Software Testing Methodologies

BITS Pilani

Prashant Joshi

Module 5: Agenda

Module 5: Code Based Testing (1/2)

Topic 5.1	Code Based Testing Overview
Topic 5.2	Path Testing
Topic 5.3	Examples
Topic 5.4	Case Study

innovate achieve lead

BITS Pilani
Pilani Campus

Topic 5.1: Code Based Testing Overview

Code Based Testing

- Input to test design is source code or a program structure

- Salient Features
  - More Rigorous than specification testing WRT code
  - Lower Level than specification.
  - Validation WRT specification may not happen as input is code

Code Based Testing

- Techniques
  - Statement Testing
  - Branch Testing
  - Multiple Condition Testing
  - Loop Testing
  - Path Testing
  - Modified Path Testing (McCabe Path)
  - Dataflow Testing
  - Transaction Flow Testing
  - …

Statement Testing

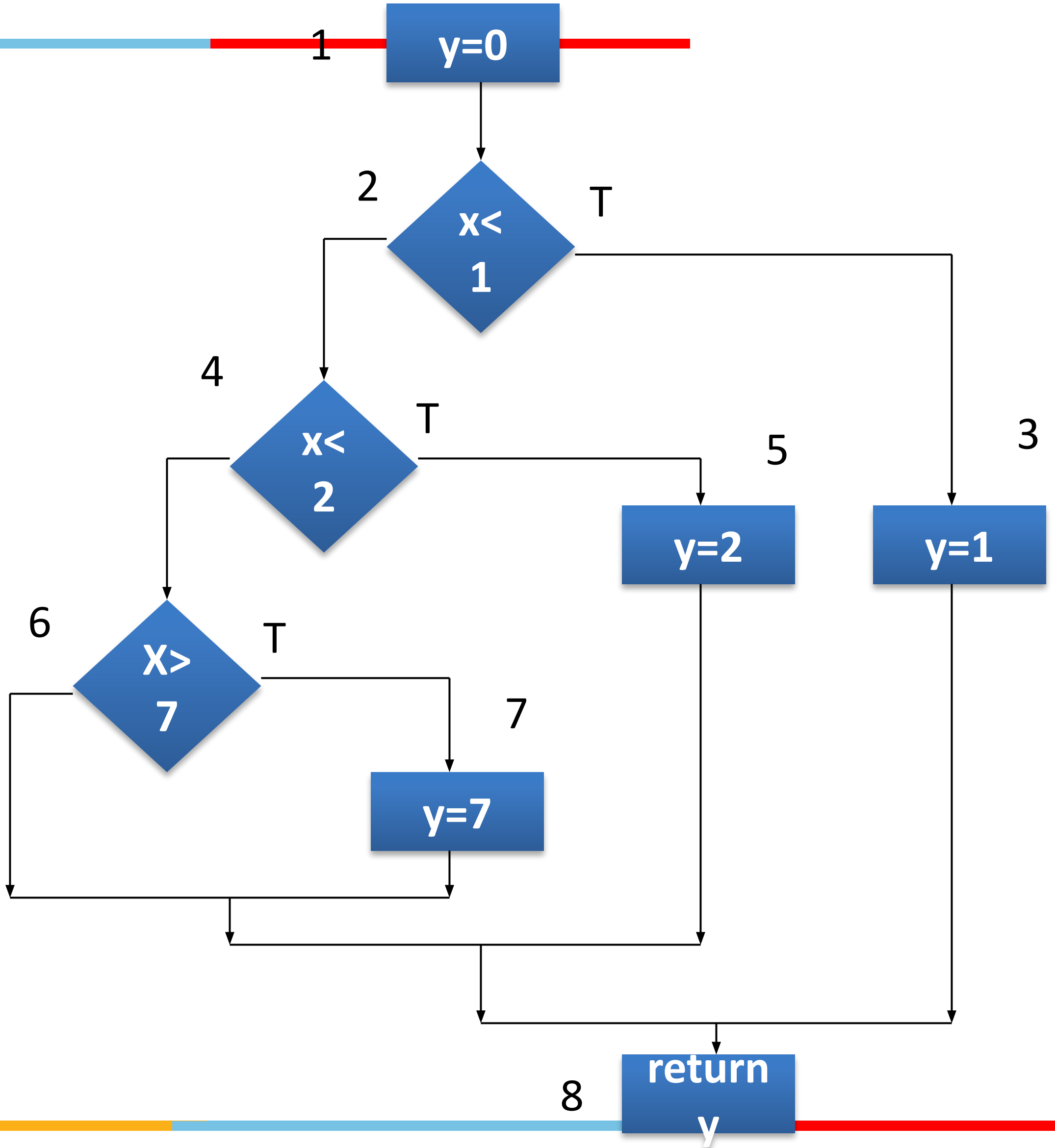
- Basic Concept
  - Every Statement in the program (code) should be covered at least once during testing
- Types of Statement
  - An assignment Statement
  - An input statement
  - An output statement
  - A function/procedure/subroutine call
  - A return statement
  - A predicate of condition statements
    - IF-THEN-ELSE
    - WHILE-DO/DO-WHILE
    - SWITCH
- A variable declaration is not a statement

