Experiment : 2(a)

Date 14-10-21

Page No. 41

- Problem Statement:- To determine the nature of roots of Quad equations. Its input is triple of +ve integers (say a,b,c) and values may be from the interval [1,100]. Generate the test case using B.V.A.

- Source Code:-

```
#include<bits/stdc++.h>
using namespace std;

void quadratic(int a, int b, int c) {
   if (a==0) {
      cout<< "Not a Quad. Equation" <<endl;
      return;
   }
int d= b*b - 4*a*c;
if (a<1 || b<1 || c<1 || a>100 || b>100 || c>100)
  cout << "Out of Range" << endl;
else if (d<0)
  cout << "Imaginary roots" << endl;
else if (d==0)
  cout << " Equal roots" <<endl;
else if
   cout << " Real roots" << endl;
}

int main() {
```

DELTA

```
int t;
cin >>t;
while (t-->0){
int a,b,c;
 cin >>a>>b>> c;
 quadratic(a,b,c);
}
return 0;
}
```

- Total no. of test cases ⇒ 4*n+1
  ⇒ 13

Test Case ID	Test Case Scenario	Test Data	Expected Result	Actual Result	Pass / Fail
TU01	Inputting val. for a, b & c.	a=0, b=50, c=50	Not Quad.	Not Quad	Pass
TU02	"	a=1, b=50, c=50	Real	Real	Pass
TU03	"	a=50, b=50, c=50	Imaginary	Imaginary	Pass
TU04	"	a=99, b=50, c=50	Imaginary	Imaginary	Pass
TU05	"	a=100, b=50, c=50	Imaginary	Imaginary	Pass
TU06	"	a=50, b=0, c=50	Imaginary	Imaginary	Pass
TU07	"	a=50, b=1, c=50	Imaginary	Imaginary	Pass
TU08	"	a=50, b=99, c=50	Imaginary	Imaginary	Pass
TU09	"	a=50, b=100, c=50	Equal	Equal	Pass
TU10	"	a=50, b=50, c=0	Real	Real	Pass
TU11	"	a=50, b=50, c=1	Real	Real	Pass
TU12	"	a=50, b=50, c=99	Imaginary	Imaginary	Pass
TU13	"	a=50, b=50, c=100	Imaginary	Imaginary	Pass

Experiment: 2(b)

Date 14-10-21

Page No. 6

- Problem Statement :- To determine the type of triangle. Its I/P is triple of +ve integers & values maybe from the interval [1, 100]. Generate the test cases using B.V.A.

- Source Code :-

```
#Include<bits/stdc++.h>
using namespace std;
void triangle(int a, int b, int c){
if(a<1 || b<1 || c<1 || a>100 || b>100 || c>100){
    cout << "Out of boundy" <<endl;
    return;
}
if((a+b>c && (b+c)>a && (a+c)>b){
if((a==b) && (b==c))
  cout << "Equilateral" <<endl;
else if((a==b) || (b==c) || (a==c))
  cout << "Isosceles" << endl;
else
  cout << "Scalar" << endl;
}
else
  cout << "It is not a triangle" << endl;
}

int main(){
int t;
cin >> t;
```

DELTA

```
while (t-->0) {
int a,b,c;
cout<< "Enter the sides of the triangle" <<endl;
cin>>a>>b>>c;
triangle(a,b,c);
}
return 0;
}
```

- Total no. of test cases = $4*n+1$
  $\Rightarrow 13$

Test Case ID	Test Case Scenario	Test Data	Expected Result	Actual Result	Pass/Fail
TU01	Inputting val. for a, b & C	a=1 , b=50 , c=50	Isosceles	Isosceles	Pass
TU02	"	a=2, b=50, c=50	Isosceles	Isosceles	Pass
TU03	"	a=99, b=50, c=50	Isosceles	Isosceles	Pass
TU04	"	a=100, b=50, c=50	Not a Triangle	Not a Triangle	Pass
TU05	"	a=50, b=50, c=50	Equilateral	Equilateral	Pass
TU06	"	a=50 , b=1 , c=50	Isosceles	Isosceles	Pass
TU07	"	a=50 , b=2 , c=50	Isosceles	Isosceles	Pass
TU08	"	a=50 , b=99 , c=50	Isosceles	Isosceles	Pass
TU09	"	a=50 , b=100 , c=50	Not a Triangle	Not a Triangle	Pass
TU10	"	a=50 , b=50 , c=1	Isosceles	Isosceles	Pass
TU11	"	a=50 , b=50, c=2	Isosceles	Isosceles	Pass
TU12	"	a=50 , b=50, c=99	Isosceles	Isosceles	Pass
TU13	"	a=500, b=50 , c=100	Not a Triangle	Not a Triangle	Pass

Experiment : 3(a)

Date 21-10-21

Page No. 8

- Problem Statement :-

Consider the example of an App that classified risk exposure (RE) as high, moderate or low on the basis of risk probability (RP) & risk impact (RI). Consider the following specification for such an app:

