

IT 314 Software Engineering

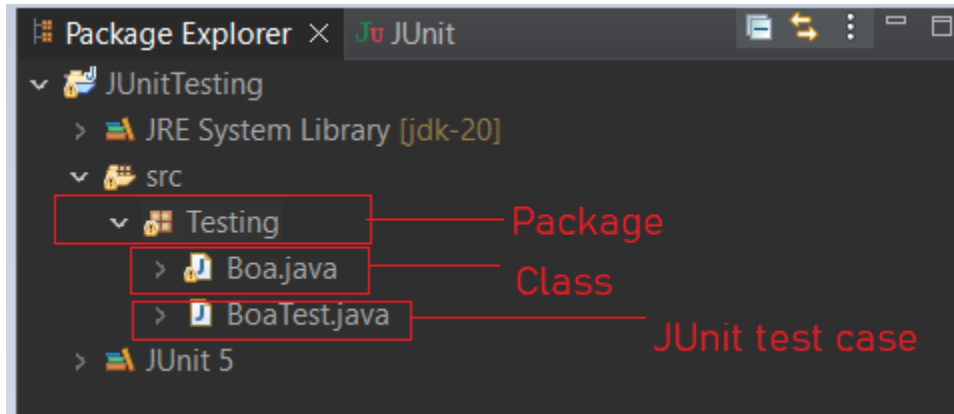
Lab 8

Name: Parikh Riki Anilbhai

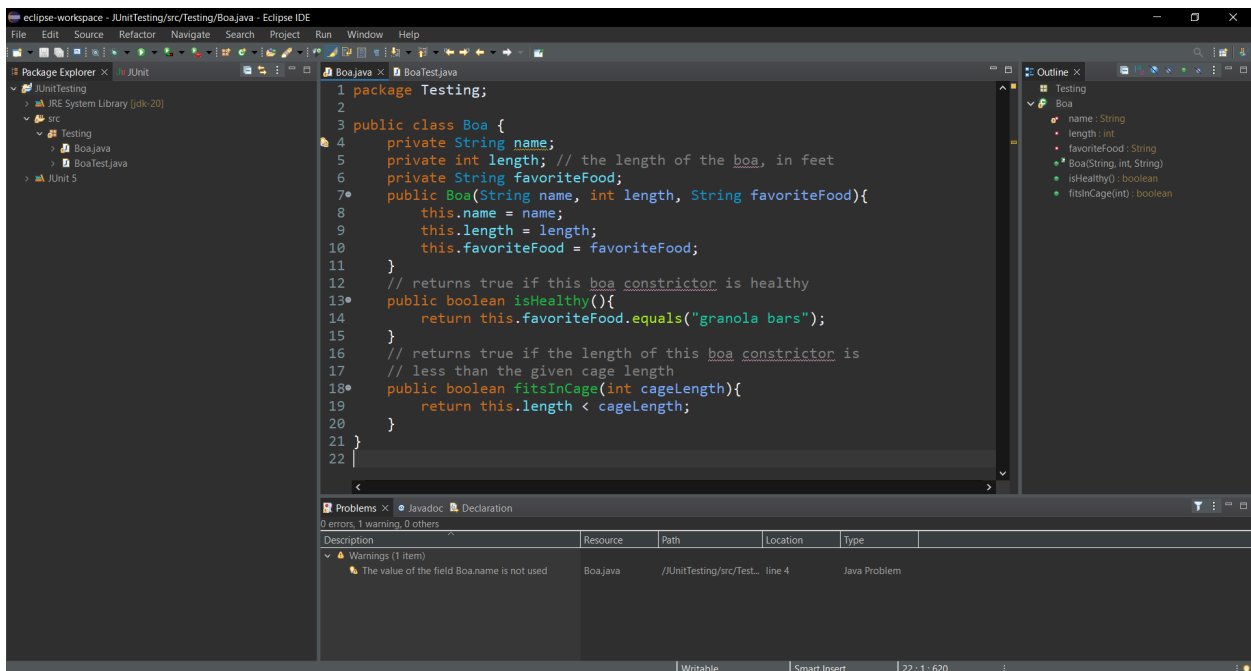
ID: 202001241

Lab Exercises:

