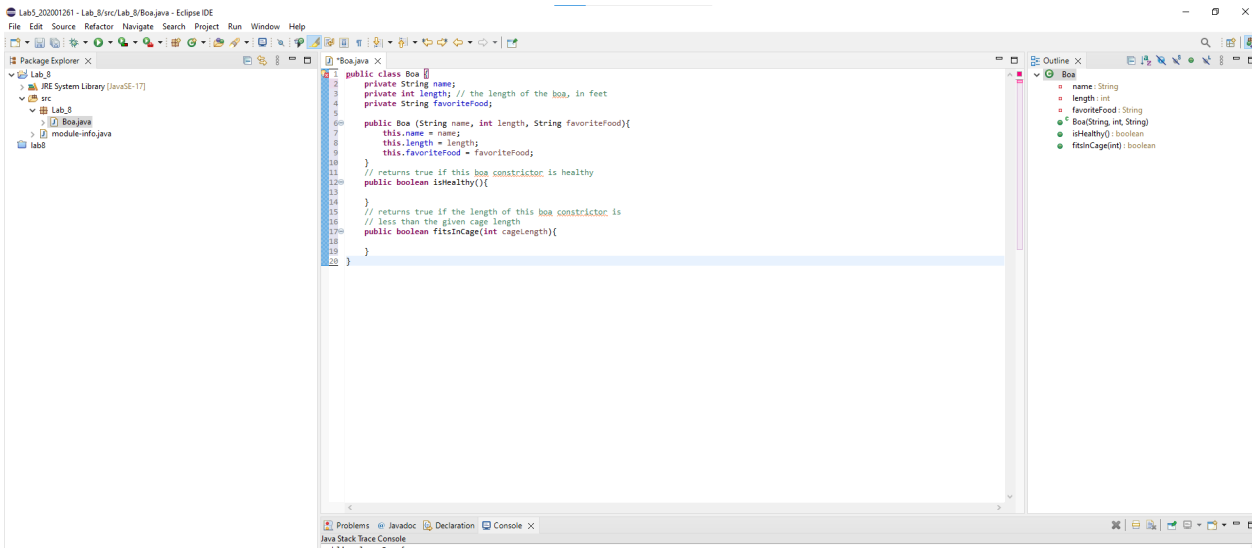
IT 314
LAB 8
Junit Testing Framework

Name: Priya Vinda **ID: 202001261**

Aim: The goal of this lab is to learn how to use JUnit to write unit tests for Java programs.

Lab Exercises

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



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

```
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
```

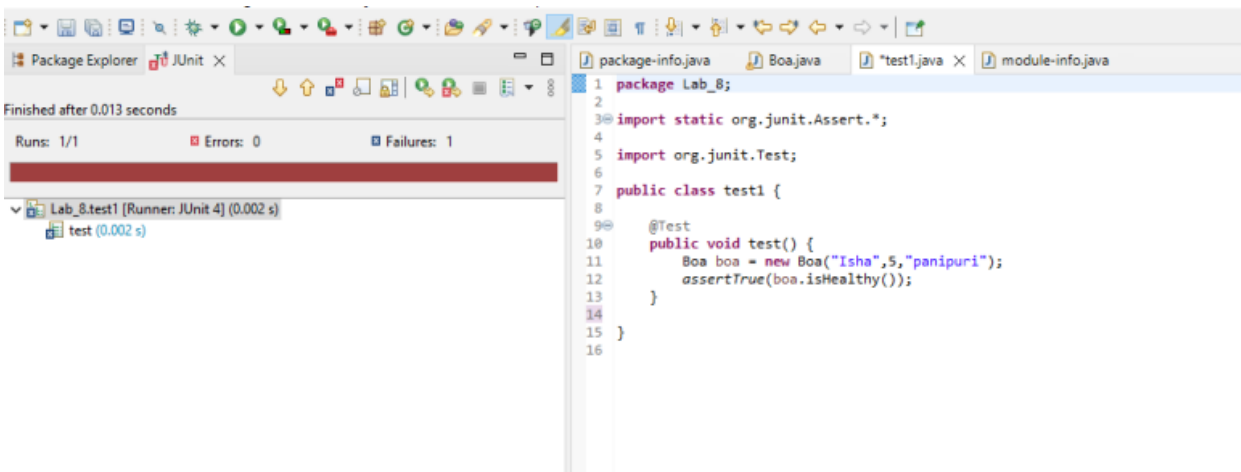
```
	public boolean isHealthy(){
		return this.favoriteFood.equals("granola bars");
		}
	// returns true if the length of this boa constrictor is less than the given cage length

	public boolean fitsInCage(int cageLength){
		return this.length < cageLength;
	}
}
```