1) The app accepts two integers RP & RI as input.
2) Both RP & RI must lie in the range of [1,5].
3) If either input is not an integer, its invalid & app o/p invalid.
4) If either i/p is an integer outside the range specified in (2), its invalid & app o/p 'out of range'.
5) Given valid inputs, the app calculate RE as the product of RP & RI as outputs.
   a) 'High' if RE > 9 b) 'Moderate' if RE neither a nor c
   c) 'low' if RE ≤ 2

- Source Code :-

```
import java.util.*;
public class main {
    Scanner s = new Scanner(System.in);
    public static void main(String[] args){
      int t= s.nextInt();
      while (t-->0) {
            int a=0;
            int b=0;
      if( s.hasNextInt()==true) { a= s.nextInt(); }
      else {
            System.out.println("Invalid");
            continue;
            }
```

DELTA®

ii)

Test Case ID	Test Case Scenario	Test Data		Expected Result	Actual Result	Pass / Fail
		RP	RI			
I1	Input values for RP & RI	2	2	Moderate	Moderate	Pass
I2	"	3	0	Out of Range	Out of Range	Pass
I3	"	3	6	Out of Range	Out of Range	Pass
I4	"	0	3	Out of Range	Out of Range	Pass
I5	"	6	3	Out of Range	Out of Range	Pass
I6	"	0	0	Out of Range	Out of Range	Pass
I7	"	0	6	Out of Range	Out of Range	Pass
I8	"	6	0	Out of Range	Out of Range	Pass
I9	"	6	6	Out of Range	Out of Range	Pass

```
if (s.hasNextInt() { b = s.nextInt(); }
else { System.out.println ("Invalid");
       continue ; }
if (a<1 || a>5 || b<1 || b>5) {
       System.out.println ("Out of range");
       continue; }
int c = a*b;
if (c > a)
       System.out.println ("High");
 else if (c<= 2)
       System.out.println ("low");
 else
       System.out.println ("moderate");
}}}
```

i) Partition the domain of each parameter into equivalence classes, label the classes & list them.

⇒ The following equivalence classes can be generated for input

$I_1 = \{ 1 \le RP \le 5 \ \& \ 1 \le RI \le 5 \}$ (Both RP & RI are valid)

$I_2 = \{ 1 \le RP \le 5 \ \& \ RI < 1 \}$ (RP is valid & RI is invalid)

$I_3 = \{ 1 \le RP \le 5 \ \& \ RI > 5 \}$ (RP is valid & RI is invalid)

$I_4 = \{ RP < 1 \ \& \ 1 \le RI \le 5 \}$ (RP is invalid & RI is valid)

$I_5 = \{ RP > 5 \ \& \ 1 \le RI \le 5 \}$ (RP is invalid & RI is valid)

$I_6 = \{ RP < 1 \ \& \ RI < 1 \}$ (Both RP & RI are invalid)

$I_7 = \{ RP < 1 \ \& \ RI > 5 \}$ (Both RP & RI are invalid)

$I_8 = \{ RP > 5 \ \& \ RI < 1 \}$ (Both RP & RI are invalid)

$I_9 = \{ RP > 5 \ \& \ RI > 5 \}$ (Both RP & RI are invalid).

iii)

Test Case ID	Test Case Scenario	Test Input		Expected Result (RE)	Actual Result (RE)	Pass / Fail
		RP	RI			
T01	Input values of RP & RI	3	4	High	High	Pass
T02	"	1	1	low	low	Pass
T03	"	2	3	Moderate	Moderate	Pass
T04	"	'A'	2	Invalid	Invalid	Pass
T05	"	2	'A'	Invalid	Invalid	Pass
T06	"	6	3	Out of Range	Out of Range	Pass
T07	"	3	6	Out of Range	Out of Range	Pass

ii) Develop a set of test cases for the app to satisfy each choice coverage of the equivalence classes. Indicate the equivalence classes covered by each test case & include the expected result.

⇒ Table on page [9].

iii) Partition the output domain and develop additional test cases to cover any class not covered by test cases in (ii).

⇒ O_1 = {High} (RE is greater than 9)

O_2 = {low} (RE is less than or equal to 2)

O_3 = {Moderate} (RE is greater than 2 & less than 9)

O_4 = {Invalid} (Either input is not an integer) (Rule count=2)

O_5 = {Out of Range} (Either input is out of range) (Rule count=2)

Table on the left.

