

**BCS3323 SOFTWARE TESTING & MAINTENANCE**

**SEMESTER II SESSION 2016/2017**

**LAB 3**

**PREPARED FOR:**

**DR. ABDULRAHMAN AHMED AL-SEWARI**

**PREPARED BY:**

|  |  |
| --- | --- |
| **NAME** | **MATRIC NO** |
| **AHMAD SOLEHIN BIN SHARUDDIN** | **CB15003** |
| **EZZRA NORANNIS BT MOHAMMAD KHAIRUDDIN** | **CB14016** |

1. Assuming the range of a,b,c take the following ranges (with minimum increments of 1):
2. **0 < a ≤ 40**

|  |  |  |
| --- | --- | --- |
| **Lower Invalid** | **Valid** | **Upper invalid** |
| 0 | 1~40 | 41 |

Equivalence Partitioning :{-6, 30, 60}

Boundary Value :{ 10, 11,

40, 41}

1. **0 ≤ b < 30**

|  |  |  |
| --- | --- | --- |
| **Lower Invalid** | **Valid** | **Upper invalid** |
| -1 | 0~29 | 30 |

Equivalence Partitioning :{-2, 20, 40}

Boundary Value :{-1, 0

29, 30}

1. **0 ≤ c ≤ 25**

|  |  |  |
| --- | --- | --- |
| **Lower Invalid** | **Valid** | **Upper invalid** |
| -1 | 0~25 | 26 |

Equivalence Partitioning :{-2, 5, 30}

Boundary Value :{-1, 0

25, 26}

Derive the most minimum set of test cases based on Equivalence Partitioning as well as Boundary Value Analysis to test the method ***public static void triangle (int a, int b, int c)***

1. Consider the following conditions

Scalene: a + b > c, where a<b<c

Isosceles: a + a > c, where b=a

Equilateral: a = a = a, where b=a, c=a, and a >0

If necessary, develops additional test cases to cover the aforementioned conditions.

1. Based on the developed test cases, derive the appropriate test oracle. (Hint: use Excel table)

**REQ\_001:** Test for scalene triangle

**REQ\_002:** Test for isosceles triangle

**REQ\_003:** Test for equilateral triangle

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Req. ID** | **TestCase ID** | **Test Condition** | **a** | **b** | **c** | **Expected** | **Actual** | **Test Status** |
| REQ\_001 | REQ\_001\_T001 | Invalid value a | 0 | 10 | 10 | Scalene triangle | Not a triangle | Fail |
| REQ\_001 | REQ\_001\_T002 | Invalid value b | 11 | -5 | 50 | Scalene triangle | Not a triangle | Fail |
| REQ\_001 | REQ\_001\_T003 | Valid and different value of a, b, c | 12 | 35 | 37 | Scalene triangle | Right angled triangle | Pass |
| REQ\_001 | REQ\_001\_T004 | Valid and same value of a and b | 12 | 12 | 10 | Scalene triangle | Isosceles triangle | Fail |
| REQ\_001 | REQ\_001\_T005 | Valid and same value of a, b, c | 12 | 12 | 12 | Scalene triangle | Equilateral  triangle | Fail |
| REQ\_002 | REQ\_002\_T001 | Invalid Value a, b | 7 | -7 | 21 | Isosceles triangle | Impossible triangle | Fail |
| REQ\_002 | REQ\_002\_T002 | Invalid Value a, c | 9 | -6 | 32 | Isosceles triangle | Impossible triangle | Fail |
| REQ\_002 | REQ\_002\_T003 | Invalid Value b | -5 | 20 | 30 | Isosceles triangle | Impossible triangle | Fail |
| REQ\_002 | REQ\_002\_T004 | Valid Same value a, c | 27 | 27 | 60 | Isosceles triangle | Isosceles triangle | Pass |
| REQ\_002 | REQ\_002\_T005 | Valid value a, b,c | 19 | 20 | 90 | Isosceles triangle | Any triangle | Fail |
| REQ\_003 | REQ\_003\_T001 | Valid value for a,b,c | 20 | 20 | 20 | Equilateral  triangle | Equilateral  triangle | Pass |
| REQ\_003 | REQ\_003\_T002 | Invalid value for a,b,c | -9 | 10 | 10 | Equilateral  triangle | Impossible  triangle | Fail |
| REQ\_003 | REQ\_003\_T003 | Valid value for a,b,c | 15 | 30 | 40 | Equilateral  triangle | Right-angled Triangle | fail |
| REQ\_003 | REQ\_003\_T004 | Invalid value for a,b,c | 20 | 25 | 20 | Equilateral  triangle | Isosceles  Triangle | fail |
| REQ\_003 | REQ\_003\_T005 | Valid value for a,b,c | 11 | 37 | 100 | Equilateral  triangle | Any Triangle | fail |

*class DetermineTriangle*

*{*

*public static void triangle (int a, int b, int c)*

*{*

*int min,med, max;*

*if (a>b)*

*{*

*max=a;*

*min = b;*

*}*

*else*

*{*

*max = b;*

*min = a;*

*}*

*if (c>max)*

*max = c;*

*else if (c<max)*

*min = c;*

*med = a+b+c-min-max;*

*if (max>min+med)*

*System.out.println( "Impossible triangle");*

*else if (max==min)*

*System.out.println( "Equilateral triangle");*

*else if (max==med||med==min)*

*System.out.println( "Isoceles triangle");*

*else if (max\*max==min\*min + med\*med)*

*System.out.println( "Rightangled triangle");*

*else*

*System.out.println(“Any triangle”);;*

*}*

*}*

**Figure 2. Code Unit to DetermineTriangle.java**