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).

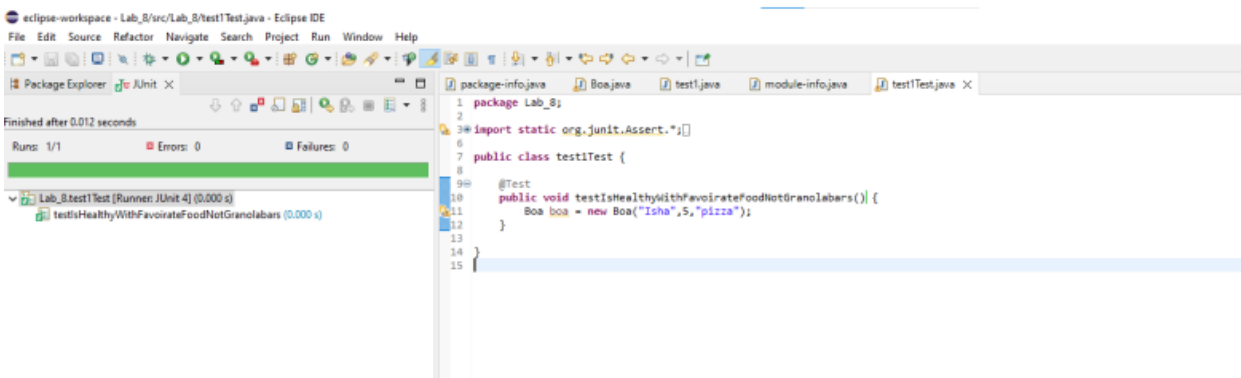
Code:

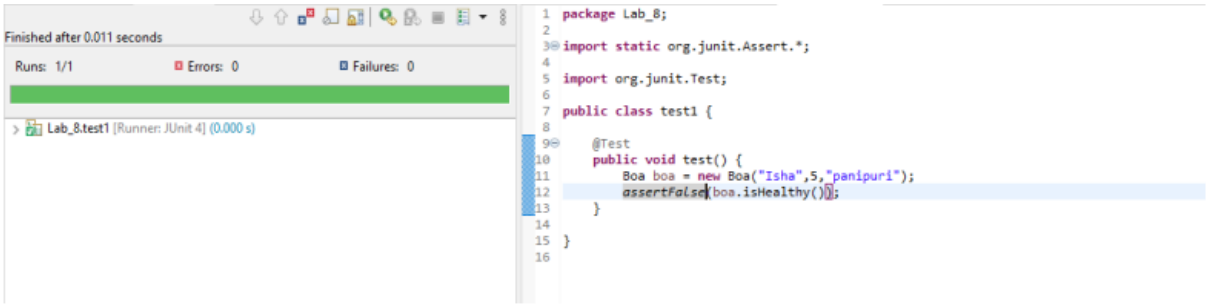
```
package Lab_8;
import static org.junit.Assert.*;
import org.junit.Test;
public class test1 {
@Test
public void isHealthyforgranolabarsfood() {
Boa boa = new Boa("Isha",5,"granola bars");
assertTrue(boa.isHealthy());
}
public void isHealthyforgranolabarsnotasfavoiratefood() {
Boa boa = new Boa("Isha",5,"panipuri");
assertFalse(boa.isHealthy());
}
public void testFitsInCagewhenLengthLessThanCageLength() {
Boa boa = new Boa("Isha",5,"granola bars");
assertTrue(boa.fitsInCage(10));
}

public void testFitsInCagewhenLengthGreaterThanCageLength() {
Boa boa = new Boa("Isha",5,"granola bars");
assertFalse(boa.fitsInCage(3));
}
}
```