Test Case ID	Test Case Scenario	Test Input	Expected Output	Actual Output	Pass / Fail
T001	Input age, destination, dep. day & stay dur^{n}.	Age=?, dist=?, day=?, stay=?	Impossible	Impossible	Pass
T002	"	"	Impossible	Impossible	Pass
T003	"	"	100%	100%	Pass
T004	"	"	Impossible	Impossible	Pass
T005	"	"	40%	40%	Pass
T006	"	"	0%	0%	Pass
T007	"	"	30%	30%	Pass
T008	"	"	20%	20%	Pass
T009	"	"	0%	0%	Pass
T010	"	"	25%	25%	Pass
T011	"	"	Impossible	Impossible	Pass

Experiment: 3(b)

Date 21-10-21

Page No. 11

- Problem Statement:-

Develop a limited entry decision table for the following decision situation:

An airline offers only flights in Germany & Europe. Under special conditions, a discount is offered, a discount w.r.t. the normal fare. Rules:

1) Passengers older than 18 with destinations in Germany are offered a discount of 20% if the departure is not on a Monday or Friday. If the passengers stay least 6 days at the destination, an additional discount of 10% is offered. For destinations outside of Germany are offered a discount of 25%, if the departure is not on a Monday or Friday.
2) Passengers older than 2 but younger than 18 years are offered a discount of 40% for all destinations.
3) Children under 2 travel for free.

- Source Code :-

```
important java.util.*;
public class Main {
  public static void main (String[] args) {
    Scanner s = new Scanner (System.in);
    int t = s.nextInt();
  while (t-->0) {
    System.out.println ("Enter age");
    int age = s.next Int();
    if (age<0) { System.out.println ("Invalid age");
                 continue; }
```

DELTA

Decision Table :-

	1	2	3	4	5	6	7	8	9	10	11
c1 : age<2	T	T	T	F	F	F	F	F	F	F	F
c2 : 2≤age≤18	T	F	F	T	T	F	F	F	F	F	F
c3 : age>18	–	T	F	T	F	T	T	T	T	T	F
c4 : destination = Germany	–	–	–	–	–	T	T	T	F	F	–
c5 : departure = Mon ! Frid.	–	–	–	–	–	T	F	F	T	F	–
c6 : stay ≥ 6days	–	–	–	–	–	–	T	F	–	–	–
Rule Count	16	8	8	8	8	2	1	1	2	2	8
a1 : discount = 0%											
a2 : discount = 20%						X			X		
a3 : discount = 25%								X			
a4 : discount = 30%										X	
a5 : discount = 40%					X		X				
a6 : discount = 100%			X								
a7 : impossible	X	X		X							X

```
System.out.println ("Enter destination");
String dist = s.nextLine();
System.out.println ("Enter departure day (Monday-Sunday)");
String day = s.nextline();
System.out.println("Enter no. days for which you'll stay at destination");
int duration = s.nextInt();
int discount = 0;
if (age<2) {  discount = 100;
              System.out.println(discount);
             continue; }
else if (age<18) { discount =40;
                   System.out.println (discount);
                   continue; }
else { if ( dist.toLowerCase().equals ("Germany")) {
    if (day.toLowerCase().equals ("Monday")=false & day = toLower
    Case().equals ("Friday")==false) {  discount=20;
                                        if (duration >=6) {
                                            discount= 30; }}}
else {
 if (day.toLowerCase().equals ("Monday")==false and day=
     toLowerCase().equals ("Friday")==false ) {
          discount= 25;
     }
}
System.out.println (discount);
 continue; }}}
```

- Limited Entry Decision Table :-

Conditions :	Actions :
c1 : age < 2	a_1 discount = 0%
c2 : 2 ≤ age ≤ 18	a_2 discount = 20%
c3 : age > 18	a_3 discount = 25%
c4 : destination is Germany	a_4 discount = 30%
c5 : departure on Monday or Friday	a_5 discount = 40%
c6 : Stay duration ≥ 6 days	a_6 discount = 100%
	a_7 discount = impossible

Q.) For each rule, design the test case :

⇒ Table on page 11.

Test Cases For Given State Chart Diagram:-

Test Case ID	Test Case Input Event (Method)	Test Condn.	Expected Result	State	Pass / Fail
1.1	initialize()			empty	
1.2	push(x)			filled	
1.3	pop	height = 1	print val	empty	
1.4	delete			end state	pass
2.1	initialize()			empty	
2.2	push(x)			filled	
2.3	pop	height > 1	print val	filled	
2.4	pop	height = 1		empty	
2.5	delete			end state	pass
3.1	initialize()			empty	
3.2	push(x)	height = 1		filled	
3.3	push(x)	height > 1		filled	
3.4	pop	height > 1	printval	filled	
3.5	pop	height = 1	print val	empty	
3.6	delete			end state	pass
4.1	initialize()				
4.2	push(x)	height = 1		empty	
4.3	push(x)	height = max-1		filled	
4.4	push(x)	height = max		full	
4.5	pop	height = max-1	print val	full	
4.6	pop	height = 1		filled	pass
4.7	delete			end state	