Statement Testing

Example

```
int F(int x)
1    y=0;
2,3 If (x<1) {y=1};
    else {
4,5  if(x<2) {y=2};
      else
6,7    if (x>7) {y=7};
    }
8   return y;
    }
```

Test #1: x=0
1, 3, 8
Test #2:x=1
1, 2, 5, 8
Test #3:x=10
1, 2, 4, 7, 8



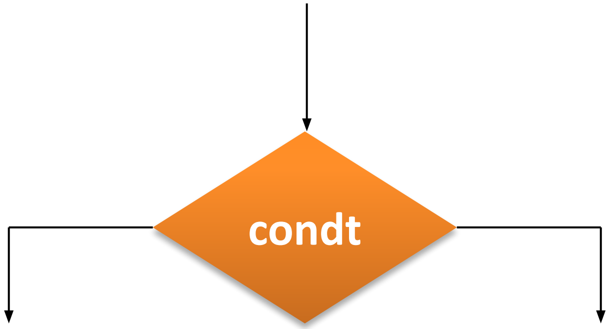
Branch Testing

- Basic Concept
  - Every branch in the program (code) should be executed at least once during testing.
  - What does this coverage constitute?
    - IF-THEN-ELSE
    - WHILE-DO
    - SWITCH
  - Review the earlier example and check what branches we cover with the 3 test cases
    - Check with Test #4:x=5

Branch Testing

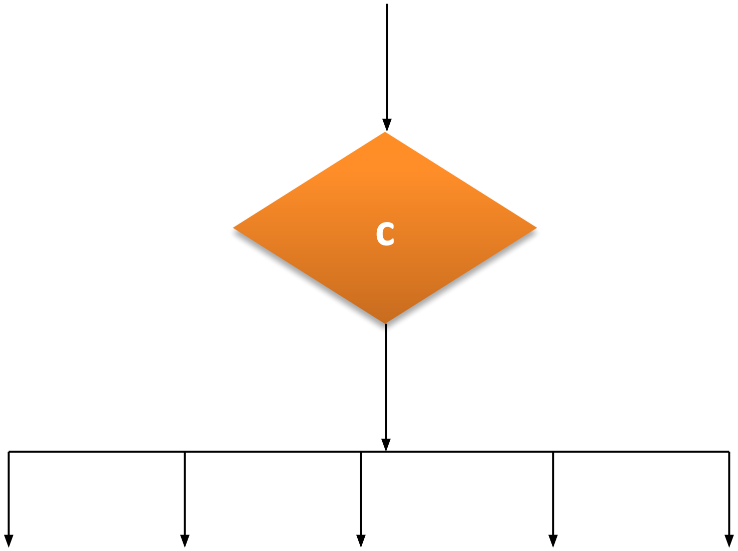
IF Statement

- IF (condition) THEN, ELSE



SWITCH

- SWITCH-CASE



Salient Features

- More demanding
- When branch testing is satisfied the statement testing is also satisfied

Multiple Condition Testing

- This is testing of condition with complex predicates (OR, NOT and AND)
- IF C1 THEN, ELSE □ Branch testing ~ multiple condition
- IF (C1 AND C2 AND C3) THEN, ELSE □ Multiple condition