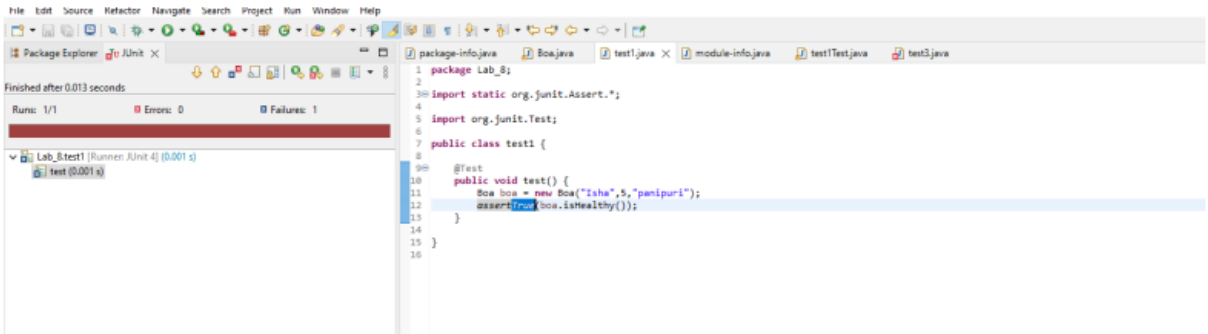
Test1:


Package Explorer
JUnit
Finished after 0.013 seconds
Runs: 1/1
Errors: 0
Failures: 1
Lab_8.test1 [Runner: JUnit 4] (0.002 s)
test (0.002 s)
package-info.java
Boa.java
*test1.java
module-info.java
package Lab_8;
import static org.junit.Assert.*;
import org.junit.Test;
public class test1 {
@Test
public void test() {
Boa boa = new Boa("Isha",5,"panipuri");
assertTrue(boa.isHealthy());
}
}

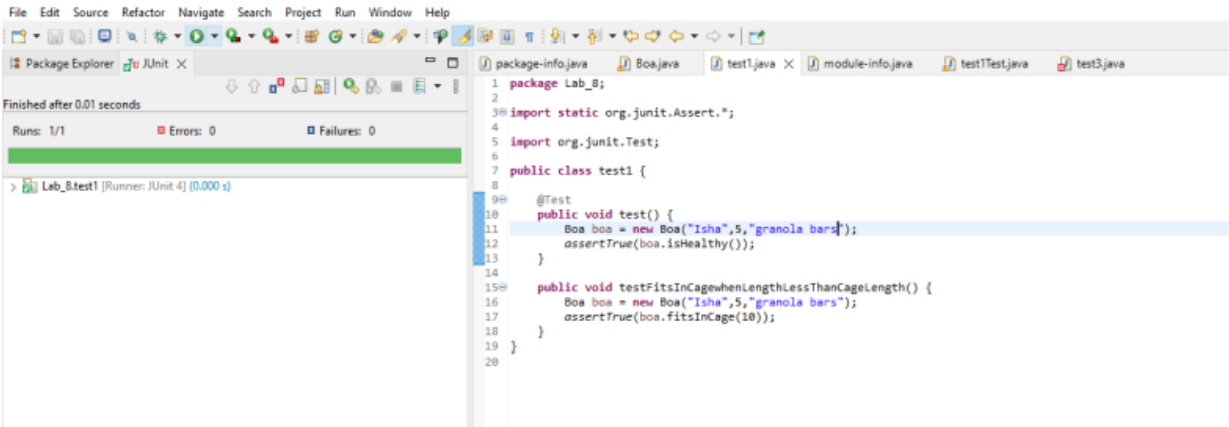
Test 2:


eclipse-workspace - Lab_8/src/Lab_8/test1Test.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
JUnit
Finished after 0.012 seconds
Runs: 1/1
Errors: 0
Failures: 0
Lab_8.test1Test [Runner: JUnit 4] (0.000 s)
testIsHealthyWithFavoirateFoodNotGranolabars (0.000 s)
package-info.java
Boa.java
test1.java
module-info.java
test1Test.java
package Lab_8;
import static org.junit.Assert.*;
public class test1Test {
@Test
public void testIsHealthyWithFavoirateFoodNotGranolabars() {
Boa boa = new Boa("Isha",5,"pizza");
}
}

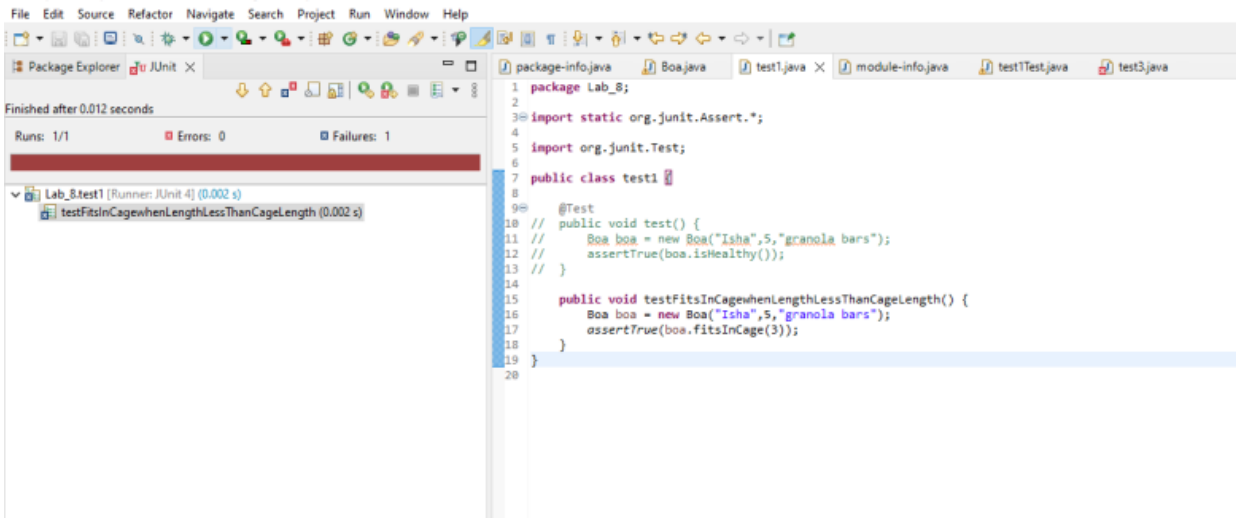




This test case works correct:



This test case fails:

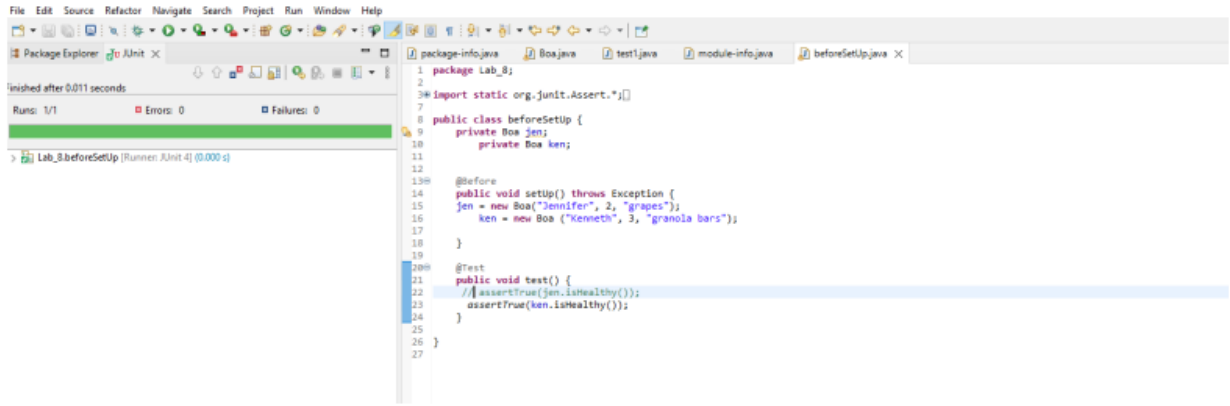


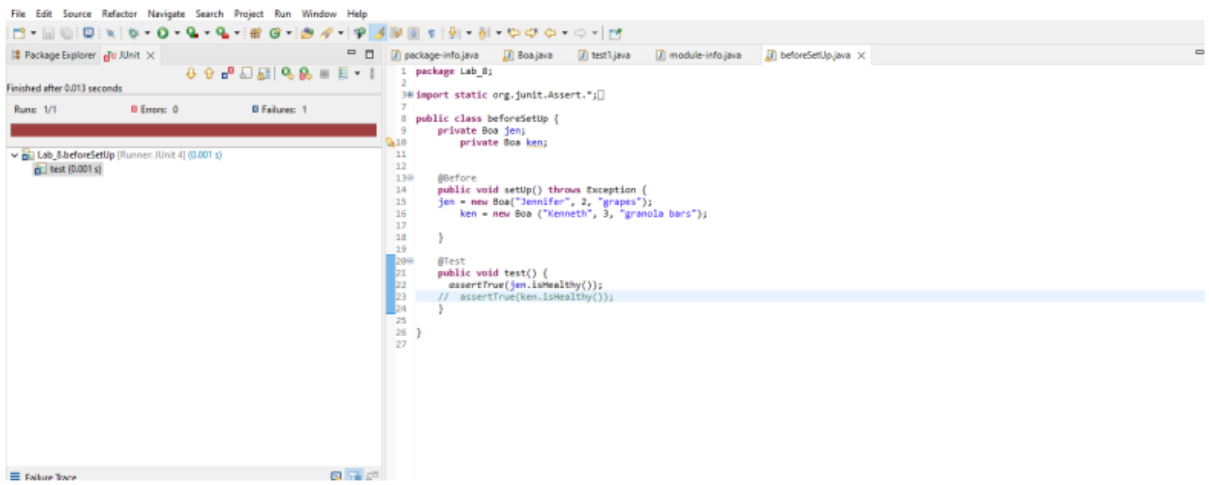
4. Now it's time to write some unit tests. Notice that the BoaTest class that JUnit created for you contains stubs for several methods. The first stub (for the method setUp()) is annotated with @Before. The @Before annotation denotes that the method setUp() will be run prior to the execution of each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

Code:

```
import static org.junit.Assert.*;
import org.junit.Before;
import org.junit.Test;

public class beforeSetUp {
        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 test() {
                assertTrue(jen.isHealthy());
                assertTrue(ken.isHealthy());
                assertTrue(jen.fitsInCage(2));
                assertTrue(ken.fitsInCage(10));
        }
```