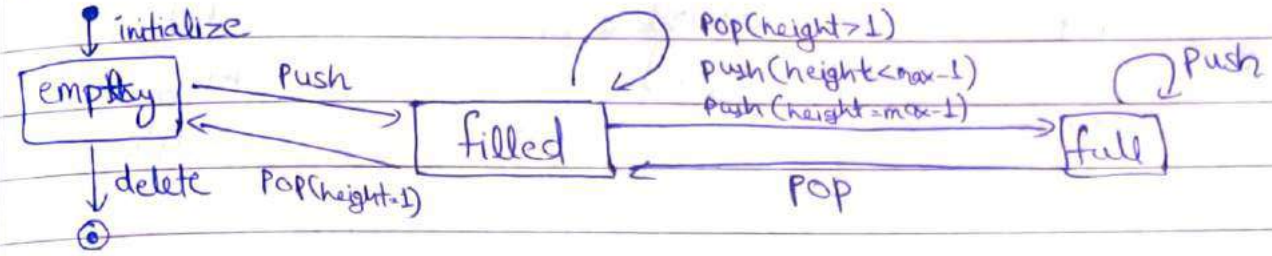
Experiment : 4

Date 28-10-21

Page No. 14

- Problem Statement :-

Generate the state transition table & the test cases



- Source Code :-

```
import java.util.*;
public class Main {
  public static void main(String[] args) {
      Scanner s = new Scanner (System.in);
      int t = s.nextInt();
      while (t-->0) {
   char choice = ' ';
   do {
     System.out.println ("1. Push\n 2. Pop\n 3. Display\n Enter Choice");
     int ch= s.nextInt();
     switch(ch) {
   case 1: { int val = s.nextInt();
             push(val); break; }
   Case 2: { pop(); } break;
   case 3: display(); break;
   default: System.out.println ("Invalid Choice"); }
System.out.println ("Do you wish to continue?");
choice = s.next().charAt(0); }
  while (ch == 'y' || ch == 'Y'),
}}
```

DELTA

5.1	initialize()			empty	
5.2	push(x)	height=0		filled	
5.3	push(x)	height>1		filled	
5.4	pop	height>1	print val	filled	
5.5	pop	height<1		filled	
5.6	pop	height=1		empty	
5.7	delete			end state	pass
6.1	initialize			empty	
6.2	delete			end state	pass

State Transition Table:-

		Resultant State				
State	Event / Method	Start	Empty	Filled	Full	End
Start	initialize()		✓			
	push(x)					
	pop()					
	delete()					
Empty	initialize()					
	push(x)			✓		
	pop()					
	delete					✓
Filled	initialize()					
	push(x)			✓	✓	
	pop()					
	delete()		✓	✓		
Full	initialize()					
	push(x)				✓	
	pop()			✓		
	delete					

```
public static int height = 5;
public static int[] stack = new int [height];
public static int top = 0;
public static void push(int x) {
    if (top == height-1) {
            System.out.println("Filled"); return; }
    else { stack[top] = x; top++; }}

public static void pop() {
    if (top==0) { System.out.println("Empty"); return; }
    else { System.out.println (Stack[top]); top--; }}

public static void display () {
 for(int i=0; i<=top ; i++) {
        System.out.println(stack[i]); }}}
```

Illegal Test Cases

Test Case ID	Test State	Test Event	Expected Res.	Actual Res.	Pass/Fail
7	empty	initialize	Exception	Exception	Pass
8	empty	pop	"	"	"
9	filled	initialize	"	"	"
10	filled	pop(height-0)	"	"	"
11	full	initialize	"	"	"
12	end	any	"	"	"

Experiment: 5(a)

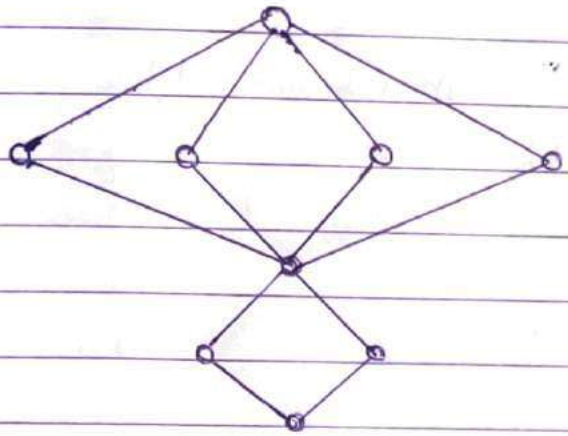
Date 04-11-21

Page No. 16

Problem Statement:-

Calculate the Cyclomatic complexity separately for each of the following.

a)



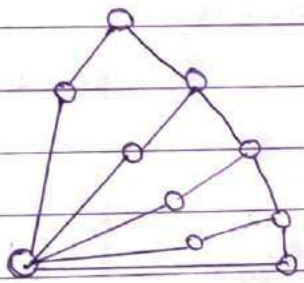
Number of edges = 12

Number of nodes = 9

Number of unconnected components 'p' = 1

Cyclomatic complexity = 12-9+2x1 ⇒ 5

b)



Number of edges = 13

Number of nodes = 10

Number of unconnected component = 1

Cyclomatic complexity = 13-10+2x1 ⇒ 5

Assuming that these 2 control flow graphs are two unconnected parts of the same graph.

e=25, n=19, p=2 ∴ Cyclomatic complexity = e-n+2p = 10.

