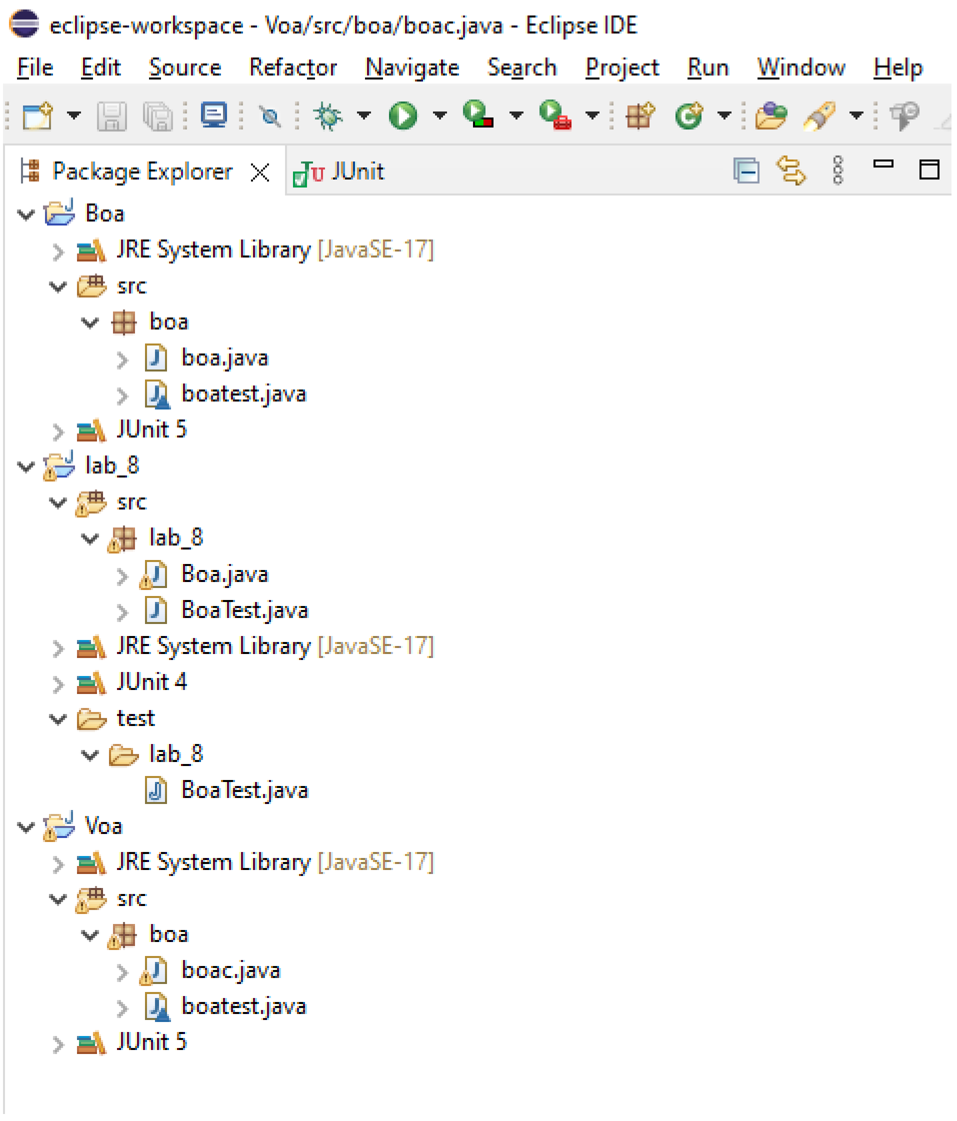
Lab-8
Software engineering
Id: 202001278
Name : shraddha doshi

Lab Exercises
1. Create a new Eclipse project, and within the project create a package.


