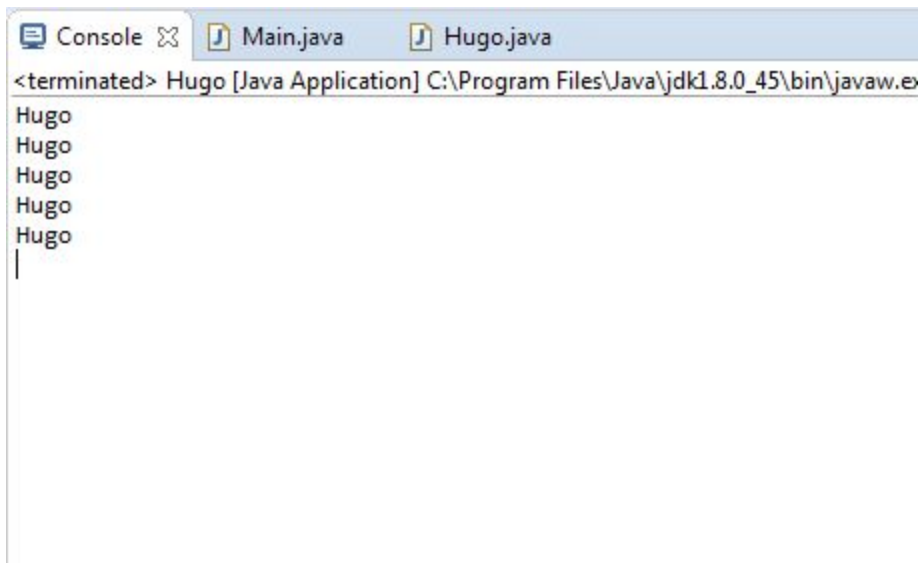


Student No.s : 960689 & 951428

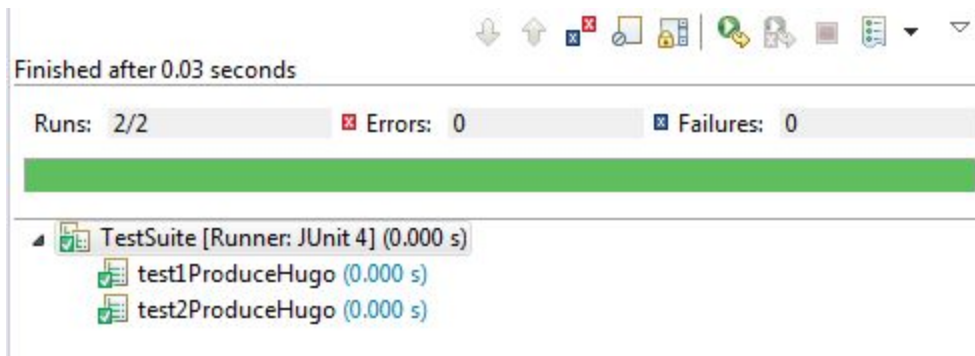
Task 3.1



The screenshot shows a Java IDE with a console window. The console title bar includes tabs for 'Console', 'Main.java', and 'Hugo.java'. The console text shows the command prompt '<terminated> Hugo [Java Application] C:\Program Files\Java\jdk1.8.0_45\bin\javaw.e' followed by five lines of the word 'Hugo'.

```
<terminated> Hugo [Java Application] C:\Program Files\Java\jdk1.8.0_45\bin\javaw.e
Hugo
Hugo
Hugo
Hugo
Hugo
|
```

Task 3.2



The screenshot shows a Java IDE test runner window. At the top, it says 'Finished after 0.03 seconds'. Below this, a summary bar shows 'Runs: 2/2', 'Errors: 0', and 'Failures: 0'. A green progress bar is visible. The test results are listed below, showing a 'TestSuite [Runner: JUnit 4] (0.000 s)' with two sub-tests: 'test1ProduceHugo (0.000 s)' and 'test2ProduceHugo (0.000 s)', both marked with green checkmarks.

