IT-314: Software Engineering

Lab Assignment: 08

Title: Functional Testing (Black-Box)

Lab Group: G2

Name: Parth Vadodaria

ID: 202201174

Q.1. Consider a program for determining the previous date. Its input is triple of day, month and year with the following ranges 1 <= month <= 12, 1 <= day <= 31, 1900 <= year <= 2015.The possible output dates would be previous date or invalid date. Design the equivalence class test cases?

Equivalence Classes

E1. Month ranges from 1 to 12 (valid).
E2. Month below 1 (invalid).
E3. Month above 12 (invalid).
E4. Day ranges from 1 to 30 and Month value 1, 3 to 12 (valid).
E5. Day ranges from 1 to 28 and Month value 2 (valid).
E6. Day below 1 (invalid).
E7. Day above 31 (invalid).
E8. Day value 31 and Month value 1,3,5,7,8,10,12 (valid).
E9. Day value 31 and Month value 2,4,6,9,11 (invalid).
E10. Day value 30 and Month value 2 (invalid).
E11. Year ranges from 1900 to 2015 (valid).
E12. Year below 1900 (invalid).
E13. Year above 2015 (invalid).
E14. Day value 29 and Month value 2 and Year is a leap year (valid).
E15. Day value 29 and Month value 2 and Year is not a leap year (invalid).
E16. Day, month and year value numeric (valid).
E17. Any of the input day, month or year value not numeric (invalid).

Sr.no.	Test case (Format: Month, day, year)	Expected Outcome	Covered Equivalence Classes
1	1,24,2006	Previous Date	E1, E4, E11, E16
2	2,28,2015	Previous Date	E1, E5, E11, E16
3	3,31,1900	Previous Date	E1, E8, E11, E16
4	2,29,2000	Previous Date	E1, E14
5	0,24,2006	Invalid Date	E2
6	13,22,2006	Invalid Date	E3
7	12,0,2006	Invalid Date	E6
8	12,32,2006	Invalid Date	E7
9	4,31,2006	Invalid Date	E9
10	2,30,2006	Invalid Date	E10
11	12,22,1899	Invalid Date	E12
12	12,22,2016	Invalid Date	E13
13	2,29,2001	Invalid Date	E15
14	3,a76,2001	Invalid Date	E17

Q2. Write a set of test cases (i.e., test suite) – specific set of data – to properly test the programs. Your test suite should include both correct and incorrect inputs.

1. Enlist which set of test cases have been identified using Equivalence Partitioning and Boundary Value Analysis separately.

2. Modify your programs such that it runs, and then execute your test suites on the program. While executing your input data in a program, check whether the identified expected outcome (mentioned by you) is correct or not.

P1. The function linearSearch searches for a value v in an array of integers a. If v appears in the array a, then the function returns the first index i, such that a[i] == v; otherwise, -1 is returned.

Equivalence Classes

E1. v is a numeric (valid).
E2. v is not a numeric (error).
E3. Array a contains only numeric (valid).
E4. Array a contains values other than numeric (error).
E5. v is present in array a one time.
E6. v is not present in array a.
E7. V is present in array more than one time.

Sr.no.	Test case (Format: v, [a1, a2])	Expected Outcome	Actual Output	Covered Equivalence Classes
1	1, [-2,4,5,1000,-193]	-1	-1	E1,E3,E6
2	1, [-2,1,5,1000,-193]	1	1	E1,E3,E5
3	1, [-2,1,5,1,-193]	1	1	E1,E3,E7
4	Ab1, [-2,4,5,1000]	error		E2
5	1,[-2,abc,4,5]	error		E4

Boundary Analysis

Sr.no.	Test case (Format: v, [a1, a2])	Expected Outcome	Actual Output
1	1, []	-1	-1
2	a, [-2,1,5,1000,-193]	Error	-1
3	A, [-2,-4,1,65,0]	Error	3
4	2.0, [-2,1,2,3]	2	2

P2. The function countItem returns the number of times a value v appears in an array of integers a.

Equivalence Classes

E1. v is a numeric (valid).
E2. v is not a numeric (error).
E3. Array a contains only numeric (valid).
E4. Array a contains values other than numeric (error).