eclipse-workspace - Voa/src/boa/boac.java - Eclipse IDE

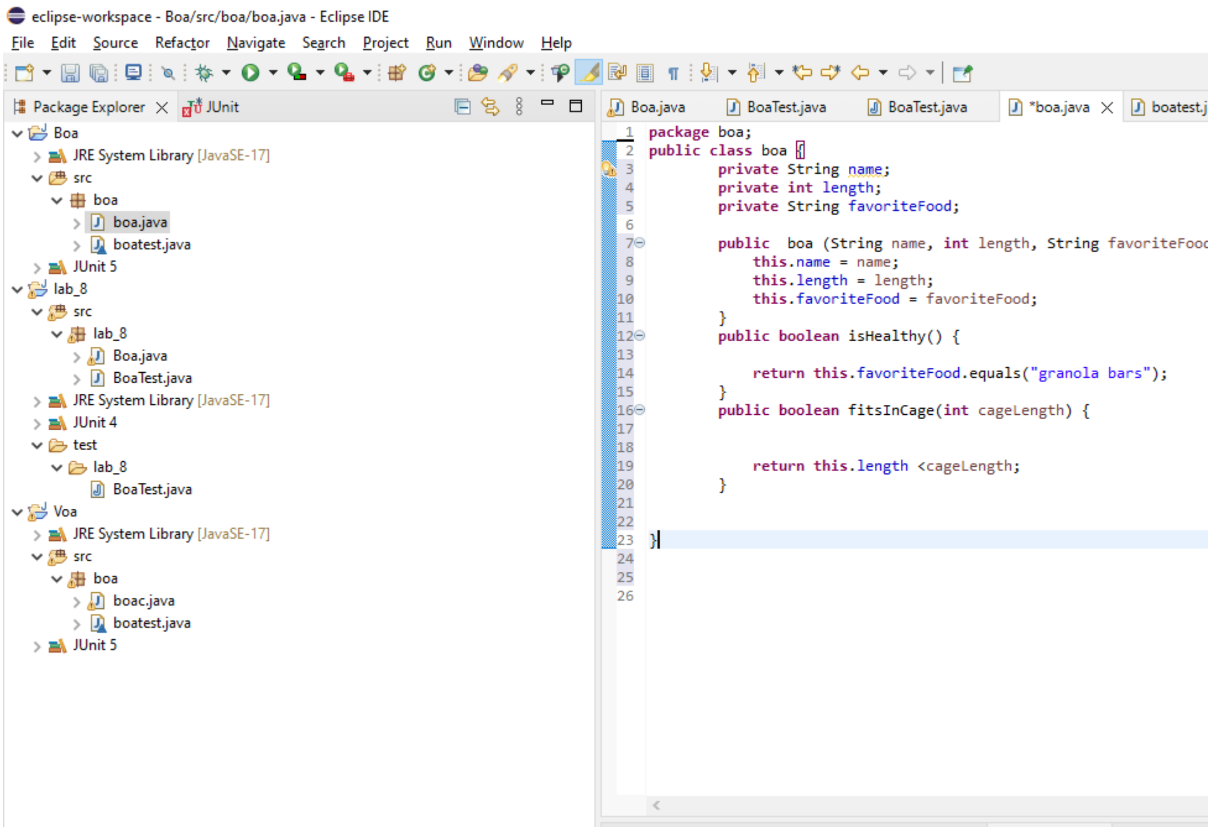
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
JUnit

```
Boa
  JRE System Library [JavaSE-17]
  src
    boa
      boa.java
      boatest.java
  JUnit 5
lab_8
  src
    lab_8
      Boa.java
      BoaTest.java
  JRE System Library [JavaSE-17]
  JUnit 4
  test
    lab_8
      BoaTest.java
Voa
  JRE System Library [JavaSE-17]
  src
    boa
      boac.java
      boatest.java
  JUnit 5
```

As you can see first i have created voa the project name in src i have created a package called boa and in in package i have created boac class and got boac.java and then i have created boatest.java through voa project name

