IT314 Lab8

202201441: Het Panchotiya

Question 1

In Equivalence Partitioning, you divide the input into valid and invalid partitions:

- **Month**:
  - Valid partition: 1≤ month≤12
  - Invalid partition: month<1, month>12
- **Day**:
  - Valid partition: 1≤day≤31 (taking care of different month lengths)
  - Invalid partition: day<1, day>31 , days that don't exist for somemonths
- **Year**:
  - Valid partition: 1900≤year≤2015
  - Invalid partition: year<1900, year>2015

In Boundary Value Analysis, you focus on values at the edges of your partitions:

- **Month boundaries**:
  - Test with months like 1 and 12 (valid) and just outside, like 0 and 13 (invalid).
- **Day boundaries**:
  - Test with days at the beginning (1) and end (31) and just beyond these values, like 0 and 32.
- **Year boundaries**:
  - Test with years 1900 and 2015 (valid), and years just below and above this range.

Test Case	Input (Day, Month, Year)	Test Type	Expected Outcome	Reason

Case 1	(15, 6, 2005)	Equivalence Partitioning	(14, 6, 2005)	Valid mid-range values for day, month, and year.
Case 2	(1, 3, 2000)	Equivalence Partitioning	(29, 2, 2000)	Valid leap year transition (March 1 to Feb 29).
Case 3	(1, 1, 1900)	Equivalence Partitioning	(31, 12, 1899)	Valid start boundary for year.
Case 4	(31, 12, 2015)	Equivalence Partitioning	(30, 12, 2015)	Valid end boundary for year.
Case 5	(32, 1, 2005)	Equivalence Partitioning	Error (Invalid day)	Day exceeds valid range for January.
Case 6	(29, 2, 1900)	Equivalence Partitioning	Error (Invalid date)	February 29 does not exist in the year 1900 (non-leap year).
Case 7	(15, 13, 2005)	Equivalence Partitioning	Error (Invalid month)	Month exceeds valid range (1-12).
Case 8	(15, 6, 2016)	Equivalence Partitioning	Error (Year out of range)	Year exceeds upper boundary (2015).
Case 9	(1, 1, 2000)	Boundary Value Analysis	(31, 12, 1999)	Transition from Jan 1 to Dec 31 of the previous year.
Case 10	(1, 12, 2000)	Boundary Value Analysis	(30, 11, 2000)	Transition from Dec 1 to Nov 30.
Case 11	(31, 12, 2015)	Boundary Value Analysis	(30, 12, 2015)	Valid boundary for end of year 2015.
Case 12	(1, 13, 2005)	Boundary Value Analysis	Error (Invalid month)	Invalid boundary for month (13 out of range).

Case 13	(31, 1, 2000)	Boundary Value Analysis	(30, 1, 2000)	Transition within January.
Case 14	(1, 2, 2000)	Boundary Value Analysis	(31, 1, 2000)	Transition from Feb 1 to Jan 31.
Case 15	(29, 2, 2000)	Boundary Value Analysis	(28, 2, 2000)	Leap year boundary in February.
Case 16	(0, 1, 2000)	Boundary Value Analysis	Error (Invalid day)	Day below valid range.
Case 17	(32, 1, 2000)	Boundary Value Analysis	Error (Invalid day)	Day exceeds valid range.
Case 18	(1, 1, 1900)	Boundary Value Analysis	(31, 12, 1899)	Boundary at start of valid year range.
Case 19	(31, 12, 2015)	Boundary Value Analysis	(30, 12, 2015)	Boundary at end of valid year range.
Case 20	(15, 6, 1899)	Boundary Value Analysis	Error (Year below range)	Year is below valid boundary.
Case 21	(15, 6, 2016)	Boundary Value Analysis	Error (Year out of range)	Year exceeds upper boundary.

Program 1

Equivalence Partitioning :

- The array includes the value v.
- The array does not include the value v.
- The array is empty.
- The array contains the value v more than once.

