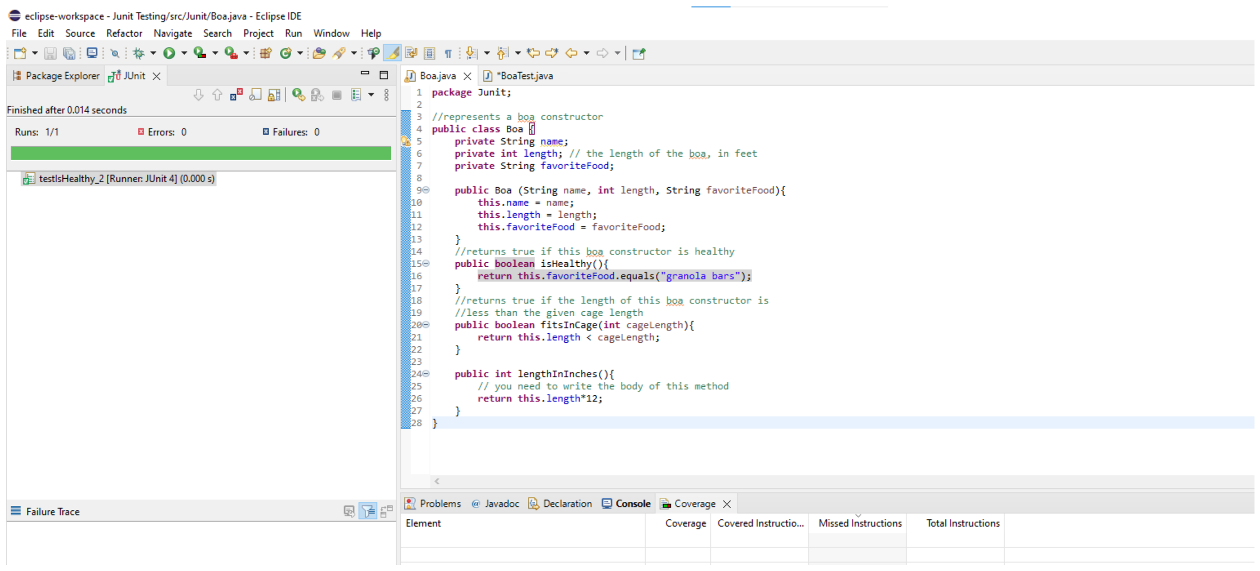
SOFTWARE ENGINEERING

LAB:8
TANK NANDANI JAGDISHBHAI
STUDENT ID:202001201

3.Boa.java file:

```
package Junit;


import static org.junit.Assert.*;


import org.junit.Before;
import org.junit.Test;


public class BoaTest {

  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");
  }
  @Test
  public void testIsHealthy_1() {
    boolean output = jen.isHealthy();
    assertEquals(output,false);
  }
  @Test
  public void testFitsInCage_1() {
    boolean output = jen.fitsInCage(5);
    assertEquals(output, true);
  }}
```


eclipse-workspace - Junit Testing/src/Junit/Boa.java - Eclipse IDE

Finished after 0.014 seconds

Runs: 1/1 Errors: 0 Failures: 0

testIsHealthy_2 [Runner: JUnit 4] (0.000 s)