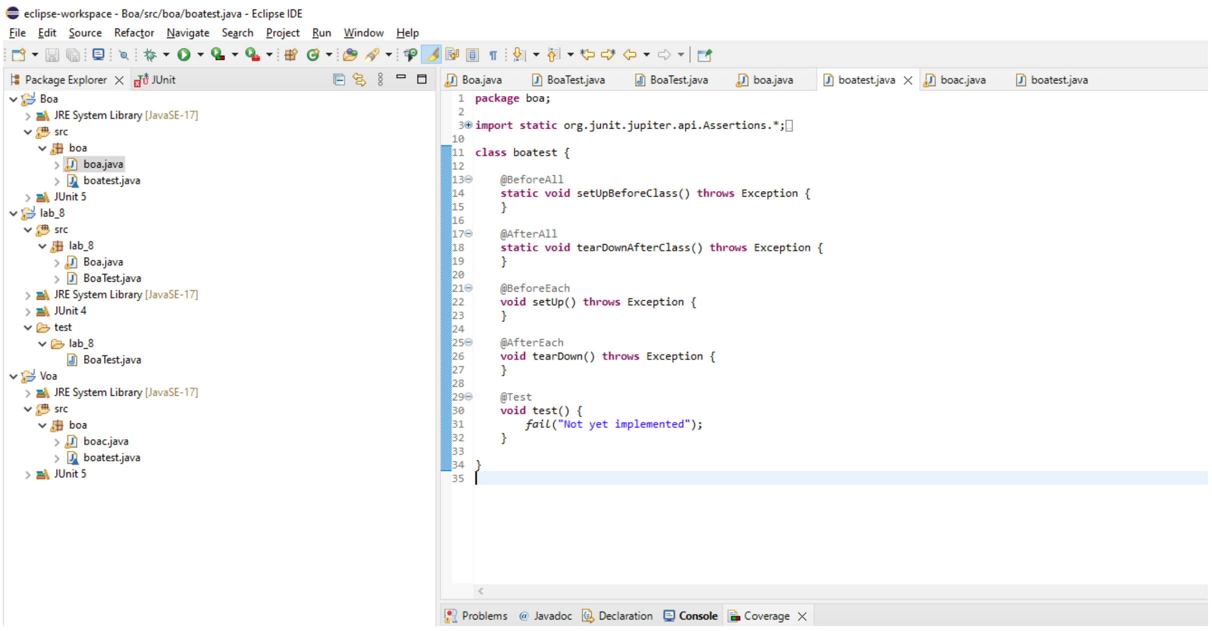
2. Create a class for a Boa. Here's the code you can use (you may copy/paste):

```
// 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;
}
}
```


eclipse-workspace - Boa/src/boa/boa.java - Eclipse IDE

```
package boa;
public class boa {
        private String name;
        private int length;
        private String favoriteFood;

        public  boa (String name, int length, String favoriteFooc
            this.name = name;
            this.length = length;
            this.favoriteFood = favoriteFood;
        }
        public boolean isHealthy() {

            return this.favoriteFood.equals("granola bars");
        }
        public boolean fitsInCage(int cageLength) {


            return this.length <cageLength;
        }


}
```

3. Follow the instructions in the JUnit tutorial in the section "Creating a JUnit Test Case in Eclipse". You'll be creating a test case for the class Boa. When you're asked to select test method stubs, select both isHealthy() and fitsInCage(int).