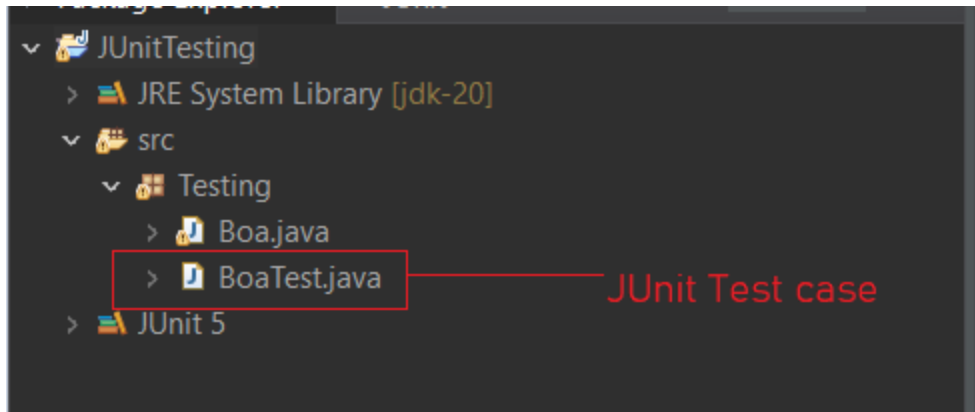
1. Create a new Eclipse project, and within the project, create a package.



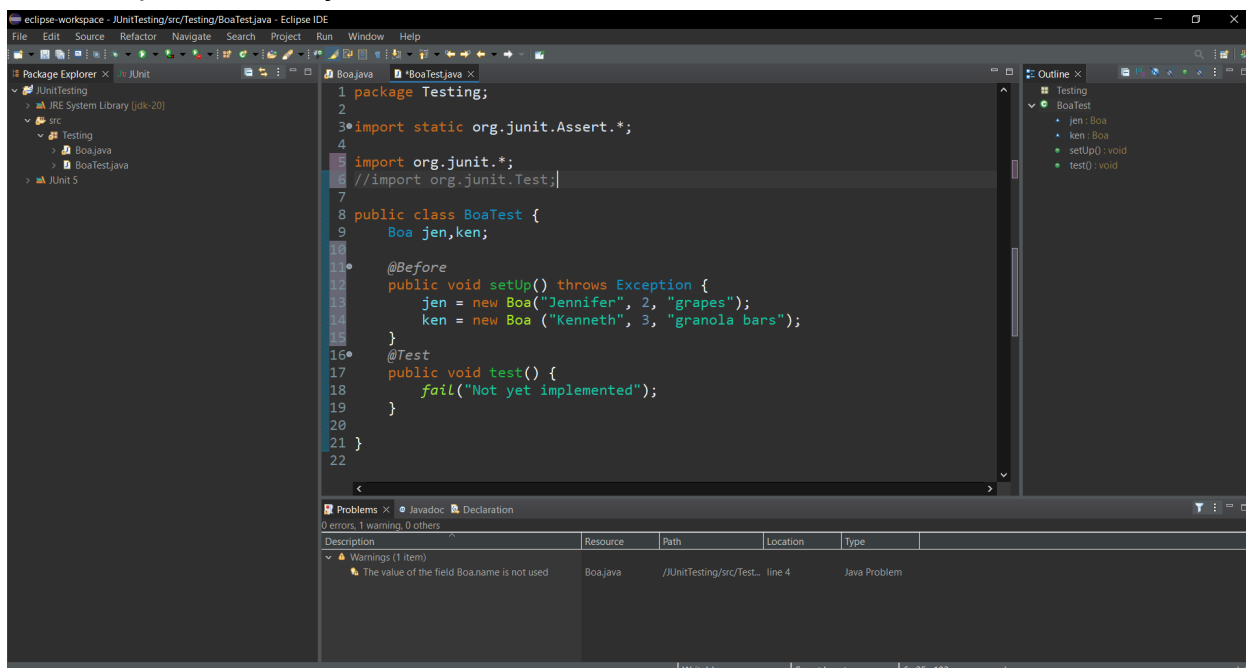
2. Create a class for a Boa. Here's the code you can use:



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.