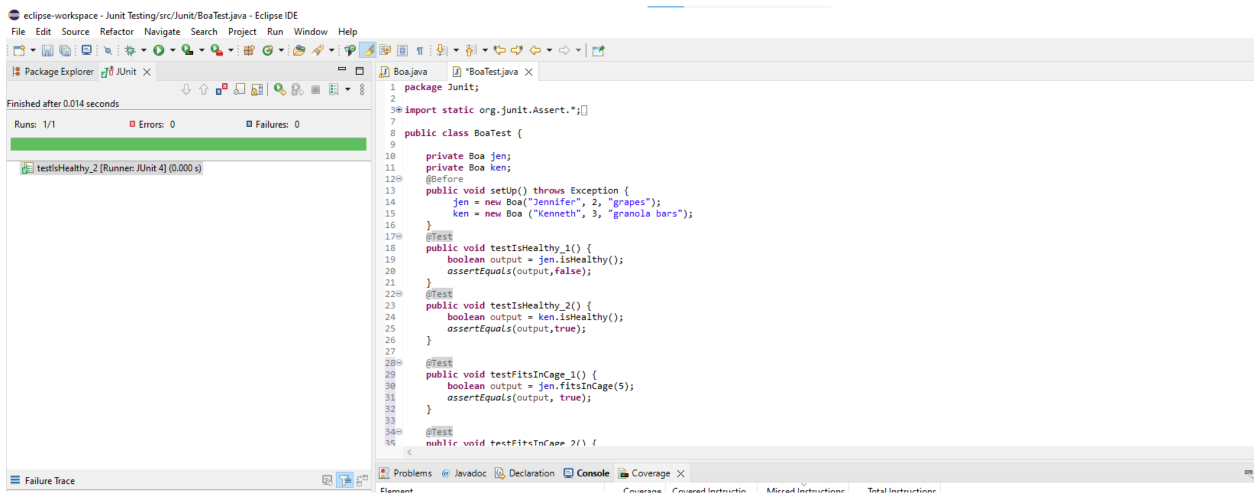
```java
package Junit;

//represents a boa constructor
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 constructor is healthy
    public boolean isHealthy(){
        return this.favoriteFood.equals("granola bars");
    }
    //returns true if the length of this boa constructor is
    //less than the given cage length
    public boolean fitsInCage(int cageLength){
        return this.length < cageLength;
    }

    public int lengthInInches(){
        // you need to write the body of this method
        return this.length*12;
    }
}
```

Element	Coverage	Covered Instructio...	Missed Instructions	Total Instructions

4. Modify Setup method:

BoaTest.java file:

```java
package Junit;

import static org.junit.Assert.*;

import org.junit.Before;
import org.junit.Test;

public class BoaTest {

  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");
  }
}
```



```
Finished after 0.014 seconds
Runs: 1/1    Errors: 0    Failures: 0
testIsHealthy_2 [Runner: JUnit 4] (0.000 s)
```

```java
package Junit;

import static org.junit.Assert.*;

public class BoaTest {

    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");
    }
    @Test
    public void testIsHealthy_1() {
        boolean output = jen.isHealthy();
        assertEquals(output,false);
    }
    @Test
    public void testIsHealthy_2() {
        boolean output = ken.isHealthy();
        assertEquals(output,true);
    }

    @Test
    public void testFitsInCage_1() {
        boolean output = jen.fitsInCage(5);
        assertEquals(output, true);
    }

    @Test
    public void testFitsInCage_2() {
```

5. @Test stubs

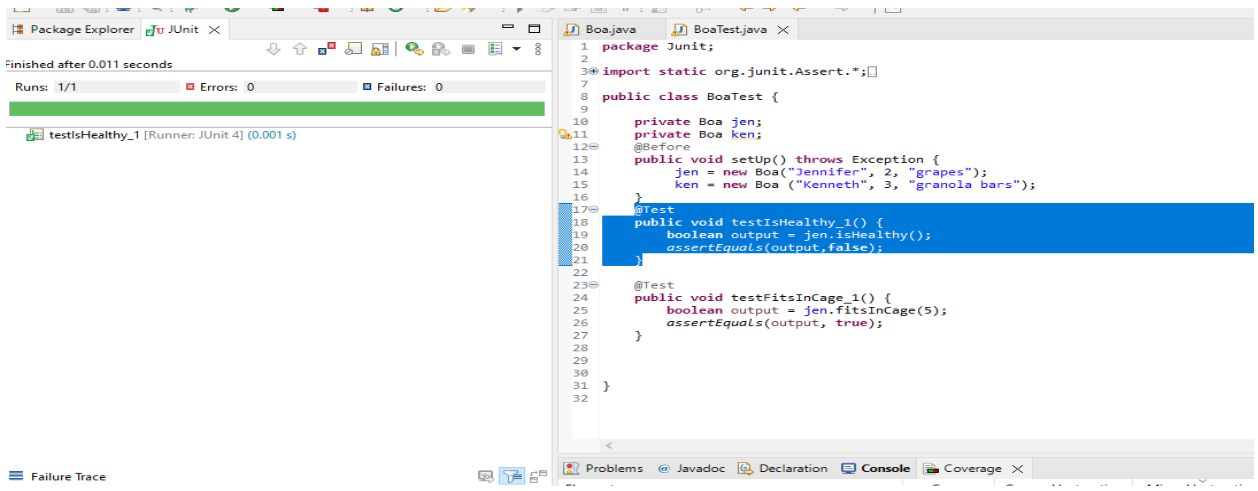
A.Testing for testIshealthy() function:

Code:

```java
 @Test
  public void testIsHealthy() {
    boolean output = jen.isHealthy();
    assertEquals(output,false);

  }
```

I created two test case for testIshealthy function and test as below.mow we test using junit testing method.