- In case of first condition there is a single condition so the values can be true or false
- In case of second condition, it is a complex predicate made up C1 AND C2 AND C3.
- To test this “Test all combinations of simple predicates”

Multiple Condition

Example

```
input (x, y)
if (x>0) and (y<1) then z=1
    P1         P2    else z=0
If (x>10) and (z>0) then u=1
    Q1          Q2    else u=0
```

Design test cases for P1, P2 and Q1 and Q2

Each P1 and P2, and Q1 and Q2 for 4 conditions in pairs

Multiple Condition

Example

```
input (x, y)
if (x>0) and (y<1) then z=1
    P1         P2   else z=0
If (x>10) and (z>0) then u=1
    Q1          Q2   else u=0
```

Design test cases for P1, P2 and Q1, Q2

Each P1 & P2 and Q1 &Q2 for 4 conditions in pairs

P1	P2
x>0	y<1
T	T
T	F
F	T
F	F

	x	y
Test #1	15	0
Test #2	15	5
Test #3	-1	0
Test #4	-1	5

Ensure that the case worked out is for values of x and y to evaluate Q1 and Q2 and not a single statement under consideration alone

Q1	Q2
x>10	z>0
T	T
T	F
F	T
F	F

	x	y
Test #1	15	0
Test #2	15	5
Test #3	5	0
Test #4	-1	0

Multiple Condition

Salient Features

- Very demanding testing technique
- Frequently used with high reliability system requirements
- Non-executable combination may exist