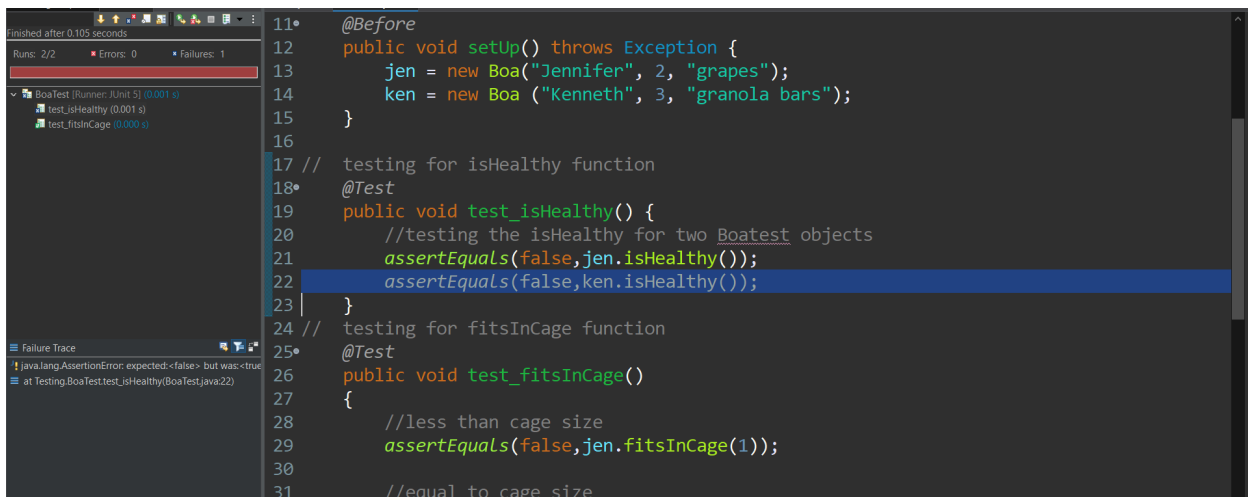
4. Now, it's time to write some unit tests. Notice that the BoaTest class that JUnit created for you contains stubs for several methods. The first stub (for the method `setUp()`) is annotated with `@Before`. The `@Before` annotation denotes that the method `setUp()` will be run prior to the execution 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, as follows:



5. Now we have to implement the `testIsHealthy()` and `testFitsInCage()` functions in the “BoaTest” class

```
16
17 // testing for isHealthy function
18* @Test
19 public void test_isHealthy() {
20     //testing the isHealthy for two Boatest objects
21     assertEquals(false,jen.isHealthy());
22     assertEquals(false,ken.isHealthy());
23 }
24 // testing for fitsInCage function
25* @Test
26 public void test_fitsInCage()
27 {
28     //less than cage size
29     assertEquals(false,jen.fitsInCage(1));
30
31     //equal to cage size
32     assertEquals(false,jen.fitsInCage(2));
33
34     //greater than cage size
35     assertEquals(true,jen.fitsInCage(3));
36
37     //some other test cases
38     assertEquals(true,jen.fitsInCage(4));
39     assertEquals(true,jen.fitsInCage(1000));
40     assertEquals(false,jen.fitsInCage(-10));
41     assertEquals(true,ken.fitsInCage(90000));
42     assertEquals(false,jen.fitsInCage(-1));
43     assertEquals(false,ken.fitsInCage(2));
44     assertEquals(false,ken.fitsInCage(1));
```