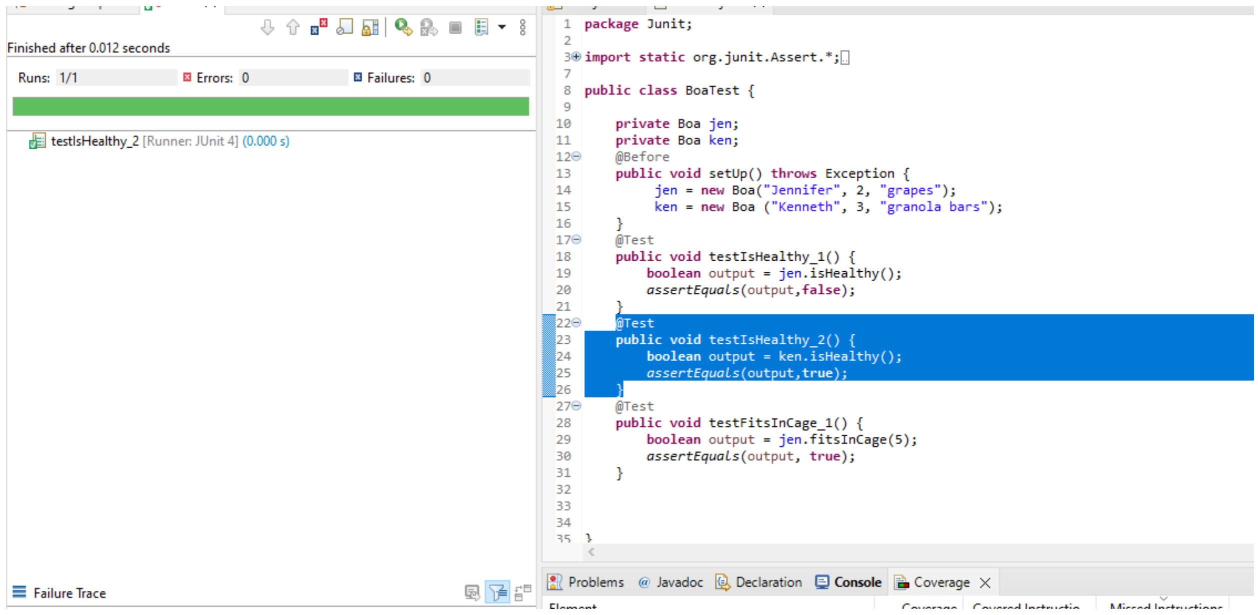
1. `testIshealthy1()`



```
Finished after 0.011 seconds
Runs: 1/1    Errors: 0    Failures: 0
testIsHealthy_1 [Runner: JUnit 4] (0.001 s)
```

```java
package Junit;

import static org.junit.Assert.*;

public class BoaTest {

    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");
    }
    @Test
    public void testIsHealthy_1() {
        boolean output = jen.isHealthy();
        assertEquals(output,false);
    }

    @Test
    public void testFitsInCage_1() {
        boolean output = jen.fitsInCage(5);
        assertEquals(output, true);
    }



}
```

2. `testIshealthy2()`



```
Finished after 0.012 seconds
Runs: 1/1    Errors: 0    Failures: 0
testIsHealthy_2 [Runner: JUnit 4] (0.000 s)
```

```java
package Junit;

import static org.junit.Assert.*;

public class BoaTest {

    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");
    }
    @Test
    public void testIsHealthy_1() {
        boolean output = jen.isHealthy();
        assertEquals(output,false);
    }
    @Test
    public void testIsHealthy_2() {
        boolean output = ken.isHealthy();
        assertEquals(output,true);
    }
    @Test
    public void testFitsInCage_1() {
        boolean output = jen.fitsInCage(5);
        assertEquals(output, true);
    }



}
```

B.Testing for testFitsInCage()