Software Testing Methodologies

BITS Pilani

Prashant Joshi

innovate achieve lead

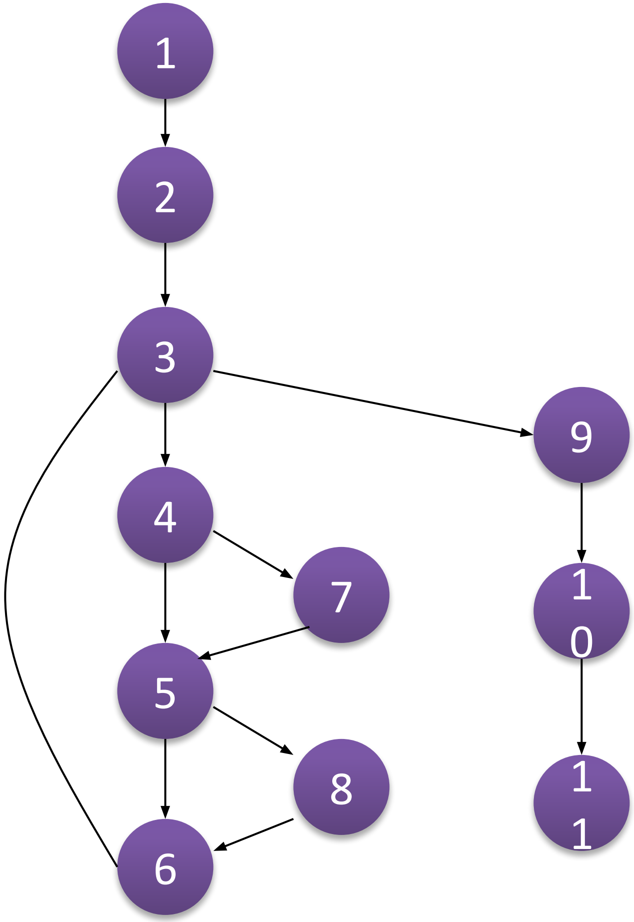
BITS Pilani
Pilani Campus

Topic 5.2: Path Testing

Control Flow Graph

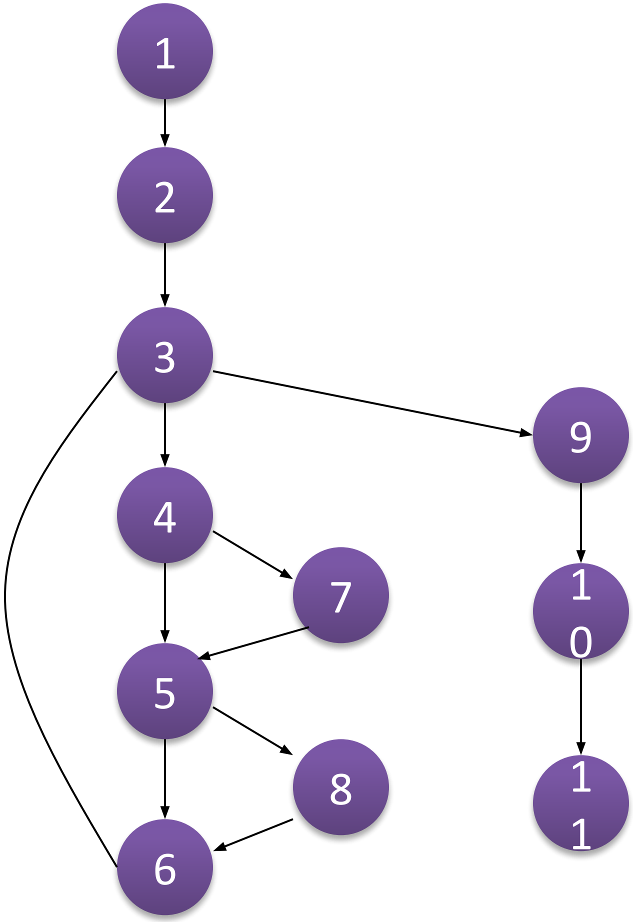
- A control Flow Graph consists of Nodes and Edges. Edges are between nodes and are directed
- A Node: A statement (i.e. executable atomic entity in a program)
  - An assignment statement
  - An input/output statement
  - Predicate of a condition
- An Edge: An edge represents a flow of control between two nodes/statements
- A control flow graph can be used to represent nodes with software modules or functions to depict a full functionality

Control Flow Graph



- A path in a control flow graph of a program is a sequence of nodes (statements) in a control flow graph
- A path represents a possible execution of the program

Loop Testing



Simple Loop

- Test #1: Skip the loop
- Test #2: Iterate the loop once
- Test #3: Iterate the loop several times (normal Case)
- Test #4: Iterate max number of times
- Test #5: Iterate the loop (max-1) number of times

```
mathtable (x, y)
int i, j;
For (i=x;i<=x;i++) {
  print(j);
  j=j+j;
}
```

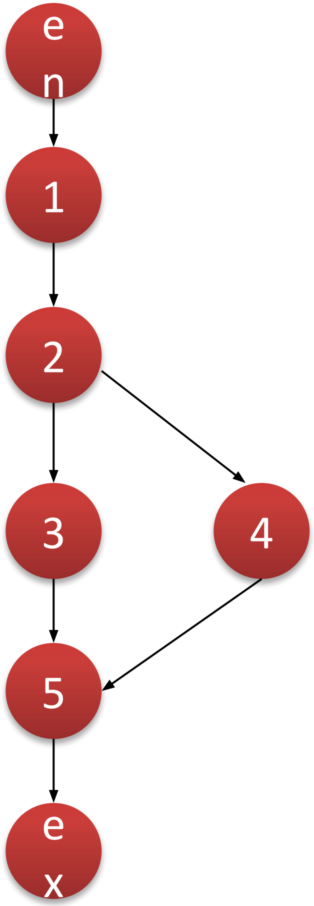
Work out the above example as per the loop testing concept

Path Testing

Basic Concept

- To design a test suite (a set of test cases) for which every possible path is executed at least once

```
1     input (x)
2, 3 if (x<10) y=0;
4       else y=1;
5  output (y)
```