Boundary Value Analysis:

- Test with an array of length 1.
- Test with an array of larger size.
- Test when v matches the first or last element.

Test Case	Input (v, a[])	Test Type	Expected Outcome	Reason
Case 1	(5, [1, 2, 3, 4, 5])	Equivalence Partitioning	4	Array contains v at index 4.
Case 2	(3, [1, 2, 3, 4, 5])	Equivalence Partitioning	2	Array contains v at index 2.
Case 3	(6, [1, 2, 3, 4, 5])	Equivalence Partitioning	-1	Array does not contain v.
Case 4	(3, [])	Equivalence Partitioning	-1	Array is empty, so v cannot be found.
Case 5	(3, [3, 1, 3, 4, 5])	Equivalence Partitioning	0	Array contains v multiple times, first occurrence at index 0.
Case 6	(5, [5])	Boundary Value Analysis	0	Single element array where v is the first and only element.

Case 7	(10, [5, 6, 7, 8, 9, 10])	Boundary Value Analysis	5	v is at the last index of the array.
Case 8	(1, [1, 2, 3, 4, 5])	Boundary Value Analysis	0	v is at the first index of the array.
Case 10	(7, [-5, -3, 0, 7, 10])	Boundary Value Analysis	3	Array contains negative, zero, and positive values; v is present.
Case 11	(-3, [-5, -3, 0, 7, 10])	Boundary Value Analysis	1	v is a negative number in the array.
Case 12	(100, [5, 10, 20, 30, 40])	Boundary Value Analysis	-1	v is much larger than any element in the array.
Case 13	(-100, [5, 10, 20, 30, 40])	Boundary Value Analysis	-1	v is much smaller than any element in the array.

Program 2

Equivalence Partitioning:

- The array does not include v.
- The array includes v exactly once.
- The array contains v multiple times.
- The array is empty.

Boundary Value Analysis:

- Array with only one element.
- Array with a larger size.
- v appears at the first and last element of the array.

Test Case	Input (v, a[])	Test Type	Expected Outcome	Reason
Case 1	(5, [1, 2, 3, 4, 5])	Equivalence Partitioning	1	Array contains v exactly once at the last index.
Case 2	(3, [1, 2, 3, 4, 5])	Equivalence Partitioning	1	Array contains v exactly once at the middle.
Case 3	(6, [1, 2, 3, 4, 5])	Equivalence Partitioning	0	Array does not contain v.
Case 4	(3, [])	Equivalence Partitioning	0	Array is empty, so v cannot be found.
Case 5	(3, [3, 1, 3, 4, 5])	Equivalence Partitioning	2	Array contains v multiple times (two occurrences).
Case 6	(5, [5])	Boundary Value Analysis	1	Single element array where v is the first and only element.
Case 7	(3, [3, 3, 3, 3, 3])	Boundary Value Analysis	5	Array contains v at all positions (5 occurrences).
Case 8	(10, [5, 6, 7, 8, 9, 10])	Boundary Value Analysis	1	v is at the last index of the array.
Case 9	(1, [1, 2, 3, 1, 1])	Boundary Value Analysis	3	v appears multiple times, including the first index.

Case 10	(-3, [-5, -3, 0, 7, 10])	Boundary Value Analysis	1	v is a negative number present in the array.
Case 12	(100, [5, 10, 20, 30, 40])	Boundary Value Analysis	0	v is much larger than any element in the array, not found.
Case 13	(-100, [5, 10, 20, 30, 40])	Boundary Value Analysis	0	v is much smaller than any element in the array, not found.
Case 14	(0, [0, 1, 0, 2, 0])	Boundary Value Analysis	3	v appears multiple times as zero (3 occurrences).

Program 3

Equivalence Partitioning:

- The array contains v.
- The array does not contain v.
- The array is empty.
- The array contains v multiple times.

Boundary Value Analysis (BVA):

- Test with an array of length 1.
- Test when v matches the first or last element.
- Test when v matches the middle element.