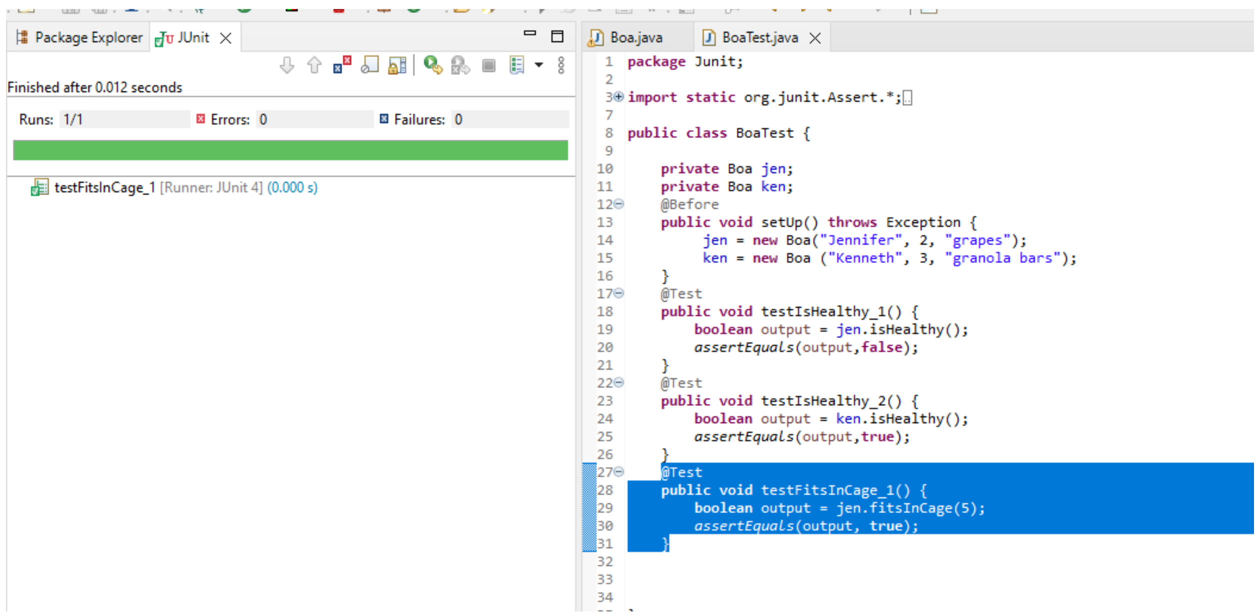
Code:

```java
@Test
  public void testFitsInCage() {
    boolean output = jen.fitsInCage(5);
    assertEquals(output, true);
```

    }

I created three test cases for function testFitsInCage()and test as below.mow we test using junit testing method.

1.testFitsInCage_1()