DELTA

Variable	Defined at Node	Used at
x	1	2,3
y	1	2,4
a	3,4	5

Thus ~~the~~ path with beginning and end node are given as:

Variable	du path (begin end)
x	(1,2)
	(1,3)
y	(1,2)
	(1,4)
a	(3,5)
	(4,5)

Variable	du-paths	Path Feasible	do clear paths
x	1,2	Yes	(1,2) Definition clear
	1-2,3	Yes	(1,2-3) Definition clear
y	1,2	Yes	(1,2) Definition clear
	1-2,4	Yes	(1,2-4) Definition clear
a	3,5	Yes	(3,5) Definition clear
	4,5	Yes	(4,5) Definition clear

All those du paths which are not definitions clear paths are potential troublesome paths. They ~~s~~started du tested and identified on top priority. The path for variable a (3,5) should the tested as there is redefinition of variable at node 4.

Experiment: 5(b)

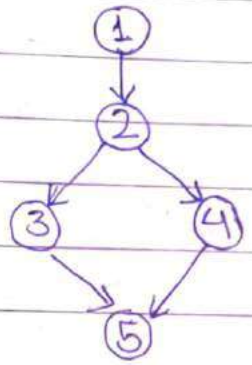
Date 04-11-21

Page No. 17

- Problem Statement:-

Identify the error prone paths (data anomaly) using data flow testing.

1.) read x, y
2.) if (x > y)
3.) a = x + 1;
    else
4.) a = y - 1;
5.) print a;

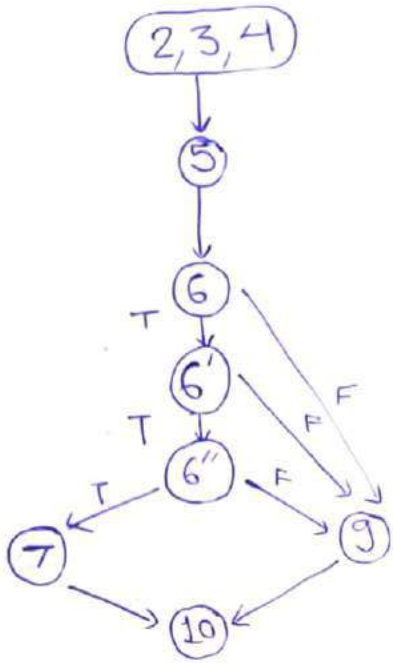


- Source Code:-

```
import java.util.*;
public class Main{
  public static void main (String[] args){
    Scanner s = new scanner (System.in);
    int t = s.nextInt();
    while (t-- > 0){
      int x = s.nextInt();
      int y = s.nextInt();
      int a;
      if (x > y)
        a = x + 1;
      else
        a = y - 1;
      System.out.println(a);
}}}
```

DELTA

• Control Flow Graph



a.)

Test Case ID	Ceiling Value	Business Type	Commercial	Expected O/P	Actual O/P	Pass/Fail
T001	650000	F	N	Y	Y	Pass
T002	3000	F	N	N	N	Pass

b.)

Test Case ID	Test Case Scenario	Input			Expected O/P	Actual O/P	Pass/ Fail
		Ceiling Val.	Business Type	Commercial			
1	Input Ceiling Value, Business Type, Commercial	650000	F	N	Y	Y	Pass
2	"	3000	F	N	N	N	Pass

Experiment: 6

Date 11/11/21

Page No. 13

- Problem Statement:-

```
FUNCTION Document Required;
BEGIN
  VAR Ceilingvalue INTEGER,
  VAR BusinessType, Commercial, DocumentRequired TEXT;
  READ CeilingValue, BusinessType, Commercial;
  IF( Ceilingvalue >=650000 and BusinessType=='F') and Commercial=='N')
      DocumentRequired= 'y';
  ELSE
      DocumentRequired = 'N';
  RETURN DocumentRequired,
  END;
```

i) Modify the control flow graph, if required.

=> No modification required.

- Source Code:-

```
import java.util.*;
public class Main {
public static Void main (String[] args){
    Scanner s= new Scanner (System.in);
     int t = s.nextInt();
while (t-->0) {
   int Ceilingvalue = s.nextInt();
String BusinessType = s.nextLine();
String Commercial = s.nextLine();
char c= documentrequired (Ceilingvalue, BusinessType, Commercial);
System.out.println(c); }}
```

DELTA

c.)