- Path#1: en, 1, 2, 3, 5, ex
- Path#2: en, 1, 2, 4, 5, ex

- Test #1: 5
- Test #2: 15

- All paths may not be executable

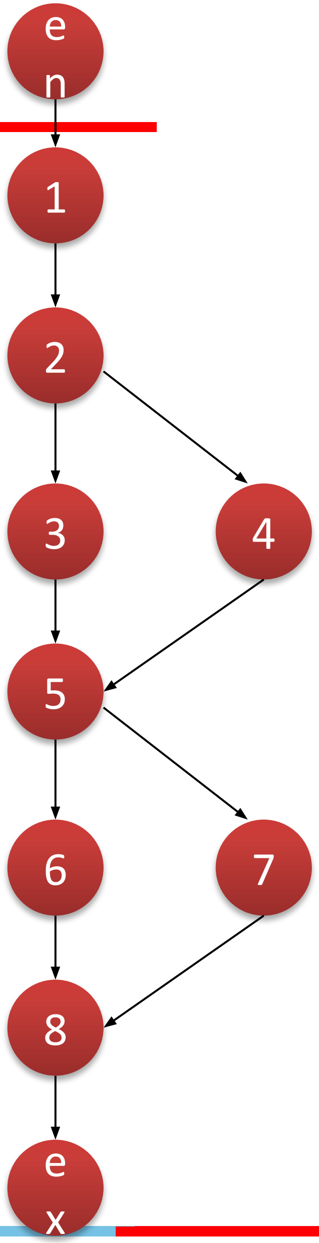
Path Testing - Example

```
input (x)
if (x<10) then y=0
          else y=1
if (x<30) then z=1
          else z=2
output (y, z)
```

Branch testing : 2 branches

Test
Path#1: 1, 2, 3, 5, 6, 8 T1 : x=5
Path#2: 1, 2, 3, 5, 7, 8 T2 : x=?
Path#3: 1, 2, 4, 5, 6, 8 T3 : x=15
Path#4: 1, 2, 4, 5, 7, 8 T4 : x=35

(x<10) and x>30) is not possible
Therefore, Path#2 is not possible



Path Testing

# of branches = 1

Paths

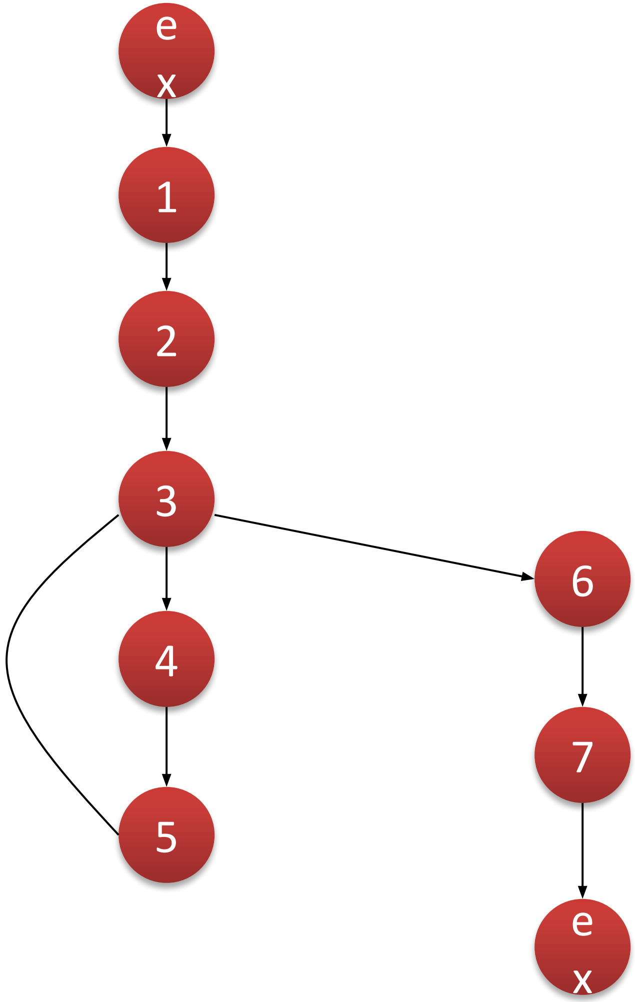
P1: 1 2 3 6 7

P2: 1 2 3 4 5 3 6 7

P3: 1 2 3 4 5 3 4 5 3 6 7

Such we can have infinite paths

Such situations use loop testing



McCabe Path Testing

Complexity, Effort, # of tests….