Sr.no.	Test case (Format: v, [a1, a2])	Expected Outcome	Actual Output	Covered Equivalence Classes
1	1, [-2,4,5,1000,-193]	0	0	E1,E3
2	1, [-2,1,5,1000,-193]	1	1	E1,E3
3	1, [-2,1,5,1,-193]	2	2	E1,E3

4	Ab1, [-2,4,5,1000]	error		E2
5	1,[-2,abc,4,5]	error		E4

Boundary Analysis

Sr.no.	Test case (Format: v, [a1, a2])	Expected Outcome	Actual Output
1	1, []	0	0
2	a, [-2,1,5,1000,-193]	Error	0
3	A, [-2,-4,1,65,0]	Error	1
4	2.0, [-2,1,2,3]	1	1

P3. The function binarySearch searches for a value v in an ordered array of integers a. If v appears in the array a, then the function returns an index i, such that a[i] == v; otherwise, -1 is returned.

Equivalence Classes

E1. v is not present in array a.
E2. v is present in array a.
E3. v is present more than one time.

Sr.no.	Test case (Format: v, [a1, a2])	Expected Outcome	Actual Output	Covered Equivalence Classes
1	1, [-2,4,5,1000]	-1	-1	E1
2	1, [-2,1,5,1000]	1	1	E2
3	1, [-2,1,1,5,193]	??		E3

Boundary Analysis

Sr.no.	Test case (Format: v, [a1, a2])	Expected Outcome	Actual Output
1	1, []	-1	-1
2	-2, [-2,1,5,1000]	0	0
3	66, [-4,-2,1,65]	-1	-1
4	-3, [-2,1,2,3]	-1	-1
5	1000, [-2,1,5,1000]	3	3

P4. The following problem has been adapted from The Art of Software Testing, by G. Myers (1979). The function triangle takes three integer parameters that are interpreted as the lengths of the sides of a triangle. It returns whether the triangle is equilateral (three lengths equal), isosceles (two lengths equal), scalene (no lengths equal), or invalid (impossible lengths).

Equivalence Classes

E1. a is less than or equal to b+c and b is less than or equal to a+c and c is less than or equal to a+b.

E2. a is less than or equal to b+c and b is less than or equal to a+c and c is greater than a+b.
E3. a is less than or equal to b+c and b is greater than a+c and c is less than or equal to a+b.
E4. a is greater than b+c and b is less than or equal to a+c and c is less than or equal to a+b.
E5. a is equal to b and not equal to c.
E6. b is equal to c and not equal to a.
E7. a is equal to c and not equal to b.
E8. a is equal to b and c.
E9. Either of a or b or c or all are 0.

Sr.no.	Test case (Format: a, b, c)	Expected Outcome	Actual Output	Covered Equivalence Classes
1	3,4,5	2	2	E1
2	1,7,9	3	3	E2
3	1,9,7	3	3	E3
4	9,1,7	3	3	E4
5	1,1,3	1	1	E5
6	3,1,1	1	1	E6
7	1,3,1	1	1	E7
8	3,3,3	0	0	E8
9	0,1,1	3	3	E9
10	1,0,1	3	3	E9
11	1,0,1	3	3	E9

Boundary Analysis

Sr.no.	Test case (Format: a, b, c)	Expected Outcome	Actual Output
1	1,3,4	2	2
2	4,1,3	2	2
3	3,4,1	2	2
4	0,0,0	3	3

P5. The function prefix (String s1, String s2) returns whether or not the string s1 is a prefix of string s2 (you may assume that neither s1 nor s2 is null).

Equivalence Classes

E1. Length of s1 is greater than s2.
E2. Length of s1 is less than or equal to s2 and s1 is prefix of s2.
E3. Length of s1 is less than or equal to s2 and S1 is not a prefix of s2.