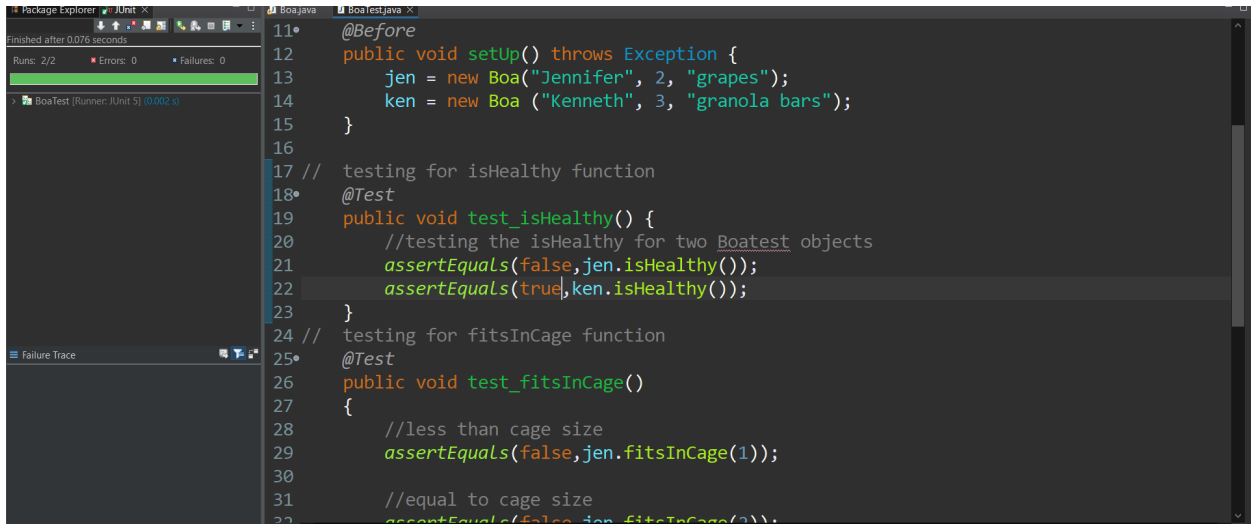
6. Run your tests.



```
11* @Before
12 public void setUp() throws Exception {
13     jen = new Boa("Jennifer", 2, "grapes");
14     ken = new Boa ("Kenneth", 3, "granola bars");
15 }
16
17 // testing for isHealthy function
18* @Test
19 public void test_isHealthy() {
20     //testing the isHealthy for two Boatest objects
21     assertEquals(false,jen.isHealthy());
22     assertEquals(false,ken.isHealthy());
23 }
24 // testing for fitsInCage function
25* @Test
26 public void test_fitsInCage()
27 {
28     //less than cage size
29     assertEquals(false,jen.fitsInCage(1));
30
31     //equal to cage size
```

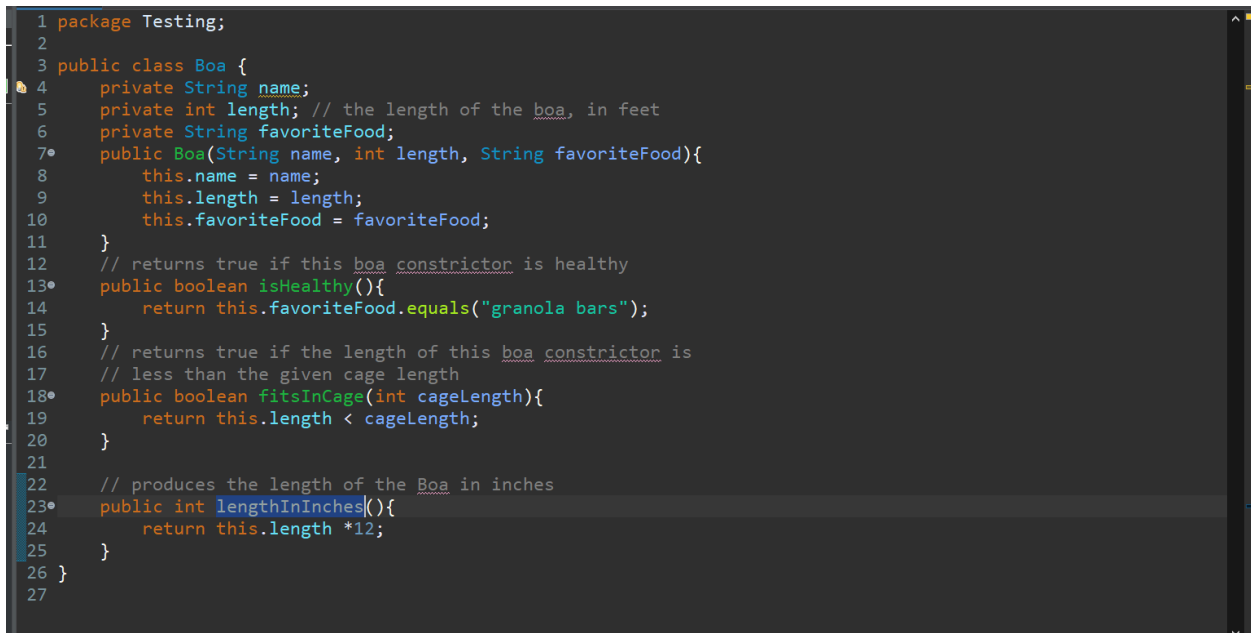
Able to find the failure in line 22 because `ken.isHealth()`=true, but in the code, it is false, so got a failure