}





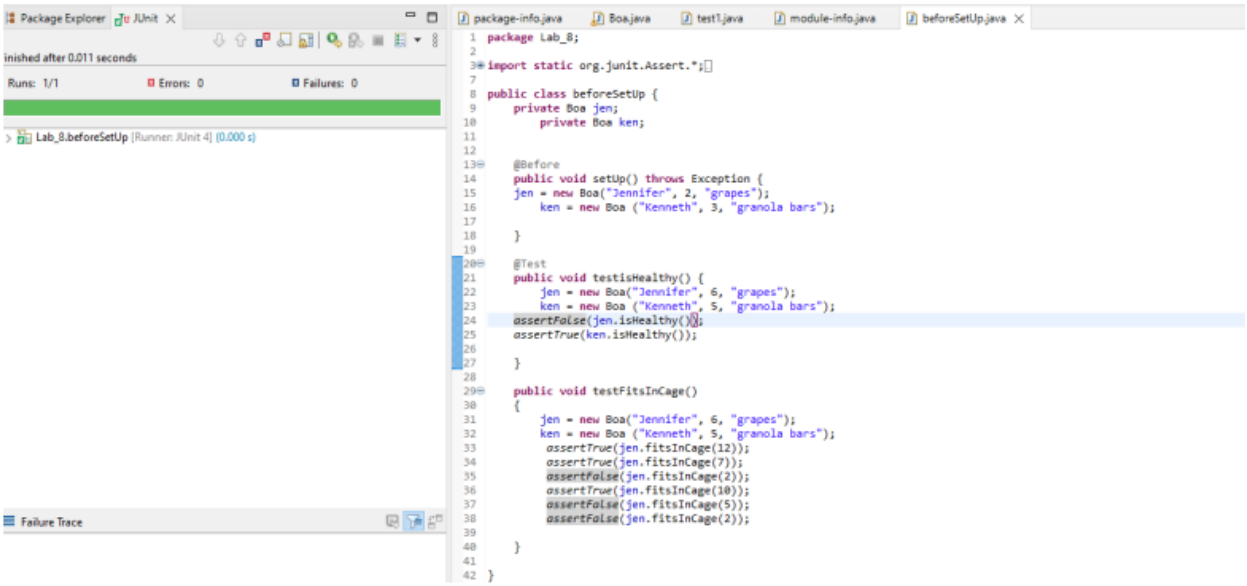
Here, Jen's favorite food is grapes which are not healthy, which is shown in the red bar.

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?

Code:

```
@Test
public void testisHealthy() {
        jen = new Boa("Jennifer", 6, "grapes");
        ken = new Boa ("Kenneth", 5, "granola bars");
        assertFalse(jen.isHealthy());
        assertTrue(ken.isHealthy());
}
public void testFitsInCage()
{
        jen = new Boa("Jennifer", 6, "grapes");
        ken = new Boa ("Kenneth", 5, "granola bars");
        assertTrue(jen.fitsInCage(12));
        assertTrue(jen.fitsInCage(7));
        assertFalse(jen.fitsInCage(2));
        assertTrue(jen.fitsInCage(10));
        assertFalse(jen.fitsInCage(5));
        assertFalse(jen.fitsInCage(2));
}
```

