

Software Engineering IT314

Lab-8 Unit Testing with JUnit

Name: Vyom Patel

Student ID: 202001216

Date: 20/04/2023

➤ **Sample code for the 'Boa' class created for testing.**

```
// represents a boa constrictor
public class Boa {
    private String name;
    private int length; // the length of the boa, in feet
    private String favoriteFood;
    public Boa (String name, int length, String favoriteFood){
        this.name = name;
        this.length = length;
        this.favoriteFood = favoriteFood;
    }
    // returns true if this boa constrictor is healthy
    public boolean isHealthy(){
        return this.favoriteFood.equals("granola bars");
    }
    // returns true if the length of this boa constrictor is
    // less than the given cage length
    public boolean fitsInCage(int cageLength){
        return this.length < cageLength;
    }
}
```

➤ **Now we create a Junit Test case for testing the above 'Boa' class and named it 'Boa_test'.**

```
package test_lab8;
import static org.junit.jupiter.api.Assertions.*;
```

```

import org.junit.Before;
import org.junit.jupiter.api.Test;
class Boa_test {
    @Test
    public void testIsHealthyWithFavoriteFoodGranolaBars() {
        Boa boa = new Boa("Benny", 5, "granola bars");
        assertTrue(boa.isHealthy());
    }

    @Test
    public void testIsHealthyWithFavoriteFoodNotGranolaBars() {
        Boa boa = new Boa("Benny", 5, "mice");
        assertFalse(boa.isHealthy());
    }

    @Test
    public void testFitsInCageWhenLengthLessThanCageLength() {
        Boa boa = new Boa("Benny", 5, "granola bars");
        assertTrue(boa.fitsInCage(10));
    }

    @Test
    public void testFitsInCageWhenLengthGreaterThanCageLength() {
        Boa boa = new Boa("Benny", 20, "granola bars");
        assertFalse(boa.fitsInCage(10));
    }
}

```

- Now we ran the above test cases successfully.
- Now we use the **@Before** tag to run another test case.
- The **@Before** tag runs before every test case(**@Test**).

➤ **Q4: The code snippet below show how to execute it.**

```

private Boa jen;
private Boa ken;

```

```

@Before
public void setUp() throws Exception {
    jen = new Boa("Jennifer", 2, "grapes");
    ken = new Boa("Kenneth", 3, "granola bars");
}

```

➤ **Q5: Below is the test code for the question.**

```

@Test
public void testIsHealthy() {
    Boa jen = new Boa("Jen", 5, "granola bars");
    Boa ken = new Boa("Ken", 6, "mice");

    assertTrue(jen.isHealthy());
    assertFalse(ken.isHealthy());
}

```

```

@Test
public void testFitsInCage() {
    Boa jen = new Boa("Jen", 5, "granola bars");
    Boa ken = new Boa("Ken", 6, "mice");

    assertFalse(jen.fitsInCage(2));
    assertTrue(jen.fitsInCage(10));
    assertTrue(jen.fitsInCage(15));
    assertFalse(ken.fitsInCage(4));
    assertTrue(ken.fitsInCage(15));
    assertTrue(ken.fitsInCage(20));
}

```

➤ **Q7: Creating another method in the 'Boa' class.**

Method Code:

```

public int lengthInInches() {
    return this.length * 12;
}

```

Testing code:

@Test

```
public void testLengthInInches() {  
    Boa boa = new Boa("John", 5, "grapes");  
    int expectedLengthInInches = 60;  
    int actualLengthInInches = boa.lengthInInches();  
    assertEquals(expectedLengthInInches, actualLengthInInches);  
}
```

□ Here is the snippet for the complete testing code.

The screenshot shows the Eclipse IDE interface. The main editor displays the `Boa_test.java` file, which contains the following code:

```
1 package test_lab8;  
2  
3 import static org.junit.jupiter.api.Assertions.*;  
4  
5 import org.junit.Before;  
6 import org.junit.jupiter.api.Test;  
7  
8 class Boa_test {  
9  
10     @Test  
11     public void testIsHealthyWithFavoriteFoodGranolaBars() {  
12         Boa boa = new Boa("Benny", 5, "granola bars");  
13         assertTrue(boa.isHealthy());  
14     }  
15  
16     @Test  
17     public void testIsHealthyWithFavoriteFoodNotGranolaBars() {  
18         Boa boa = new Boa("Benny", 5, "mice");  
19         assertFalse(boa.isHealthy());  
20     }  
21  
22     @Test  
23     public void testFitsInCageWhenLengthLessThanCageLength() {  
24         Boa boa = new Boa("Benny", 5, "granola bars");  
25         assertTrue(boa.fitsInCage(10));  
26     }  
27  
28     @Test  
29     public void testFitsInCageWhenLengthGreaterThanCageLength() {  
30         Boa boa = new Boa("Benny", 20, "granola bars");  
31         assertFalse(boa.fitsInCage(10));  
32     }  
33 }  
34  
35 private Boa jen;  
36 private Boa ken;  
37  
38 @Before  
39 public void setUp() throws Exception {  
40     jen = new Boa("Jennifer", 2, "grapes");  
41     ken = new Boa("Kenneth", 3, "granola bars");  
42 }  
43  
44
```

The left sidebar shows the Project Explorer with the `Boa_test` class selected. The bottom status bar indicates that the test was finished after 0.071 seconds, with 7/7 runs, 0 errors, and 0 failures.

The bottom right pane shows the Coverage report for the `Lab_008` element:

Element	Coverage	Covered Instructions	Missed Instructions	Total Instructions
Lab_008	50.3 %	74	73	147

File Edit Source Refactor Navigate Search Project Run Window Help

Project ExplorerJUnit

Finished after 0.071 seconds

Runs: 7/7Errors: 0Failures: 0

> Boa_test [Runner: JUnit 5] (0.014 s)

Failure Trace

Boa.javaBoa_test.java

```
38 @Before
39 public void setUp() throws Exception {
40     jen = new Boa("Jennifer", 2, "grapes");
41     ken = new Boa("Kenneth", 3, "granola bars");
42 }
43
44
45 @Test
46 public void testLengthInInches() {
47     Boa boa = new Boa("John", 5, "grapes");
48     int expectedLengthInInches = 60;
49     int actualLengthInInches = boa.lengthInInches();
50     assertEquals(expectedLengthInInches, actualLengthInInches);
51 }
52
53 @Test
54 public void testIsHealthy() {
55     Boa jen = new Boa("Jen", 5, "granola bars");
56
57     Boa ken = new Boa("Ken", 6, "mice");
58
59     assertTrue(jen.isHealthy());
60     assertFalse(ken.isHealthy());
61 }
62
63 @Test
64 public void testFitsInCage() {
65     Boa jen = new Boa("Jen", 5, "granola bars");
66
67     Boa ken = new Boa("Ken", 6, "mice");
68
69     assertFalse(jen.fitsInCage(2));
70     assertTrue(jen.fitsInCage(10));
71     assertTrue(jen.fitsInCage(15));
72     assertFalse(ken.fitsInCage(4));
73     assertTrue(ken.fitsInCage(15));
74     assertTrue(ken.fitsInCage(20));
75 }
76
77
78
79 }
80
81
```

Coverage

Element	Coverage	Covered Instructio...	Missed Instructions	Total Instructions
> Lab_008	50.3 %	74	73	147