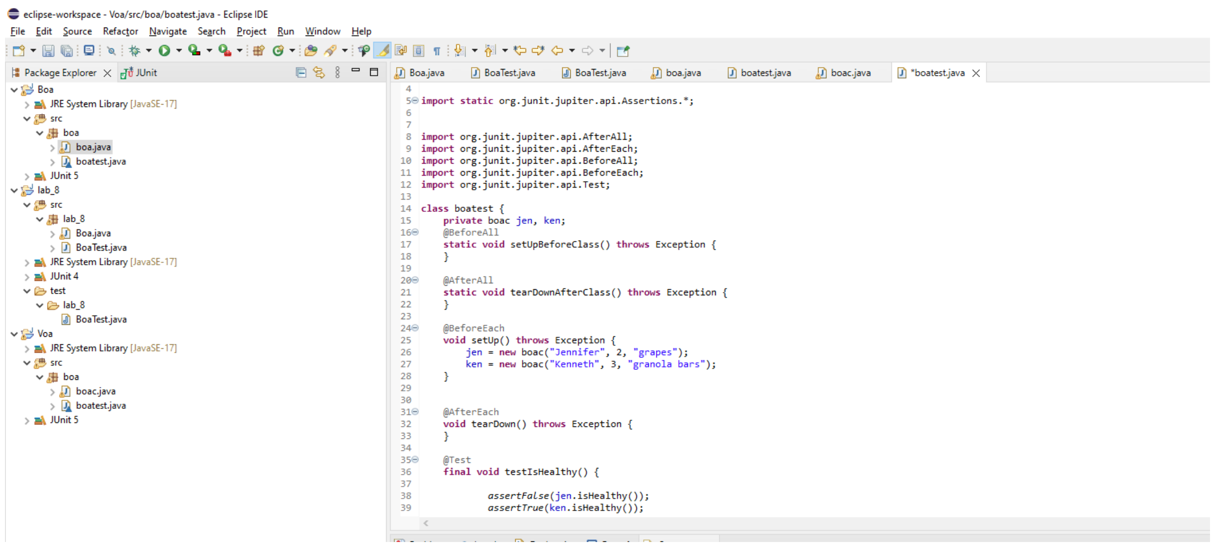

eclipse-workspace - Boa/src/boa/boatest.java - Eclipse IDE

```
package boa;

import static org.junit.jupiter.api.Assertions.*;

class boatest {

    @BeforeAll
    static void setUpBeforeClass() throws Exception {
    }

    @AfterAll
    static void tearDownAfterClass() throws Exception {
    }

    @BeforeEach
    void setUp() throws Exception {
    }

    @AfterEach
    void tearDown() throws Exception {
    }

    @Test
    void test() {
        fail("Not yet implemented");
    }

}
```

Here created a test file as shown above

4. Now it's time to write some unit tests. Notice that the BoaTest class that JUnit created for you contains stubs for several methods. The first stub (for the method setUp()) is annotated with @Before. The @Before annotation denotes that the method setUp() will be run prior to the execution of each test method. setUp() is typically used to initialise data needed by each test. Modify the setUp() method so that it creates a couple of Boa objects, as follows:

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


eclipse-workspace - Voa/src/boa/boatest.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
JUnit
Boa.java
BoaTest.java
BoaTest.java
boa.java
boatest.java
boac.java
*boatest.java

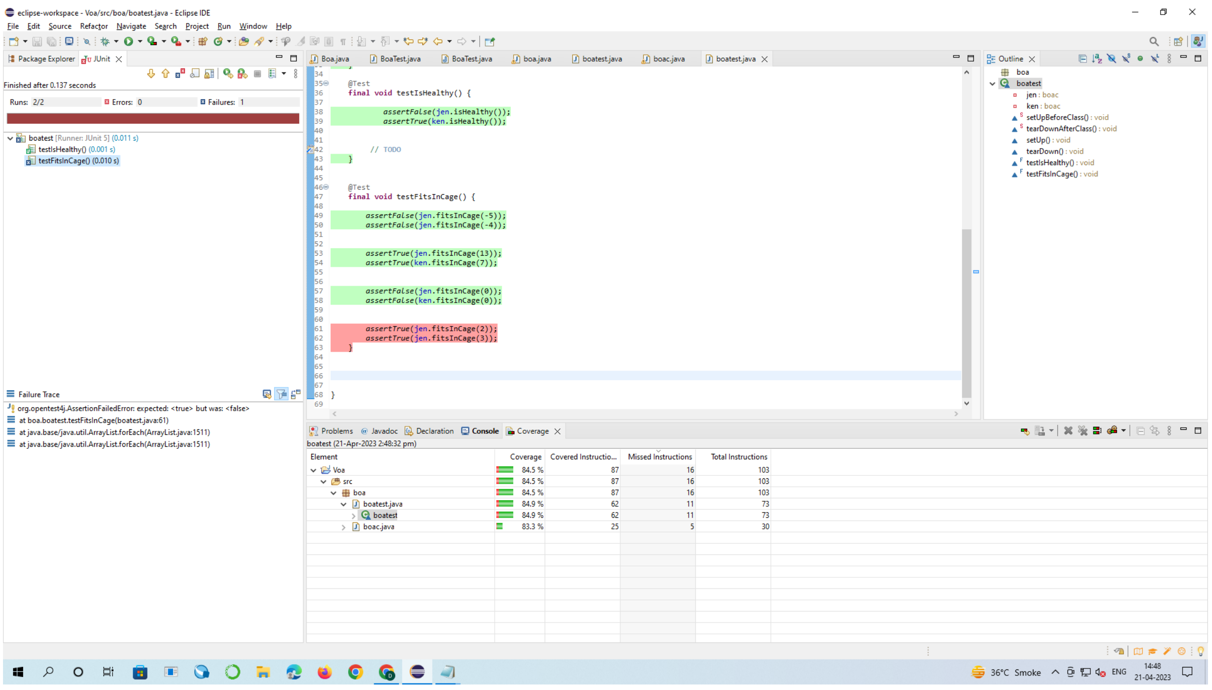
```
Boa
  JRE System Library [JavaSE-17]
  src
    boa
      boa.java
      boatest.java
  JUnit 5
lab_8
  src
    lab_8
      Boa.java
      BoaTest.java
  JRE System Library [JavaSE-17]
  JUnit 4
  test
    lab_8
      BoaTest.java
Voa
  JRE System Library [JavaSE-17]
  src
    boa
      boac.java
      boatest.java
  JUnit 5
```

```java

import static org.junit.jupiter.api.Assertions.*;


import org.junit.jupiter.api.AfterAll;
import org.junit.jupiter.api.AfterEach;
import org.junit.jupiter.api.BeforeAll;
import org.junit.jupiter.api.BeforeEach;