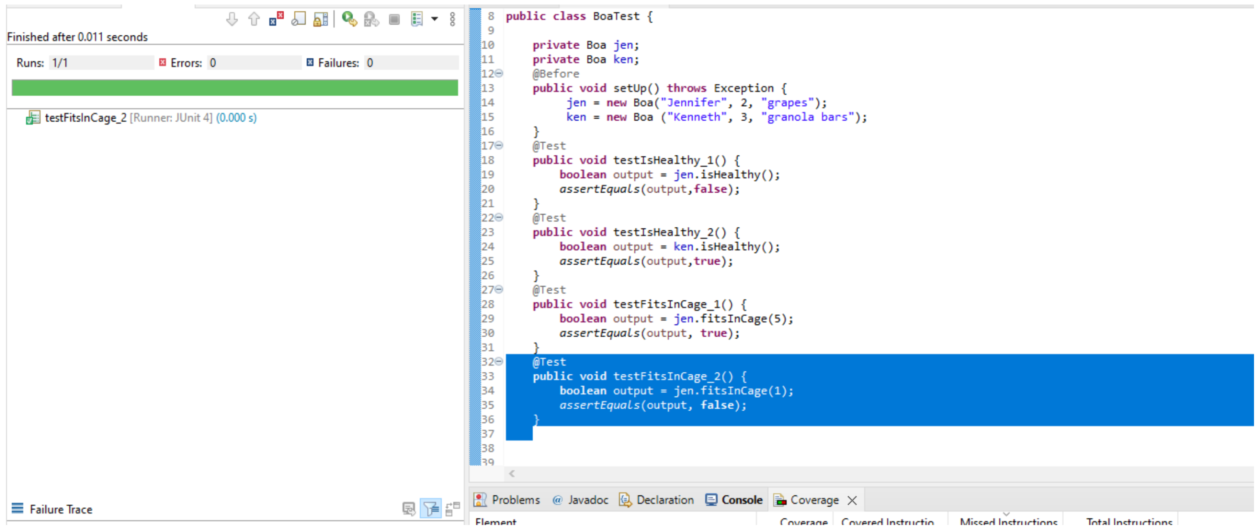
Finished after 0.012 seconds

Runs: 1/1 Errors: 0 Failures: 0

testFitsInCage_1 [Runner: JUnit 4] (0.000 s)

```
package Junit;

import static org.junit.Assert.*;

public class BoaTest {

    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");
    }
    @Test
    public void testIsHealthy_1() {
        boolean output = jen.isHealthy();
        assertEquals(output,false);
    }
    @Test
    public void testIsHealthy_2() {
        boolean output = ken.isHealthy();
        assertEquals(output,true);
    }
    @Test
    public void testFitsInCage_1() {
        boolean output = jen.fitsInCage(5);
        assertEquals(output, true);
    }
```

2.testFitsInCage_2()



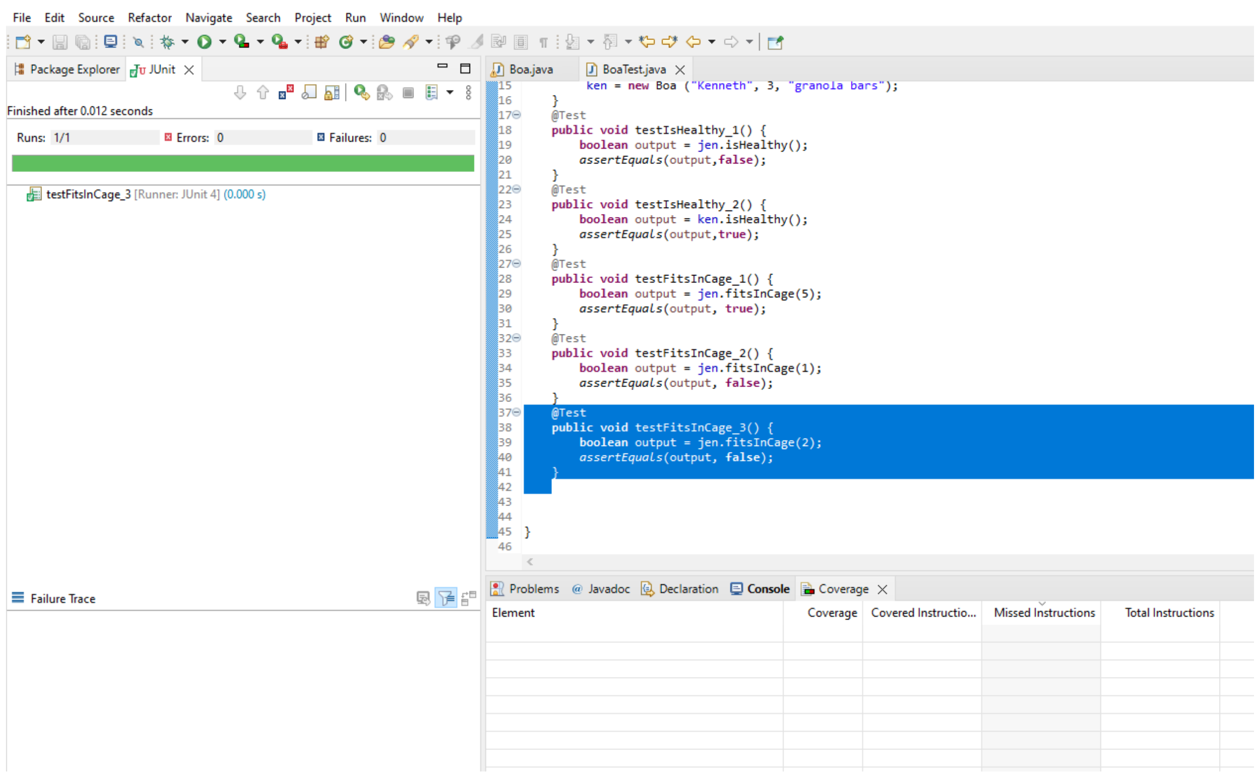
Finished after 0.011 seconds

Runs: 1/1 Errors: 0 Failures: 0

testFitsInCage_2 [Runner: JUnit 4] (0.000 s)

```
public class BoaTest {

    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");
    }
    @Test
    public void testIsHealthy_1() {
        boolean output = jen.isHealthy();
        assertEquals(output,false);
    }
    @Test
    public void testIsHealthy_2() {
        boolean output = ken.isHealthy();
        assertEquals(output,true);
    }
    @Test
    public void testFitsInCage_1() {
        boolean output = jen.fitsInCage(5);
        assertEquals(output, true);
    }
    @Test
    public void testFitsInCage_2() {
        boolean output = jen.fitsInCage(1);
        assertEquals(output, false);
    }
```