Program 1

Program 2

- Complexity
  #1 > #2
- Effort in testing
  #1 > #2
- Number of Tests
  #1 > #2

Cyclomatic Number

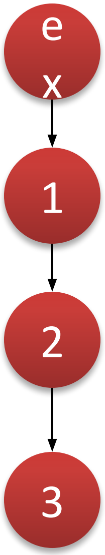
McCabe’s cyclomatic number (end 70’s)

$v = e - n + 2$

e: # of edges in a control graph

n: # of nodes in a control graph

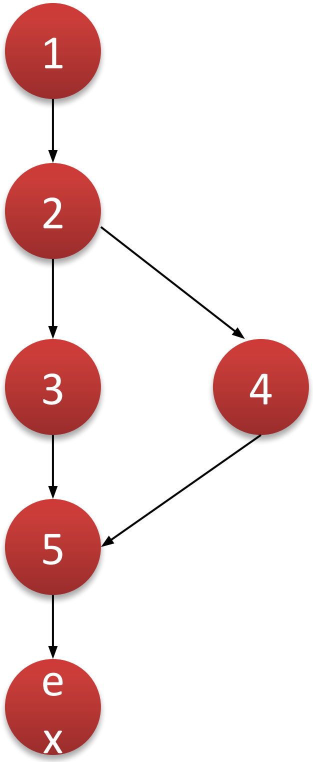
Examples:



$e = 3$

$n = 4$

$$\begin{aligned} v &= e - n + 2 \\ &= 3 - 4 + 2 \\ &= 1 \end{aligned}$$



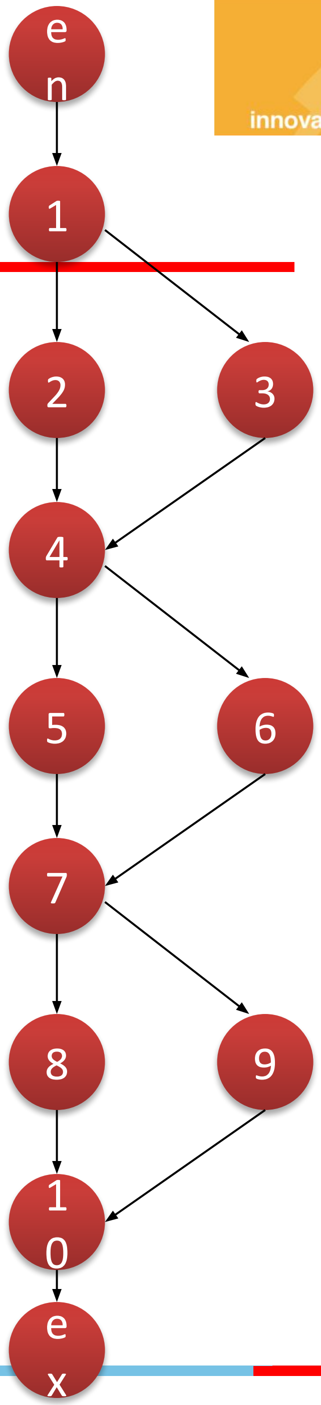
$e = 6$

$n = 6$

$$\begin{aligned} v &= e - n + 2 \\ &= 6 - 6 + 2 \\ &= 2 \end{aligned}$$

McCabe Path

Number of Paths?



McCabe Path

$e = 14$

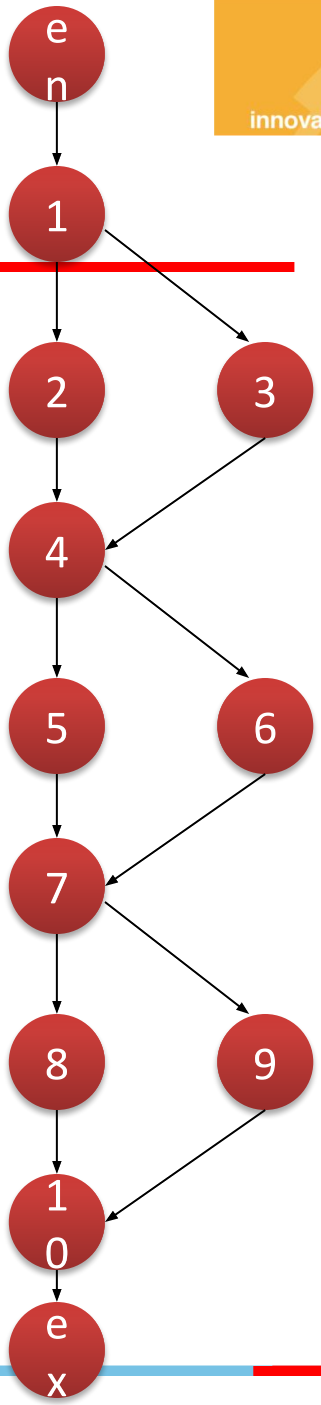
$n = 12$

$v = e - n + 2$

$= 14 - 12 + 2$

$= 4$

Please observe carefully the four distinct paths



McCabe – Testing Criteria

Testing Criteria

- Every branch must be executed at least once
- At least “v” distinct paths must be executed
    - $v = e - n + 2$
- # of test cases is a function of program complexity