So if we change line 22 to the correct one, we should not get any errors



```
11 * @Before
12 public void setUp() throws Exception {
13     jen = new Boa("Jennifer", 2, "grapes");
14     ken = new Boa("Kenneth", 3, "granola bars");
15 }
16
17 // testing for isHealthy function
18 * @Test
19 public void test_isHealthy() {
20     //testing the isHealthy for two Boatest objects
21     assertEquals(false, jen.isHealthy());
22     assertEquals(true, ken.isHealthy());
23 }
24 // testing for fitsInCage function
25 * @Test
26 public void test_fitsInCage()
27 {
28     //less than cage size
29     assertEquals(false, jen.fitsInCage(1));
30
31     //equal to cage size
32     assertEquals(false, ken.fitsInCage(3));
```

7. Add a new method to the Boa class called lengthInInches(). Add a new test case to the BoaTest class that tests the lengthInInches() method.



```
1 package Testing;
2
3 public class Boa {
4     private String name;
5     private int length; // the length of the boa, in feet
6     private String favoriteFood;
7 * public Boa(String name, int length, String favoriteFood){
8     this.name = name;
9     this.length = length;
10    this.favoriteFood = favoriteFood;
11 }
12 // returns true if this boa constructor is healthy
13 * public boolean isHealthy(){
14     return this.favoriteFood.equals("granola bars");
15 }
16 // returns true if the length of this boa constructor is
17 // less than the given cage length
18 * public boolean fitsInCage(int cageLength){
19     return this.length < cageLength;
20 }
21
22 // produces the length of the Boa in inches
23 * public int lengthInInches(){
24     return this.length * 12;
25 }
26 }
27
```

Test Cases and Testing:

Finished after 0.077 seconds

Runs: 3/3 Errors: 0 Failures: 0

BoaTest (Runner: JUnit 5) (0.002 s)

test_lengthInInches (0.001 s)

test_isHealthy (0.000 s)

test_fitInCage (0.001 s)

Failure Trace

```
29     assertEquals(false,jen.fitsInCage(1));
30
31     //equal to cage size
32     assertEquals(false,jen.fitsInCage(2));
33
34     //greater than cage size
35     assertEquals(true,jen.fitsInCage(3));
36
37     //some other test cases
38     assertEquals(true,jen.fitsInCage(4));
39     assertEquals(true,jen.fitsInCage(1000));
40     assertEquals(false,jen.fitsInCage(-10));
41     assertEquals(true,ken.fitsInCage(90000));
42     assertEquals(false,jen.fitsInCage(-1));
43     assertEquals(false,ken.fitsInCage(2));
44     assertEquals(false,ken.fitsInCage(1));
45     assertEquals(true,ken.fitsInCage(12020));
46
47 }
48
49 @Test
50 public void test_lengthInInches()
51 {
52     assertEquals(24,jen.lengthInInches());
53     assertEquals(36,ken.lengthInInches());
54 }
55 }
56
```