Test Case ID	Test Case Scenario	Test Input			Expected O/P	Actual O/P	Pass / Fail
		Ceiling Value	Business Type	Commercial			
1	Input ceiling value, business Type, commercial	650000	F	N	Y	Y	Pass
2	"	3000	F	N	N	N	Pass

d) Condition Coverage

Test Case ID	Test Case Scenario	Test Input			Expected O/P	Actual O/P	Pass / Fail
		Ceiling Value	Business Type	Commercial			
1	Input Ceiling Val, business type ← commercial	650000	F	N	Y	Y	Pass
2	"	650000	NP	N	N	N	Pass
3	"	650000	F	Y	N	N	Pass
4	"	3000	F	N	N	N	Pass

e.) Multiconditional Coverage

Test Case ID	Test Case Scenario	Test Input			Exp. O/P	Actual O/P	Pass / Fail
		Ceiling Val	Business Type	Commercial			
1	I/P ceiling val, business type, commercial	650000	F	N	Y	Y	Pass
2	"	650000	F	Y	N	N	Pass
3	"	650000	NP	N	N	N	Pass
4	"	650000	NP	Y	N	N	Pass
5	"	3000	F	N	N	N	Pass
6	"	3000	F	Y	N	N	Pass
7	"	3000	NP	N	N	N	Pass
8	"	3000	NP	Y	N	N	Pass

```
public char documentrequired (int Ceilingvalue, String BusinessType, String Commercial)
{ if (Ceilingvalue > 650000 && BusinessType equals ("F") &&
                    commercial.equals("N")) {
  return 'Y';
  else return 'N'; }}
```

ii) Write the test cases for the following code coverings:

a.) Statement Coverage

=> Table on page 18. For test case 1, statements covered '(2,3,4,5,6,6',7,10,11)' Statement Coverage {(9,11)}

For test case 2, statements covered "(1,2,3,4,5,6,9,10,11)" statement coverage {(9,23)}.

b.) ~~Branch~~ Decision Coverage

=> Table on page 18. Test case (1) causes the yes outcome of the decision Test case (2) covers the no outcome of then decision Decision coverage for each test case is 50%.

c.) Branch Coverage

=> For test case (1) edges covered: 5-6, 6-6', 6-6'', 6''-7, 7-10

For test case (2) edges covered: 5-6, 6-9, 9-10

Two edges, 6'-9 & 6''-9 are not covered since they have to do with short circuit operations & hence are not required for edge coverage.

Table on next.

DELTA

f.) Condition/Decision Coverage

Test Case ID	Test Scenario	Test Input			Exp. O/P	Act. O/P	Pass/ Fail
		Ceiling Val	Business Type	Commercial			
1	Input ceiling val, business type, com.	650000	F	N	Y	Y	Pass
2	"	650000	F	Y	N	N	Pass
3	"	650000	NP	Y	N	N	Pass
4	"	650000	F	N	N	N	Pass

e.) Relational Coverage

Test Case ID	Test Scenario	Test Input			Exp. O/P	Act O/P	Pass/ Fail
		Ceiling Val.	Business Type	Commercial			
1	Input ceiling val, business type, com.	650000	F	N	Y	Y	Pass
2	"	700000	F	N	N	N	Pass
3	"	3000	F	N	N	N	Pass

Experiment : 7

Date 18/11/21

Page No. 20

Selenium : An Automation Testing Tool

1.) What is selenium? What is the latest version of selenium?

⇒ Selenium is a free open source automated testing framework used to validate applications across different browsers and platforms. User can use multiple programming languages like Java, C#, python, etc. to create selenium test scripts. Testing done using the selenium testing tool is usually referred to as selenium testing. Selenium software is not just a single tool but a suite of software which include selenium IDE, selenium remote control, webdriver and selenium grid. Selenium 4 is the latest version of selenium.

2.) Briefly discuss features of selenium.

⇒ • Selenium is an open source and portable web testing framework.
• Selenium IDE provides a playback & record failure for authorizing tests without the need to learn a test scripting language.
• Selenium supports various O.S like android, ios, linux, mac, etc.
• Selenium supports programming languages like C#, Java, python, etc.
• It supports parallel test execution which reduces time and increase the efficiency of tests.
• Selenium can be integrated with testing frameworks like Ant and Main for source code compilation and with frameworks like testNG for application testing & generating reports.
• Selenium requires fewer resources as compared to other auto-mation tools.

DELTA

3) How automation testing beats manual testing?

⇒ • Manual testing shows lower accuracy due to higher possibilities of human errors whereas automation testing depicts a higher accuracy due to computer based testing eliminating the chances or errors

• Manual testing needs time when testing is needed at a large scale whereas automation testing easily performs testing at a large scale with utmost efficiency.

• Manual testing takes more time to complete a cycle of testing and thus the turnaround time is higher. Automation testing completes a cycle of testing within record time.