Test Case	Input (v, a[])	Test Type	Expected Outcome	Reason
Case 1	(5, [1, 2, 3, 4, 5])	Equivalence Partitioning	4	Array contains v at index 4 (last element).

Case 2	(3, [1, 2, 3, 4, 5])	Equivalence Partitioning	2	Array contains v at index 2 (middle element).
Case 3	(6, [1, 2, 3, 4, 5])	Equivalence Partitioning	-1	Array does not contain v.
Case 4	(3, [])	Equivalence Partitioning	-1	Array is empty, so v cannot be found.
Case 5	(3, [3, 3, 3, 3, 3])	Equivalence Partitioning	Any valid index (0-4)	Array contains multiple occurrences of v (all elements are v).
Case 6	(5, [5])	Boundary Value Analysis	0	Single element array where v matches the first and only element.
Case 7	(1, [1, 2, 3, 4, 5])	Boundary Value Analysis	0	v is the first element in the array.
Case 8	(10, [5, 6, 7, 8, 9, 10])	Boundary Value Analysis	5	v is the last element in the array.
Case 9	(7, [5, 6, 7, 8, 9, 10])	Boundary Value Analysis	2	v is the middle element of the array.
Case 10	(-3, [-5, -3, 0, 7, 10])	Boundary Value Analysis	1	v is a negative number present at index 1 in the sorted array.
Case 11	(100, [10, 20, 30, 40, 50])	Boundary Value Analysis	-1	v is much larger than any element in the array, not found.
Case 12	(-100, [-50, -30, -10, 0, 20])	Boundary Value Analysis	-1	v is much smaller than any element in the array, not found.

Program 4

Equivalence Partitioning:

1. All three sides are equal.
2. Two sides are equal, one is different.
3. All three sides are different.
4. The sides do not satisfy the triangle inequality.

Boundary Value Analysis:

1. Minimum side lengths that can form a triangle (e.g., all sides of length 1).
2. Large integers to represent maximum side lengths.
3. One or more sides are zero.
4. One or more sides are negative.
5. Cases where the sum of two sides equals the third.

Test Case	Input (v, a[])	Test Type	Expected Outcome	Reason
Case 1	(3, 3, 3)	Equivalence Partitioning	0	Equilateral
Case 2	(3, 3, 4)	Equivalence Partitioning	1	Isosceles
Case 3	(3, 4, 5)	Equivalence Partitioning	2	Scalene
Case 4	(1, 2, 3)	Equivalence Partitioning	3	Invalid (inequality fails).
Case 5	(0, 0., 0)	Equivalence Partitioning	3	Side cannot be 0

Case 6	(1, 1, 1)	Boundary Value Analysis	0	Equilateral
Case 7	(1, 1, 2)	Boundary Value Analysis	3	Invalid (inequality fails).
Case 8	(3, -1, 4)	Boundary Value Analysis	3	Negative side
Case 9	(3, 5, -2)	Boundary Value Analysis	3	Negative edge
Case 10	(2, 2, 5)	Boundary Value Analysis	3	Invalid (inequality fails).
Case 11	(1, 2, 1)	Boundary Value Analysis	3	Invalid (inequality fails).
Case 12	(1000000, 10000000, 10000000)	Boundary Value Analysis	0	Equilateral

Program 5

Equivalence Partitioning:

1. All characters in s1 match the start of s2.
2. The characters in s1 do not match the start of s2.
3. s1 cannot be a prefix of s2.
4. An empty string is a prefix of any string.
5. A non-empty s1 cannot be a prefix of an empty s2.

Boundary Value Analysis (BVA):

1. Edge case where both strings are empty.
2. Valid prefix case.
3. Invalid prefix case.
4. Both strings are identical.
5. Testing with lengths differing by one character.