import org.junit.jupiter.api.Test;

class boatest {
    private boac jen, ken;
    @BeforeAll
    static void setUpBeforeClass() throws Exception {
    }

    @AfterAll
    static void tearDownAfterClass() throws Exception {
    }

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


    @AfterEach
    void tearDown() throws Exception {
    }

    @Test
    final void testIsHealthy() {

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

It is important to note that the *isHealthy()* function does not take any parameters. Due to this, the only test cases possible are to call the method on *Boa* objects *"jen"* and *"ken"*.

The *FitsInCage()* function takes an integer parameter as input and hence can have a range of test-cases possible.
Although *jen* and *ken* both will be executing the same function definition for *FitsInCage(<cage_length>)* function call, it is better to call methods from both the instances to ensure independence of the function definition from the instance.

5. JUnit also provided stubs for two test methods, each annotated with @Test. Work on the testIsHealthy() method first. The purpose of this method is to check that the isHealthy() method in the Boa class behaves the way it's supposed to. In the JUnit tutorial, read the section on "Writing Tests". Modify the testIsHealthy() method so that it checks the results of activating the isHealthy() method on the two Boa objects you created in setup(). Likewise, modify the testFitsInCage() method to test the results of that method. Make sure your test is robust; it should check the results when the cage length is less than the length of the boa, when the cage length is equal to the length of the boa, and when the cage length is greater than the length of the boa. Should you write tests for both jen and ken?


eclipse-workspace - Voa/src/boa/boatest.java - Eclipse IDE