• Manual testing needs more cost as it involves the hiring of expert professionals to perform testing whereas automation testing saves cost incurred as once the software infrastructure is integrated, it works for a long time.

4.) What are the challenges with manual testing?

⇒ • Not reliable: Manual testing is not reliable as there is no standard or criteria available to check out that whether the actual & expected results have been compared.

• Higher Chance of risk: In manual testing, there is a higher chance of risk involved of oversight & mistakes. In the testing process, tests may not finish, may not be attentive, or have too much to do.

• Time consuming: Limited test resources makes manual testing too time consuming.

• Selecting the right testers: It is essential for any business to deploy the right testers to test their software. The testing team should be an efficient team so that they can concentrate on expertise and skills.

Experiment :

Date

Page No.

- Incomplete Coverage: Testing is quite complex when there is a mix of multiple platforms, OS, server, clients etc. Full manual regression testing is impossible.
- When to stop testing: Stopping may be a difficult decision as it needs the core judgement of all testing processes and their significance.

5.) Selenium is used to perform which type of testing?

⇒ Selenium is used to perform automation testing.

6.) What are disadvantages of selenium?

⇒ ·) No reliable technical support from anybody: Since its open source software, there is nobody to provide reliable technical support.

·) It supports web based applications only: Selenium automates web mobile web apps but doesn't support desktop applications or hybrid mobiles.

·) Difficult to use, takes more time to create test cases: Selenium web driver has only programming interface and no IDE, so steps are written for each and every task and take more time.

·) No test tool integration for test management: No tool is available in selenium's tool suite to support test management.

·) No built-in reporting facility:- Selenium web driver doesn't have built in result report facility and help is taken from either JUnit or TestNG testing framework to generate test reports.

7.) What is parallel testing?

⇒ It is a software testing type in which multiple versions or subcomponents of an application are tested with some input

DELTA®

on different systems simultaneously to reduce test execution time. The purpose of parallel testing is finding out if legacy version and new version are behaving the same or differently and ensuring whether new version is more efficient or not.

8) Compare Selenium with other few similar automations tools.

⇒

Features	QTP	IBM RFT	Selenium
License	Required	Required	Open Source
Cost	High	High	Free as it is open source
Customer Support	High	High	Low
Hw Res. Consumptn	High	High	Low
Coding Experience	Not much	Required	Should be very good along with tech capabilities of integrating the framework.
Env Support	Only Windows	Only Windows	Windows, linux, solaris OS is browsers & JVM or IS support exists.
Lang Support	VB Script	Java & C#	Java, C#, ruby, Python, Pearl, PHP, Javascript

Experiment :

Date

Page No.

9.) Briefly discuss selenium suite of tools.

=> Selenium test suite consists of 4 major components:

i) Selenium IDE: It is implemented as firefox extension which provides record & playback functionality on test scripts. It allows testers to export recorded scripts in many languages.

ii) Selenium Remote Control: Selenium R.C. allows testers to write automated web application UI test in any of the supported programming lang. It involves Http proxy server which enables the browser to believe that the web applicat^{n} being tested comes from domain provided by proxy server.

iii) Selenium Web driver: Selenium web driver is the successor of selenium RC and is by far the most important component of selenium suite. It provides a programming interface to create and execute test cases. Test scripts are written in order to identify web pages and then desired action on those elements.

iv) Selenium Grid: Selenium grid is an important component of selenium suite which allows user to run test on various machines against different browsers in parallel.

10.) Why eclipse is required for selenium?

=> Eclipse assures that the selenium webdriver has all the dependencies which are needed for development of test cases. Before using selenium webdriver for Java for the test case development, eclipse needs to be configured. It results in the webdriver's correct configuration and its availability for the development of web automation test cases.

DELTA

Experiment : 8

Date 25/11/21

Page No.

- Problem Statement :- Study of Win Runner Testing Tool.

1.) How do you analyze test results in WinRunner tool & report the defects.

⇒ When you finish any test in WinRunner, it displays the result in a report format. The report lags general information about the test run i.e. date, operator made and total run time. Also, the report details include all the major events that occured during the run such as checkpoints, error messages, system messages or user messages. A mismatch can be found in the report panel by comparing the actual result & the expected result if a test run fails due to a defect in the application being tested, you can report information about the deflect directly from the test results window. This information is sent via email to the Quality Assurance Manager who tracks to deflect until it is fined.

2.) What are the different modes of recording in WinRunner?

⇒ There are two types of recording in WinRunner:

- Context Sensitive Recording : Which records the operations user performs on the application by identifying graphical user interface (GUI) objects. WinRunner identifies all the objects in user's windows that have been clicked on like menus, windows, lists, buttons & type of operation user perform such as enable, select, etc.

- Analog Recording : Records keyboards input, mouse clicks and the precise x & y coordinates travelled by mouse pointer across the screen i.e. WinRunner records exact coordinates travelled by the mouse.

DELTA

3.) What is the reasons that WinRunner fails to identify GUI object?