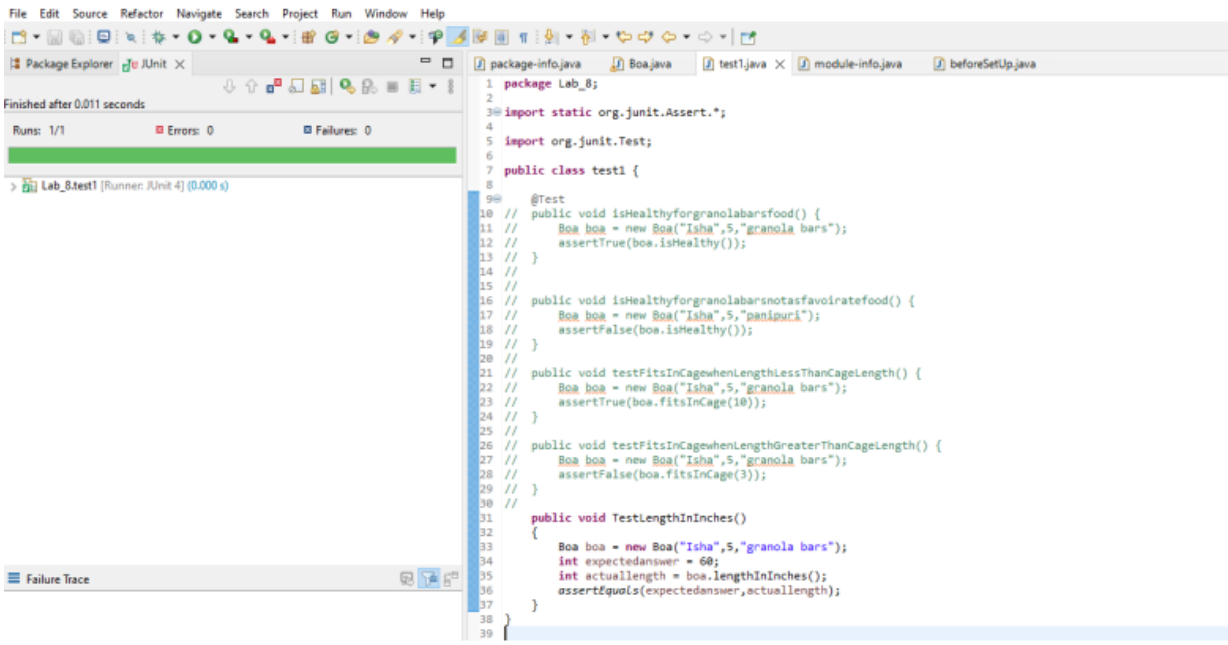


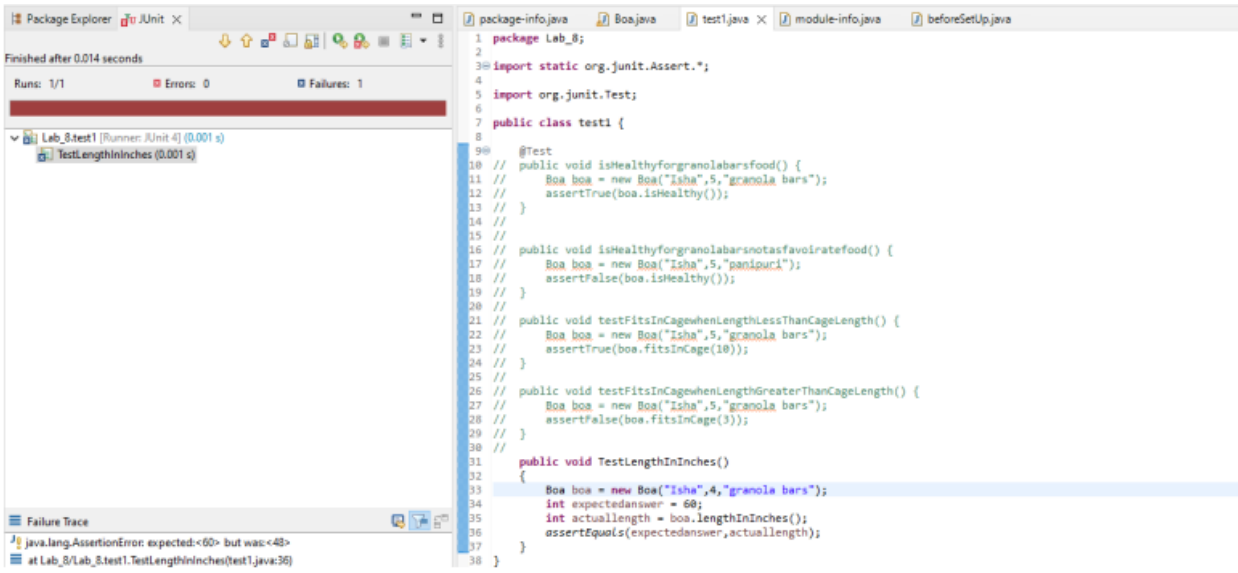
7. Add a new method to the Boa class, with this purpose and signature:
// produces the length of the Boa in inches

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.

Code added:

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





Code:

```
@Test
public void TestLengthInInches()
{
        Boa boa = new Boa("Isha",4,"granola bars");
        int expectedanswer = 60;
        int actuallength = boa.lengthInInches();
        assertEquals(expectedanswer,actuallength);
}
```