```
	@Test
	final void testIsHealthy() {

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


		// TODO
	}


	@Test
	final void testFitsInCage() {

		assertFalse(jen.fitsInCage(-5));
		assertFalse(jen.fitsInCage(-4));


		assertTrue(jen.fitsInCage(13));
		assertTrue(ken.fitsInCage(7));


		assertFalse(jen.fitsInCage(0));
		assertFalse(ken.fitsInCage(0));


		assertTrue(jen.fitsInCage(2));
		assertTrue(jen.fitsInCage(3));
	}



}
```

```
Finished after 0.137 seconds
Runs: 2/2    Errors: 0    Failures: 1

boatest [Runner: JUnit 5] (0.011 s)
    testIsHealthy() (0.001 s)
    testFitsInCage() (0.010 s)

Failure Trace
org.opentest4j.AssertionFailedError: expected: <true> but was: <false>
at boa.boatest.testFitsInCage(boatest.java:61)
at java.base/java.util.ArrayList.forEach(ArrayList.java:1511)
at java.base/java.util.ArrayList.forEach(ArrayList.java:1511)
```

boatest (21-Apr-2023 2:48:32 pm)

Element	Coverage	Covered Instructio...	Missed Instructions	Total Instructions
Voa	84.5 %	87	16	103
src	84.5 %	87	16	103
boa	84.5 %	87	16	103
boatest.java	84.9 %	62	11	73
boatest	84.9 %	62	11	73
boac.java	83.3 %	25	5	30

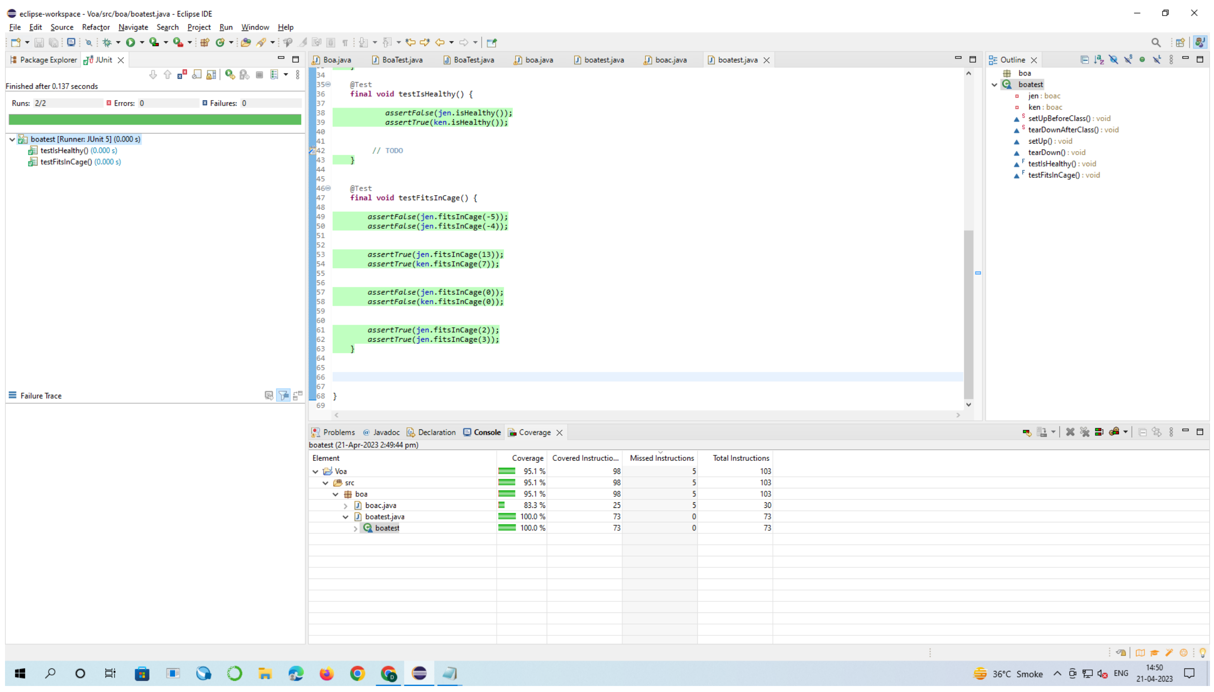
Here the error is coming from public boolean fitsInCage(int cageLength) {

```
            return this.length <cageLength;
        }
```

Here if we would change into

```
public boolean fitsInCage(int cageLength) {

            return this.length <=cageLength;
```

6.Now you can run your tests. Read the section “Running Your Test Case” in the tutorial. Did you get a green bar in the JUnit pane? If you got a red bar, use the output in the JUnit pane to determine which test(s) failed. Fix your tests, and try running the test case again.
It’s important to note that a red bar doesn’t necessarily mean that the test case is written incorrectly; it could be that the method that’s being tested isn’t correct. In fact, that’s what unit testing is supposed to do – help us find errors in our code. When a test Then fails, you need to determine if the error is in the test case itself or in the code it’s Testing.


eclipse-workspace - Voa/src/boa/boatest.java - Eclipse IDE

Finished after 0.137 seconds

Runs: 2/2   Errors: 0   Failures: 0

```
boatest [Runner: JUnit 5] (0.000 s)
    testIsHealthy() (0.000 s)
    testFitsInCage() (0.000 s)
```

```
    @Test
    final void testIsHealthy() {

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


        // TODO
    }


    @Test
    final void testFitsInCage() {

        assertFalse(jen.fitsInCage(-5));
        assertFalse(jen.fitsInCage(-4));


        assertTrue(jen.fitsInCage(13));
        assertTrue(ken.fitsInCage(7));


        assertFalse(jen.fitsInCage(0));
        assertFalse(ken.fitsInCage(0));