⇒ WinRunner fails to identify an object in GUI due to various reasons some of which are:

1) The object is not a standard windows object.

2.) If the browser used is not compatible with the WinRunner version, the GUI map editor will not be able to learn any of the objects displayed in the browser window.

4.) What do you mean by logical name of the objects?

⇒ When user clicks an object, WinRunner assigns the object a logical name which is usually the object's text label. Logical names make it easy for user to read the script. For example:

When user have selected the 'Order No' Checkbox, WinRunner has recorded the following statement in WinRunner TSL:

button_set("Order No", ON);

"Order No" is the object's logical name.

An object's logical name is determined by its class. In most cases, the logical name is the label that appears on an object.

Experiment :

Date

Page No.

5) What is the purpose of different record methods?

⇒ 1) Record: It instructs WinRunner to record all operations performed on a GUI object. This is the default record method for all classes. The only exception is the static class for which the default is pass up.

2.) Pass up: Their instructs WinRunner to record on operation performed in this class as an operation performed on the element containing the object. Usually, this element is a window, and the operation is recorded as win_mouse_click.

3) As object: This instructs WinRunner to record all operations performed on a GUI object as though its class was the "object" class.

4) Ignore: This instructs WinRunner to disregard all operations performed in the class.

DELTA®

Experiment : 9

Date 25/11/21

Page No.

. Problem Statement :- Study of test management toal (QA Complete)

1) What is bug leakage & bug release?

⇒ Bug release is when software or application is handed over to the testing team knowing that the defect is present in a release. During this the priority and security of bug is low as bug can removed before the final handover.
Bug leakage is something when the bug is discovered by the end users or customer and not by the testing team while testing the software.

2) What is difference between build & release?

⇒ Build: It is a number given to installable software that is given to the testing team by the development team.
Release: It is a number given to installable software that is handed over to the customer by the tester or developer.

3) What is the difference between the QA and software testing?

⇒ The role of QA is to monitor the quality of the process used to produce the software while the software testing is the process of ensuring the functionality of final product meets the user's requirement.

4) What are the automation challenges that SQA team faces while testing?

⇒ i) Mastering the automation tool ii) Reusability of automation script
iii) Adaptability of test case for automat^{n} iv) Automating complex test cases

DELTA

5.) What is agile testing and what is importance of agile testing?

⇒ Agile testing is software testing, is testing using agile methodology. The importance of testing is that unlike normal testing process, this testing does not wait for the development team to complete the coding first then doing testing. The coding and testing both goes simultaneously. It requires continuous customer interaction.

6.) Explain what should your QA documents should include

⇒ Testing document should include:

- List the no. of defects detected as per security level.
- Explain each requirement or business function in detail.
- Inspection Reports
- Configurations
- Test plans & test cases
- Bug reports
- User manuals
- Prepare separate reports for managers & users.

Experiment: 10

Date 25/11/21

Page No.

- Problem Statement:- Learn how to raise and report bugs using Bug Tracking tool.

1) Who benefits most from using the Bugzilla bug tracking tool?

⇒ Organizations involved in multiple software development projects especially organizations with large team of developers and testers would find Bugzilla very helpful in their efforts, as it provides them with a convenient centralized location for tracking ongoing initiatives and helps cut down on duplicate and unnecessary work. All members of the Devceps team, no matter what their involvement is during development, will find the software simple to use. Bugzilla tool provides two access modes' user and administrator. The admin role provides users & administrator. The administrator role provides users with greater access to security, workflow, monitoring and grouping features

2) Why is Jira used?

⇒ Jira is a part of family of products designed to help teams of all types manage work. Originally, Jira was designed as a bug and issue tracker but is now a powerful work management tool for all kinds of use cases from requirements and test case management to agile software development.

Bugs are just a name for to-do's stemming from problems within the software a team is building. It is important for teams to view all tasks and bugs in the backlog so they can prioritise big picture goals.

DELTA

3) What is the deployment environment for Bugzilla?

⇒ Bugzilla has been developed in LAMP where it stands for linux, Apache, MySQL & PHP technology.

4) What is Bugzilla?

⇒ Bugzilla is an open source issue 1 bug tracking system that allows developers to keep track of outstanding problems with their product. Written in perl & uses MySQL database. Bugzilla is a defect tracking tool, however, it can be used as a test management tool by linking with other test case management tools.

5.) What are bugzilla features?

⇒ • Bugzilla is powerful & it has advanced searching capabilities.
- Bugzilla supports user configurable email notifactions whenever the bug status changes.
- Bugzilla provides inter bug dependency track & graphic rep.
- Bugzilla allows users to attach bug supportive files & manage it.
- It has complete security avoid runs under Perl's taint method.
- Bugzilla supports a robust, stable RDBMS backend.
- It supports localized web user interface.
- Bugzilla has a smooth upgrade pathway among different versions.