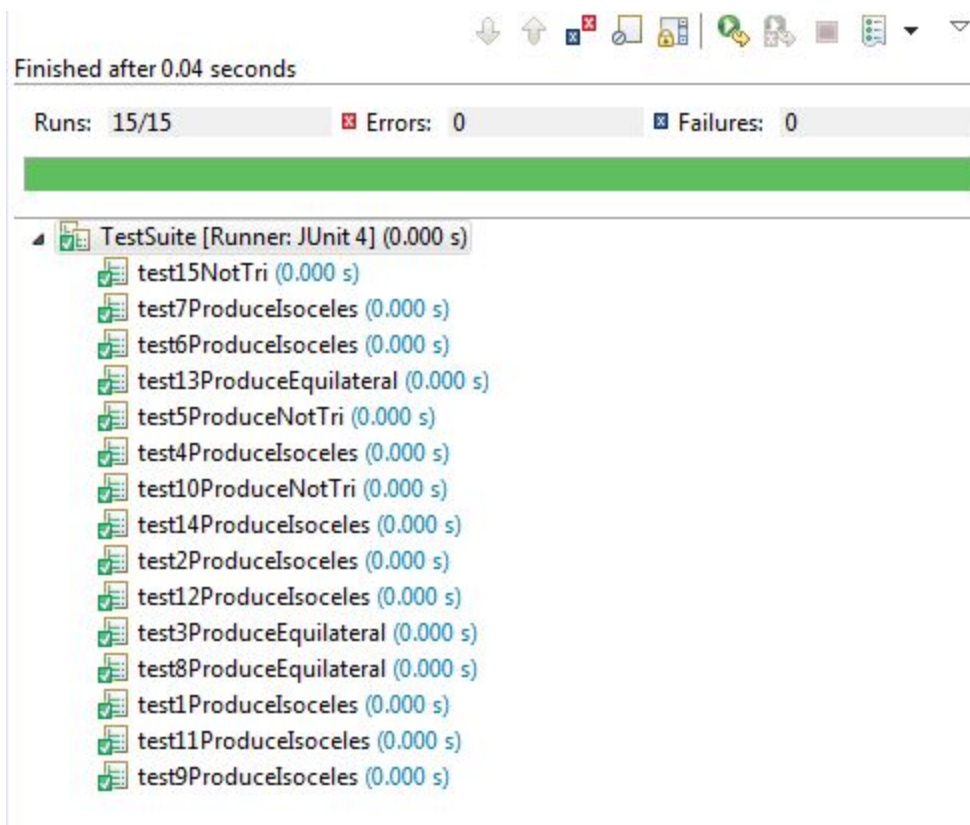
Finished after 0.03 seconds

Runs: 2/2 Errors: 0 Failures: 0

TestSuite [Runner: JUnit 4] (0.000 s)

- test1ProduceHugo (0.000 s)
- test2ProduceHugo (0.000 s)

Task 3.3



The image shows the JUnit test runner interface. At the top, there is a toolbar with various icons for navigation and actions. Below the toolbar, the status bar indicates "Finished after 0.04 seconds". The main area displays the test results, showing "Runs: 15/15", "Errors: 0", and "Failures: 0". A green progress bar is visible below the status bar. The test suite is expanded, showing a list of 15 tests, all of which passed successfully, each with a duration of 0.000 s.

Finished after 0.04 seconds

Runs: 15/15 Errors: 0 Failures: 0

TestSuite [Runner: JUnit 4] (0.000 s)

- test15NotTri (0.000 s)
- test7ProduceIsoceles (0.000 s)
- test6ProduceIsoceles (0.000 s)
- test13ProduceEquilateral (0.000 s)
- test5ProduceNotTri (0.000 s)
- test4ProduceIsoceles (0.000 s)
- test10ProduceNotTri (0.000 s)
- test14ProduceIsoceles (0.000 s)
- test2ProduceIsoceles (0.000 s)
- test12ProduceIsoceles (0.000 s)
- test3ProduceEquilateral (0.000 s)
- test8ProduceEquilateral (0.000 s)
- test1ProduceIsoceles (0.000 s)
- test11ProduceIsoceles (0.000 s)
- test9ProduceIsoceles (0.000 s)

```

1 import static org.junit.Assert.*;
2 import org.junit.Test;
3
4
5 public class TestSuite {
6
7     @Test
8     public void test1ProduceIsoceles() {
9         assertEquals(TriangleClassifier.TriangleType.ISOSCELES, TriangleClassifier.classify(100, 100 , 1));
10    }
11
12    @Test
13    public void test2ProduceIsoceles() {
14        assertEquals(TriangleClassifier.TriangleType.ISOSCELES, TriangleClassifier.classify(100, 100 , 2));
15    }
16
17    @Test
18    public void test3ProduceEquilateral() {
19        assertEquals(TriangleClassifier.TriangleType.EQUILATERAL, TriangleClassifier.classify(100, 100, 100));
20    }
21
22    @Test
23    public void test4ProduceIsoceles() {
24        assertEquals(TriangleClassifier.TriangleType.ISOSCELES, TriangleClassifier.classify(100, 100, 199));
25    }
26
27    @Test
28    public void test5ProduceNotTri() {
29        assertEquals(TriangleClassifier.TriangleType.NOT_A_TRIANGLE, TriangleClassifier.classify(100, 100, 200));
30    }
31
32    @Test
33    public void test6ProduceIsoceles() {
34        assertEquals(TriangleClassifier.TriangleType.ISOSCELES, TriangleClassifier.classify(100, 1, 100));
35    }
36
37    @Test
38    public void test7ProduceIsoceles() {
39        assertEquals(TriangleClassifier.TriangleType.ISOSCELES, TriangleClassifier.classify(100, 2, 100));
40    }
41
42    @Test
43    public void test8ProduceEquilateral() {
44        assertEquals(TriangleClassifier.TriangleType.EQUILATERAL, TriangleClassifier.classify(100, 100, 100));
45    }
46
47    @Test
48    public void test9ProduceIsoceles() {
49        assertEquals(TriangleClassifier.TriangleType.ISOSCELES, TriangleClassifier.classify(100, 199, 100));
50    }
51
52    @Test
53    public void test10ProduceNotTri() {
54        assertEquals(TriangleClassifier.TriangleType.NOT_A_TRIANGLE, TriangleClassifier.classify(100, 200, 100));
55    }
56

```

```

56
57
58     @Test
59     public void test11ProduceIsoceles() {
60         assertEquals(TriangleClassifier.TriangleType.ISOSCELES, TriangleClassifier.classify(1, 100, 100));
61     }
62
63     @Test
64     public void test12ProduceIsoceles() {
65         assertEquals(TriangleClassifier.TriangleType.ISOSCELES, TriangleClassifier.classify(2, 100, 100));
66     }
67
68     @Test
69     public void test13ProduceEquilateral() {
70         assertEquals(TriangleClassifier.TriangleType.EQUILITERAL, TriangleClassifier.classify(100, 100, 100));
71     }
72
73     @Test
74     public void test14ProduceIsoceles() {
75         assertEquals(TriangleClassifier.TriangleType.ISOSCELES, TriangleClassifier.classify(199, 100, 100));
76     }
77
78     @Test
79     public void test15NotTri() {
80         assertEquals(TriangleClassifier.TriangleType.NOT_A_TRIANGLE, TriangleClassifier.classify(200, 100, 100));
81     }
82 }
83

```