Sr.no.	Test case (Format: s1, s2)	Expected Outcome	Actual Output	Covered Equivalence Classes
1	ab1, ab12fas	true	true	E2
2	ab1, abs12fas	false	false	E3
3	ab123, ab1	false	false	E1

Boundary Analysis

Sr.no.	Test case (Format: s1, s2)	Expected Outcome	Actual Output
1	a, a	true	true
2	A, a	false	false
3	a, bac	false	false

P6. Consider again the triangle classification program (P4) with a slightly different specification: The program reads floating values from the standard input. The three values A, B, and C are interpreted as representing the lengths of the sides of a triangle. The program then prints a message to the standard output that states whether the triangle, if it can be formed, is scalene, isosceles, equilateral, or right angled. Determine the following for the above program:

Equivalence Classes

E1. a is less than or equal to b+c and b is less than or equal to a+c and c is less than or equal to a+b.
E2. a is less than or equal to b+c and b is less than or equal to a+c and c is greater than a+b.
E3. a is less than or equal to b+c and b is greater than a+c and c is less than or equal to a+b.
E4. a is greater than b+c and b is less than or equal to a+c and c is less than or equal to a+b.
E5. a is equal to b and not equal to c.
E6. b is equal to c and not equal to a.
E7. a is equal to c and not equal to b.
E8. a is equal to b and c.
E9. Either of a or b or c or all are 0.
E10. $a^2 + b^2 = c^2$
E11. $a^2 + c^2 = b^2$
E12. $b^2 + c^2 = a^2$

Sr.no.	Test case (Format: a, b, c)	Expected Outcome	Actual Output	Covered Equivalence Classes
1	3,4,6	Scalene	Scalene	E1
2	1,7,9	Invalid	Invalid	E2
3	1,9,7	Invalid	Invalid	E3
4	9,1,7	Invalid	Invalid	E4
5	1,1,3	Isosceles	Isosceles	E5
6	3,1,1	Isosceles	Isosceles	E6
7	1,3,1	Isosceles	Isosceles	E7
8	3,3,3	Equilateral	Equilateral	E8
9	0,1,1	Invalid	Invalid	E9
10	1,0,1	Invalid	Invalid	E9
11	1,0,1	Invalid	Invalid	E9
12	3,4,5	Right-angled	Right-angled	E10
13	4,3,5	Right-angled	Right-angled	E11
14	5,4,3	Right-angled	Right-angled	E12

c. For the boundary condition A + B > C case (scalene triangle), identify test cases to verify the boundary.

Sr.no.	Test case (Format: a, b, c)	Expected Outcome	Actual Output
1	3,4,6	Scalene	Scalene
2	9,10,11	Scalene	Scalene
3	21,22,23	Scalene	Scalene

d. For the boundary condition A = C case (isosceles triangle), identify test cases to verify the boundary.

Sr.no.	Test case (Format: a, b, c)	Expected Outcome	Actual Output
1	1,4,1	Isosceles	Isosceles
2	4,10,4	Isosceles	Isosceles
3	23,24,23	Isosceles	Isosceles

e. For the boundary condition A = B = C case (equilateral triangle), identify test cases to verify the boundary.

Sr.no.	Test case (Format: a, b, c)	Expected Outcome	Actual Output
1	4,4,4	Equilateral	Equilateral
2	10,10,10	Equilateral	Equilateral
3	23,23,23	Equilateral	Equilateral

f. For the boundary condition A2 + B2 = C2 case (right-angle triangle), identify test cases to verify the boundary.

Sr.no.	Test case (Format: a, b, c)	Expected Outcome	Actual Output
1	3,4,5	Right-angled	Right-angled
2	5,12,13	Right-angled	Right-angled
3	7,24,25	Right-angled	Right-angled

g. For the non-triangle case, identify test cases to explore the boundary.

Sr.no.	Test case (Format: a, b, c)	Expected Outcome	Actual Output
1	10,4,5	Non-triangle	Non-triangle
2	5,12,20	Non-triangle	Non-triangle
3	7,50,25	Non-triangle	Non-triangle

h. For non-positive input, identify test points.

Sr.no.	Test case (Format: a, b, c)	Expected Outcome	Actual Output
1	0,4,5	Non-triangle	Non-triangle
2	5,0,0	Non-triangle	Non-triangle
3	0,0,0	Non-triangle	Non-triangle