3.testFitsCage_3()



Finished after 0.012 seconds

Runs: 1/1 Errors: 0 Failures: 0

testFitsInCage_3 [Runner: JUnit 4] (0.000 s)

```java
            ken = new Boa ("Kenneth", 3, "granola bars");
    }
    @Test
    public void testIsHealthy_1() {
        boolean output = jen.isHealthy();
        assertEquals(output,false);
    }
    @Test
    public void testIsHealthy_2() {
        boolean output = ken.isHealthy();
        assertEquals(output,true);
    }
    @Test
    public void testFitsInCage_1() {
        boolean output = jen.fitsInCage(5);
        assertEquals(output, true);
    }
    @Test
    public void testFitsInCage_2() {
        boolean output = jen.fitsInCage(1);
        assertEquals(output, false);
    }
    @Test
    public void testFitsInCage_3() {
        boolean output = jen.fitsInCage(2);
        assertEquals(output, false);
    }



}
```

Element	Coverage	Covered Instructio...	Missed Instructions	Total Instructions

7. Add a new method to the Boa class, with this purpose and signature:

```java
// produces the length of the Boa in inches
public int lengthInInches(){
// you need to write the body of this method
}
```

Adding new method:

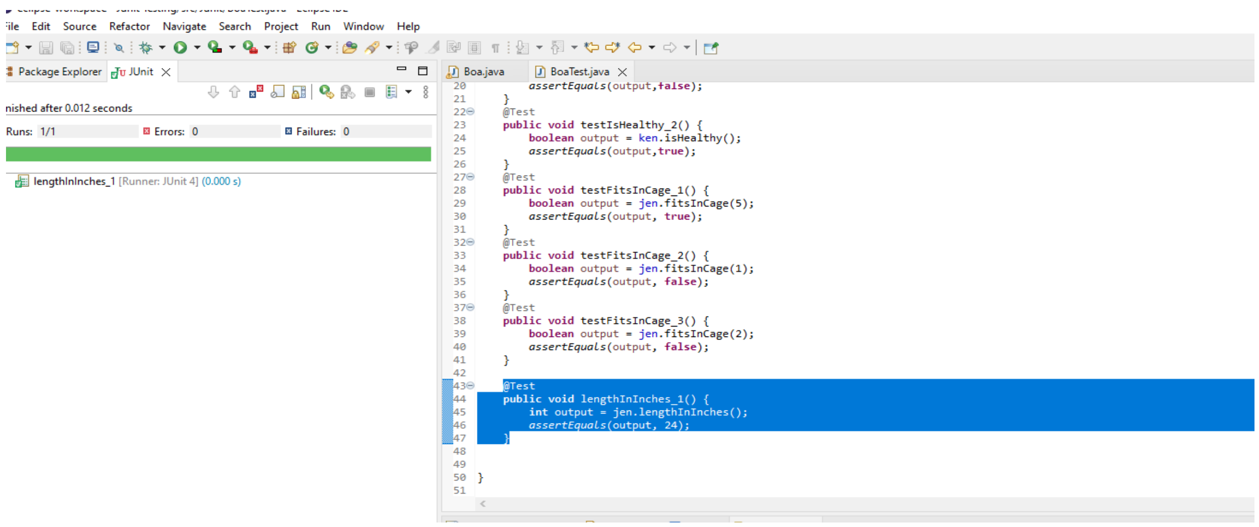
Testing for lengthInInches()

I created the two test case for lengthininches() function and then using junit testing method.