        assertTrue(jen.fitsInCage(2));
        assertTrue(jen.fitsInCage(3));
    }




}
```

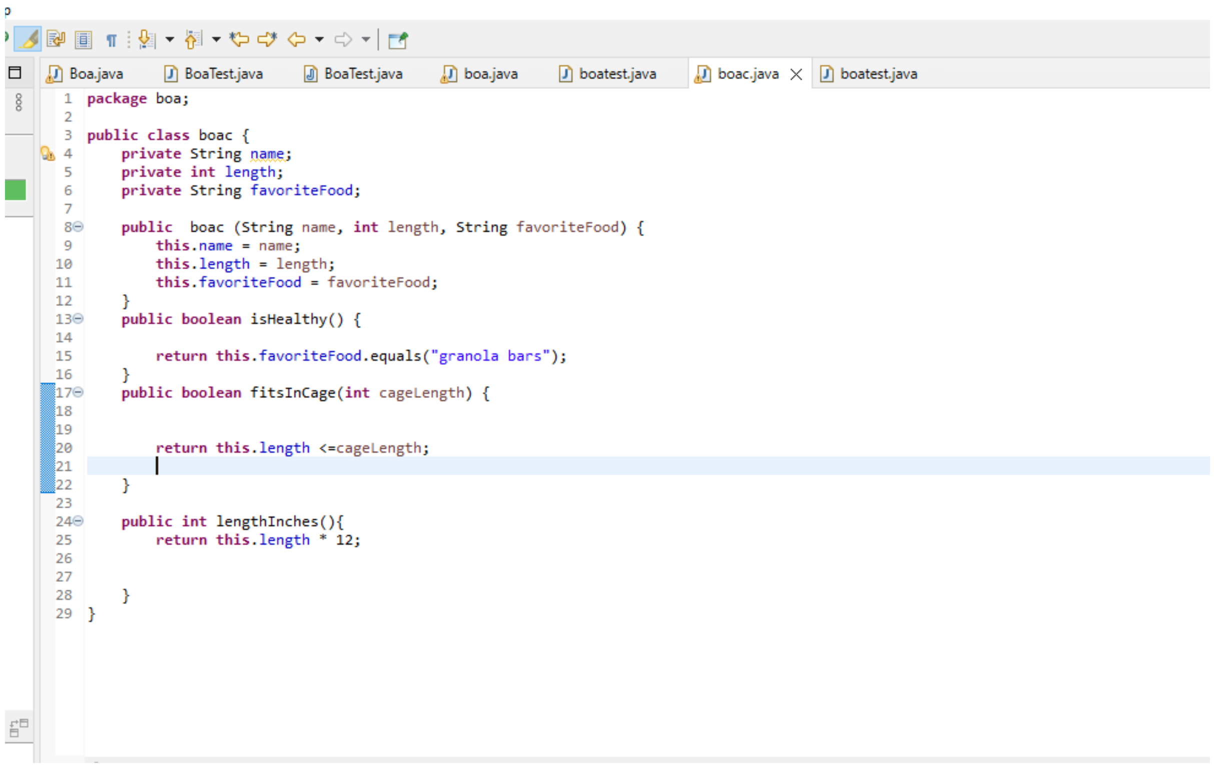
boatest (21-Apr-2023 2:49:44 pm)

Element	Coverage	Covered Instructio...	Missed Instructions	Total Instructions
Voa	95.1 %	98	5	103
src	95.1 %	98	5	103
boa	95.1 %	98	5	103
boac.java	83.3 %	25	5	30
boatest.java	100.0 %	73	0	73
boatest	100.0 %	73	0	73

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

// produces the length of the Boa in inches
public int lengthInInches(){
// you need to write the body of this method
}
Add a new test case to the BoaTest class that tests the lengthInInches() method. Make
sure you annotate the new test method with @Test. Run your tests.

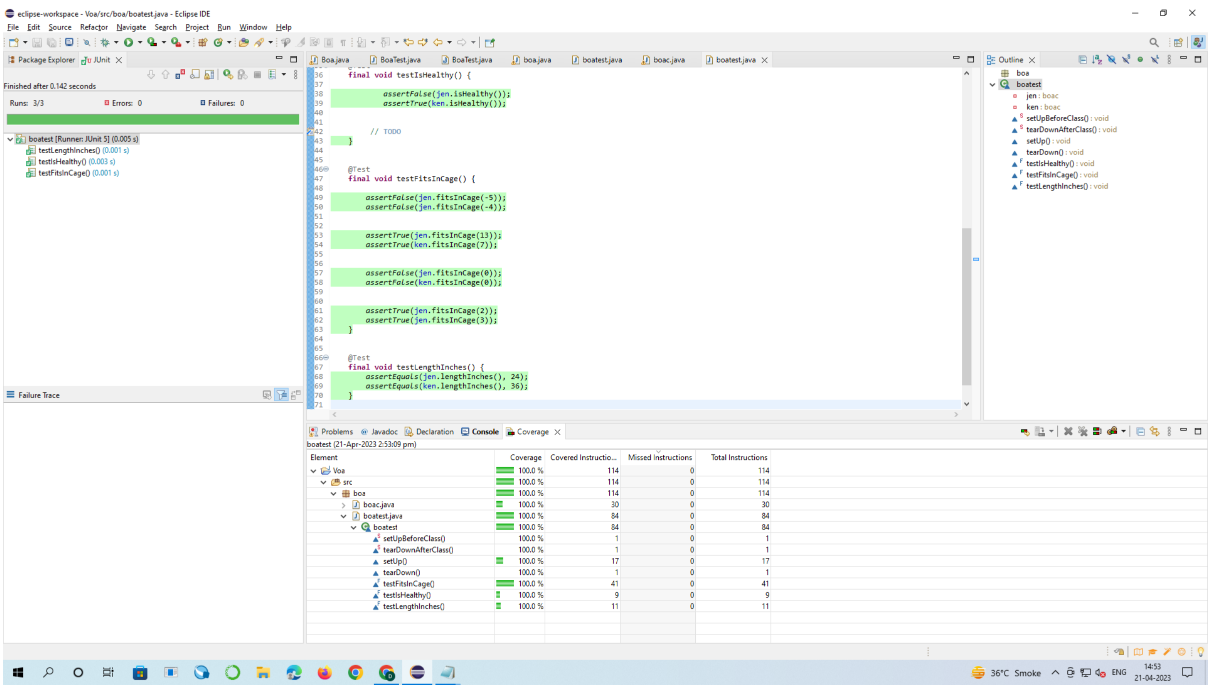
```
public int lengthInches(){
            return this.length * 12;


