3);
20        assertTrue(5 > 3);
21        assertFalse(5 < 3);
22        assertNull(null);
23        assertNotNull(new Object());
24    }
25
26 }
27
```



## Exercise 4: Arrange-Act-Assert (AAA) Pattern, Test Fixtures, Setup and Teardown Methods in JUnit

Scenario:

You need to organize your tests using the Arrange-Act-Assert (AAA) pattern and use setup and teardown methods.

Steps:

1. Write tests using the AAA pattern.
2. Use @Before and @After annotations for setup and teardown methods.

```
Main.java  TestCasesEven.java  AssertionsTest.java
9      Main obj = null;
10     @BeforeEach
11     void setUp() throws Exception {
12         obj = new Main();
13     }
14
15     @AfterEach
16     void tearDown() throws Exception {
17     }
18
19     @Test
20     void testIsEven() {
21         // fail("Not yet implemented");
22         // assertTrue(obj.isEven(2));
23
24         // Arrange
25         int input = 4;
26
27         // Act
28         boolean result = obj.isEven(input);
29
30         // Assert
31         assertTrue(result);
32     }
33
34 }
35
```

Console Debug JUnit ×

Finished after 0.181 seconds

Runs: 1/1      ✖ Errors: 0      ✖ Failures: 0

> TestCasesEven [Runner: JUnit 5] (0.032 s)

Failure Trace

**Submitted By:**

Name : Lingaraj Nayak

Superset ID:6387607