Name : Sneh Joshi
ID : 20220148
LAB 08

Equivalence Class Testing for Previous Date Program

- ### Equivalence Partitioning

Identified Equivalence Classes:

- Valid Dates:
    a. January 1, 1900 (1, 1, 1900)
    b. February 28, 1900 (28, 2, 1900) - Non-leap year
    c. March 1, 1900 (1, 3, 1900)
    d. April 30, 2015 (30, 4, 2015)

- Invalid Dates:
    - Month out of range (e.g., Month = 0 or Month = 13)
    - Day out of range (e.g., Day = 32 or Day = -1)
    - Invalid combinations (e.g., February 29 on a non-leap year)

Test Cases for Equivalence Partitioning:

Tester Action and Input Data	**Expected Outcome**
(1, 1, 1900)	Valid date: December 31, 1899
(28, 2, 1900)	Valid date: February 27, 1900
(1, 3, 1900)	Valid date: February 28, 1900

(30, 4, 2015)	Valid date: April 29, 2015
(0, 1, 2000)	Error message
(32, 1, 2000)	Error message
(29, 2, 1900)	Error message

Boundary Value Analysis

Identified Boundary Values:

- Valid Dates at Boundaries:
  - January (1st day)
  - February (28th day in non-leap year)
  - March (1st day)
- Invalid Dates at Boundaries:
  - Month = -1
  - Month = +13
  - Day = +32
  - Day = +0

Test Cases for Boundary Value Analysis:

Tester Action and Input Data	Expected Outcome
(1, 1, 2000)	Valid date: December 31, 1999
(28, 2, 2000)	Valid date: February 27, 2000
(29, 2, 2000)	Valid date: February 28, 2000
(31,12,2015)	Valid date: December 30,2015
(-1,-1,-1)	Error message
(13,-1,-1)	Error message

Programs for Searching and Triangle Classification

P1: Linear Search

```
c
int linearSearch(int v, int a[], int length)
    {for(int i = 0; i < length; i++) {
        if(a[i] == v) return i;
    }
    return -1;
}
```

P2: Count Item

```
c
int countItem(int v, int a[], int length)
    {int count = 0;

    for(int i = 0; i < length; i++)
        {if(a[i] == v) count++;
    }
    return count;
}
```

P3: Binary Search

```
c
int binarySearch(int v, int a[], int length)
    {int lo = 0;
    int hi = length -
    1;while(lo <= hi)
    {
        int mid = lo + (hi -
        lo)/2;if(a[mid] == v)
        return mid;
        else if(v < a[mid]) hi = mid -
        1;else lo = mid + 1;
    }
    return -1;
}
```

P4: Triangle Classification

C

```
int triangle(int a,int b,int c){
    if(a >= b+c || b >= a+c || c >=
        a+b)return INVALID;
    if(a == b && b == c)
        return
        EQUILATERAL;
    if(a == b || a == c || b ==
        c)return ISOSCELES;
    return SCALENE;
}
```

P6: Triangle with Floating Values

- Equivalence Classes for Triangle Classification with Floating Values
  - Valid Classes:
    - Equilateral triangles with sides equal.
    - Isosceles triangles with two sides equal.
    - Scalene triangles with all sides different.
  - Invalid Classes:
    - Non-triangles where the sum of any two sides is less than or equal to thethird side.
    - Non-positive lengths.

- Test Cases Covering Identified Equivalence Classes

Tester Action and Input Data	Expected Outcome

(3.0,3.0,3.0)	Equilateral
(4.0,4.0,6.0)	Isosceles
(3.0,4.0,5.0)	Scalene
(-3.0,-4.0,-5.0)	Invalid
(10.0,-5.0,-7.5)	Invalid

- Boundary Condition $A + B > C$

CaseTest cases:

- A=3,B=4,C=6 -> Scalene
- A=3,B=4,C=7 -> Invalid

- Boundary Condition $A = C$

CaseTest cases:

- A=5,B=5,C=6 -> Isosceles
- A=5,B=6,C=6 -> Isosceles

- Boundary Condition $A = B = C$

CaseTest cases:

- A=3,B=3,C=3 -> Equilateral
- A=4,B=4,C=4 -> Equilateral

- Boundary Condition $A^2 + B^2 = C^2$ Case

Test cases:

- A=3,B=4,C=5 -> Right angled triangle
- A=5,B=12,C=13 -> Right angled triangle

- Non-Triangle

CaseTest cases:

- A=2, B=2, C=5 -> Invalid
- A=3,B=2,C=6 -> Invalid

- Non-positive

InputTest cases:

- A=-1,B=-2,C=-3 -> Invalid
- A=0,B=2,C=-2 -> Invalid