Code:

```
@Test
public void lengthInInches() {
  int output = jen.lengthInInches();
  assertEquals(output, 24);
}
```

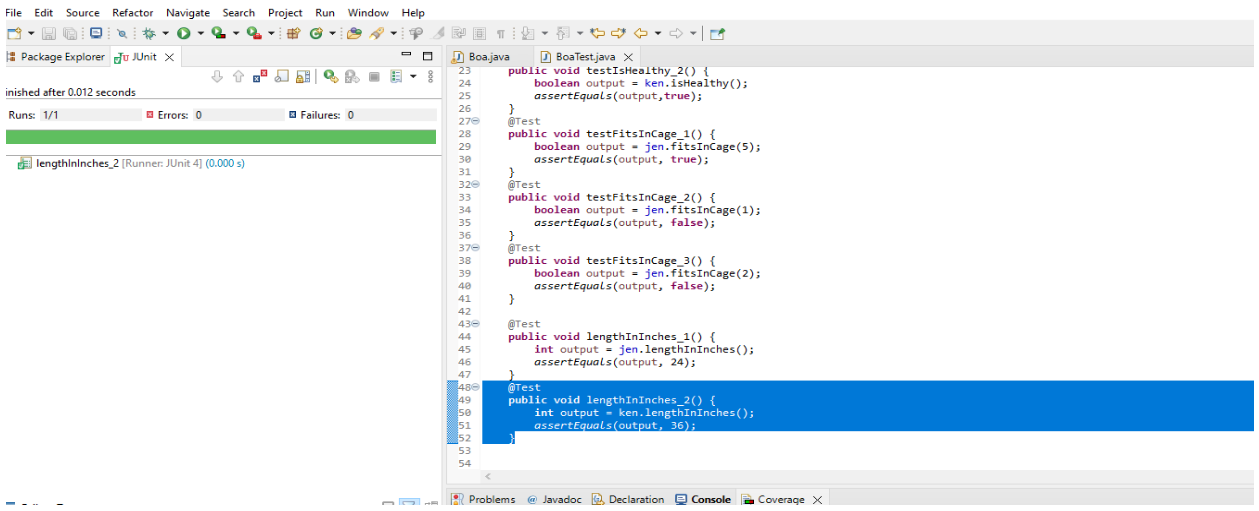
1.lengthInInches_1()


File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
JUnit
Boa.java
BoaTest.java

```
nished after 0.012 seconds
Runs: 1/1    Errors: 0    Failures: 0
lengthInInches_1 [Runner: JUnit 4] (0.000 s)
```

```
        assertEquals(output,false);
    }
    @Test
    public void testIsHealthy_2() {
        boolean output = ken.isHealthy();
        assertEquals(output,true);
    }
    @Test
    public void testFitsInCage_1() {
        boolean output = jen.fitsInCage(5);
        assertEquals(output, true);
    }
    @Test
    public void testFitsInCage_2() {
        boolean output = jen.fitsInCage(1);
        assertEquals(output, false);
    }
    @Test
    public void testFitsInCage_3() {
        boolean output = jen.fitsInCage(2);
        assertEquals(output, false);
    }

    @Test
    public void lengthInInches_1() {
        int output = jen.lengthInInches();
        assertEquals(output, 24);
    }


}
```

2.lengthInInches_2()



```
inished after 0.012 seconds
Runs: 1/1    Errors: 0    Failures: 0
lengthInInches_2 [Runner: JUnit 4] (0.000 s)
```

```
    public void testIsHealthy_2() {
        boolean output = ken.isHealthy();
        assertEquals(output,true);
    }
    @Test
    public void testFitsInCage_1() {
        boolean output = jen.fitsInCage(5);
        assertEquals(output, true);
    }
    @Test
    public void testFitsInCage_2() {
        boolean output = jen.fitsInCage(1);
        assertEquals(output, false);
    }
    @Test
    public void testFitsInCage_3() {
        boolean output = jen.fitsInCage(2);
        assertEquals(output, false);
    }

    @Test
    public void lengthInInches_1() {
        int output = jen.lengthInInches();
        assertEquals(output, 24);
    }
    @Test
    public void lengthInInches_2() {
        int output = ken.lengthInInches();
        assertEquals(output, 36);
    }

```

The modified testFitsInCage() method tests the results of the fitsInCage() method when the cage length is less than, equal to, and greater than the length of the boa for both healthyBoa and unhealthyBoa objects.

Since the setUp() method initializes two different Boa objects, healthyBoa and unhealthyBoa, there is no need to write separate tests for "jen" and "ken". The tests should cover both objects as specified in the setUp() method.