        }
```



```
package boa;

public class boac {
    private String name;
    private int length;
    private String favoriteFood;

    public  boac (String name, int length, String favoriteFood) {
        this.name = name;
        this.length = length;
        this.favoriteFood = favoriteFood;
    }
    public boolean isHealthy() {

        return this.favoriteFood.equals("granola bars");
    }
    public boolean fitsInCage(int cageLength) {


        return this.length <=cageLength;

    }

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


    }
}
```

8. Here are some other things you should know about unit testing and JUnit: Each method annotated with @Test will be run, but the order of the tests is not guaranteed. Any method annotated with @Before will be run before each test executes. Any method annotated with @After will be run after each test executes.

```
@Test
        final void testLengthInches() {
            assertEquals(jen.lengthInches(), 24);
            assertEquals(ken.lengthInches(), 36);
        }
```



eclipse-workspace - Voa/src/boa/boatest.java - Eclipse IDE

Finished after 0.142 seconds

Runs: 3/3   Errors: 0   Failures: 0

```
boatest [Runner: JUnit 5] (0.005 s)
    testLengthInches() (0.001 s)
    testIsHealthy() (0.003 s)
    testFitsInCage() (0.001 s)
```

```
    final void testIsHealthy() {

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


        // TODO
    }


    @Test
    final void testFitsInCage() {

        assertFalse(jen.fitsInCage(-5));
        assertFalse(jen.fitsInCage(-4));


        assertTrue(jen.fitsInCage(13));
        assertTrue(ken.fitsInCage(7));


        assertFalse(jen.fitsInCage(0));
        assertFalse(ken.fitsInCage(0));


        assertTrue(jen.fitsInCage(2));
        assertTrue(jen.fitsInCage(3));
    }


    @Test
    final void testLengthInches() {
        assertEquals(jen.lengthInches(), 24);
        assertEquals(ken.lengthInches(), 36);
    }
```

```
boa
boatest
    jen : boac
    ken : boac
    setUpBeforeClass() : void
    tearDownAfterClass() : void
    setUp() : void
    tearDown() : void
    testIsHealthy() : void
    testFitsInCage() : void
    testLengthInches() : void
```

boatest (21-Apr-2023 2:53:09 pm)

Element	Coverage	Covered Instructio...	Missed Instructions	Total Instructions
Voa	100.0 %	114	0	114
src	100.0 %	114	0	114
boa	100.0 %	114	0	114
boac.java	100.0 %	30	0	30
boatest.java	100.0 %	84	0	84
boatest	100.0 %	84	0	84
setUpBeforeClass()	100.0 %	1	0	1
tearDownAfterClass()	100.0 %	1	0	1
setUp()	100.0 %	17	0	17
tearDown()	100.0 %	1	0	1
testFitsInCage()	100.0 %	41	0	41
testIsHealthy()	100.0 %	9	0	9
testLengthInches()	100.0 %	11	0	11