IT-314
Software Engineering

Lab 9 : Lab Session - SEN
Specification-based Test Case Generation

Project - Online Faculty Staff Directory for Multi University

~ By Group 17

- **AYAN KHOKHAR - 201801057 (Group Leader)**
- **RUTVIK SHAH - 201801113**
- **URMIK MAKANI - 201801117**
- **TULSI SHAH - 201801180**
- **ANAND NADPARA - 201801192**
- **JHANVI SHROFF - 201801198**
- **AKASH PARAMAR - 201801212**
- **SHYAM VAGHELA - 201801214**
- **NISHTHA CHAUDHARY - 201801235**
- **TUSHAR PATIL - 201801440**

Functional testing of the Requirement Specification:

Q1). 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?
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.

Answer

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 on eclipse IDE, 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.

1. Equivalence Classes
   Day :-
   a. Days from 1 to 28 are valid.
   b. Days<1 are invalid
   c. Days 29
   d. Days 30
   e. Days 31
   f. Days>31 are invalid.

   Month :-
   a. Months with 31 days [Jan(1), Mar(3), May(5), Jul(7), Aug(8), Oct(10), Dec(12)] are valid
   b. Months with 30 days [April (04)[, June(06), September(09), November(11)]
   c. Months with 28/29 days [February (02)]
   d. Months< 1 are invalid
   e. Months>12 are invalid

   Year :-
   a) Leap years from 1900 to 2015
   b) Non-leap years from 1900 to 2015
   c) Years > 2015
   d) Years < 1900

Test Case	**Day**	**Month**	**Year**	**Output**
1	b	any	any	Invalid

2	f	any	any	Invalid
3	a	a	a	Valid
4	a	a	b	Valid
5	a	b	a	Valid
6	a	b	b	Valid
7	c	c	a	Valid
8	any	d	any	Invalid
9	any	e	any	Invalid
10	any	any	c	Invalid
11	any	any	d	Invalid
12	e	b	any	Invalid
13	d	a	any	Invalid
14	d/e	c	any	Invalid
15	b	c	any	Invalid

2. We have used C14++ language.

```
#include<bits/stdc++.h>
using namespace std;
int main(){

  int day,month,year;

  cout<<"Enter the date in format of (Day Month Year) :"<<endl;
  cin>>day>>month>>year;

  if(1900<=year<=2015)
      {
        if((month==1 || month==3 || month==5|| month==7||
month==8||month==10||month==12) && day>0 && day<=31)
        cout<<"It is valid";
        else
    if(month==4 || month==6 || month==9|| month==11 && day>0 && day<=30)
          cout<<"It is Valid";
        else
          if(month==2)
            {
            if((year%400==0 || (year%100!=0 && year%4==0)) && day>0 && day<=29)
              cout<<"It is Valid";
```

```
            else if(day>0 && day<=28)
              cout<<"It is Valid";
            else
              cout<<"It is Invalid";
            }
         else
            cout<<"It is Invalid";
       }
      else
        cout<<"It is Invalid";

   return 0;
}
```

Q2) You are testing an e-commerce system that sells products like caps and jackets. The problem is to create functional tests using boundary-value analysis and equivalence class partitioning techniques for the web page that accepts the orders. A screen prototype for the order-entry web page is shown below.

The system accepts a five-digit numeric item ID number from 00000 to 99999. The system accepts a quantity to be ordered, from 1 to 99. If the user enters a previously ordered item ID and a 0 quantity to be ordered, that item is removed from the shopping cart. Based on these inputs, the system retrieves the item price, calculates the item total (quantity times item price), and adds the item total to the cart total. Due to limits on credit card orders that can be processed, the maximum cart total is $999.99.

Answer:
Given Constraints:

ID: 00000-99999
Quantity: 1-99
Cart total: less than or equal to $999.99

Equivalence Classes:

ID:
1) ID between 00000-99999 (Both inclusive) valid
2) ID less than 00000 invalid
3) ID greater than 99999 invalid

Quantity:
4) Quantity between 1-99 (both inclusive) valid
5) Quantity less than 0 invalid
6) Quantity greater than 99 invalid
7)Quantity= 0

Cart total:
7) Cart total amount between 0-999.99(both inclusive) valid
8) Cart total greater than 999.99 invalid
9) Cart total less than 0 invalid

Test Case	ID	Quantity	Cart Total(in dollar)	Output
1	<00000	any	any	invalid
2	>99999	any	any	invalid
3	any	<0	any	invalid
4	any	>99	any	invalid
5	any	any	<0	invalid
6	any	any	>999.99	invalid
7	00000<=ID<=99999	Quantity=0	0	Valid and Item should remove from the cart
8	00000<=ID<=99999	1<=Quantity<=99	0<=Amount<=999.99	valid