Test Case	Input (s1, s2)	Test Type	Expected Outcome	Reason
Case 1	("pre", "prefix")	Equivalence Partitioning	true	s1 is a prefix of s2.
Case 2	("pre", "postfix")	Equivalence Partitioning	false	s1 does not match the start of s2.
Case 3	("prefix", "pre")	Equivalence Partitioning	false	s1 is longer than s2.
Case 4	("", "hello")	Equivalence Partitioning	true	Empty s1 is a prefix of any non-empty s2.
Case 5	("hello", "")	Equivalence Partitioning	false	Non-empty s1 cannot be a prefix of an empty s2.
Case 6	("", "")	Boundary Value Analysis	true	Both strings are empty.
Case 7	("test", "test")	Boundary Value Analysis	true	s1 equals s2 (identical).
Case 8	("hello", "he")	Boundary Value Analysis	false	s1 is longer than the start of s2.

Case 9	("pre", "prefixe")	Boundary Value Analysis	true	s1 is a prefix of s2.
Case 10	("abc", "abcd")	Boundary Value Analysis	true	s1 is a prefix of s2 (lengths differ by 1).
Case 11	("abc", "ab")	Boundary Value Analysis	false	s1 is not a prefix of s2.
Case 12	("abcdefg", "abc")	Boundary Value Analysis	false	s1 is longer than s2.
Case 13	("123", "123456")	Equivalence Partitioning	true	s1 is a prefix of s2.
Case 14	("abc", "123abc")	Equivalence Partitioning	false	s1 does not match the start of s2.

Program 6

a) Identify the equivalence classes for the system

TC1-Equilateral Triangle: All sides are equal, i.e., $A=B=C$

TC2-Isosceles Triangle: Two sides are equal, i.e., $A=B$, $A=C$, or $B=C$

TC3-Scalene Triangle: All sides are different, i.e., $A \neq B \neq C =$

TC4-Right-Angled Triangle: Satisfies $A^2+B^2=C^2$, $A^2 + B^2 = C^2$, $A^2+B^2=C^2$

TC5-Invalid Triangle: Triangle inequality does not hold, i.e., $A \geq B+C$ or similar conditions.

TC6-non-positive input: Input values $A \leq 0$, $B \leq 0$, or $C \leq 0$.

b) Identify test cases to cover the identified equivalence classes. Also, explicitly mention which test case would cover which equivalence class.

Test Case	Input (A, B, C)	Expected Result	Covered Equivalence Class
TC1	3, 3, 3	0	Equilateral triangle
TC2	4, 4, 5	1	Isosceles triangle
TC3	3, 4, 5	2	Scalene triangle
TC4	1, 2, 3	3	Invalid triangle
TC5	0, 3, 4	3	Non-positive input
TC6	-1, 2, 3	3	Non-positive input
TC7	5, 12, 13	2	Right-angled scalene triangle

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

Test Case	Input (A, B, C)	Expected Result	Remarks
TC8	10, 11, 2	2	Scalene triangle
TC9	5, 6, 10	2	Scalene Triangle

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

Test Case	Input (A, B, C)	Expected Result	Remarks
TC9	5.1, 5.1, 10	1	Isosceles triangle
TC10	10000, 10000, 1	1	Isosceles triangle

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

Test Case	Input (A, B, C)	Expected Result	Remarks
TC11	1, 1, 1	0	Equilateral triangle
TC12	1000, 1000, 1000	0	Equilateral triangle

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

Test Case	Input (A, B, C)	Expected Result	Remarks
TC13	3, 4, 5	2	Right-angled scalene triangle
TC14	5, 12, 13	2	Right-angled scalene triangle

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

Test Case	Input (A, B, C)	Expected Result	Remarks
TC15	1, 2, 3	3	Invalid triangle
TC16	1, 2, 10	3	Invalid triangle

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

Test Case	Input (A, B, C)	Expected Result	Remarks
TC17	0, 0, 0	3	Invalid triangle
TC18	-3, 4, 5	3	Invalid triangle