Software Testing Methodologies

BITS Pilani

Prashant Joshi

innovate achieve lead

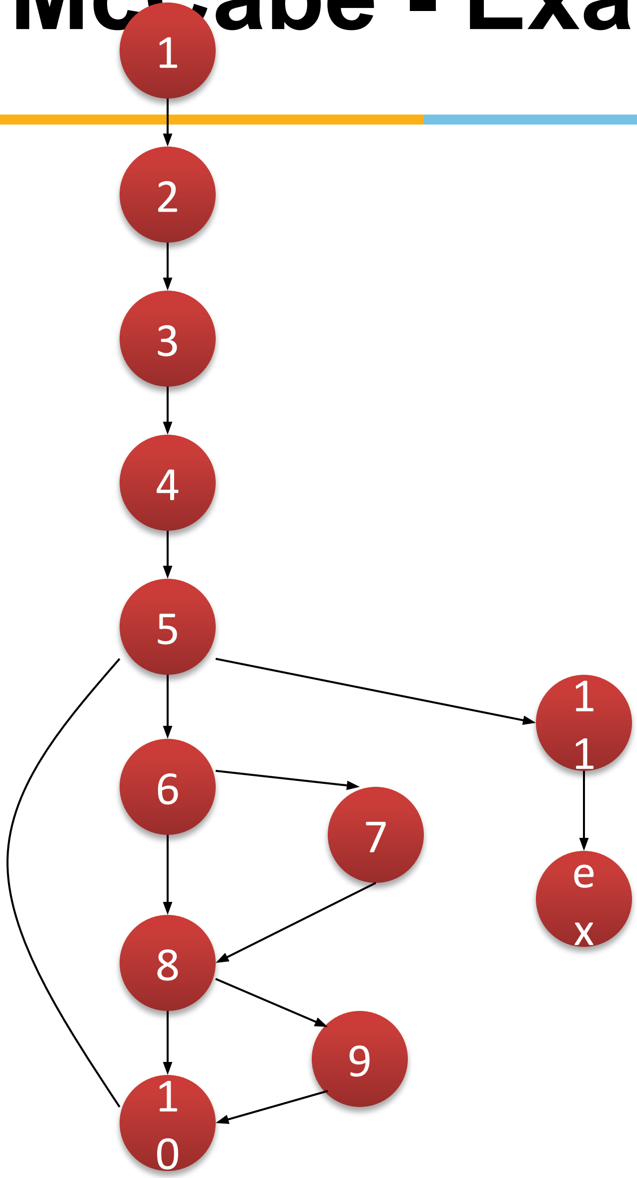
BITS Pilani
Pilani Campus

Topic 5.3: Examples

McCabe - Example

```
1       input (n, a)
2       max = a[1]
3       min = a[1]
4       i = 2
5       While I <=n do
6, 7      if max < a[i] then max = a[i]
8, 9      if min > a[i] then min = a[i]
10        i = i + 1
        endwhile
11      output (max, min)
```

McCabe - Example



e = 15

n = 13

V = 15 – 13 + 2

= 4

4 distinct test cases (minimum)

1. 1 2 3 4 5 11 [n=1 a=5]
2. 1 2 3 4 5 6 8 10 5 11 [n=2 a=5 5]
3. 1 2 3 4 5 6 7 8 10 5 11 [n=2 a= 2 4]
4. 1 2 3 4 5 6 8 9 10 5 11 [n=2 a= 4 2]

1 2 3 4 5 6 7 8 9 10 5 11

n = 2 a= ?

This path is not executable

McCabe Path Testing

- Complexity number gives bound for number of test cases
- Number of test cases is a function of complexity
- Complexity can be used for design as well
- Number of paths can be computed from

v = number of regions + 1 as well.

BITS Pilani

Software Testing Methodologies

Prashant Joshi

innovate achieve lead

BITS Pilani
Pilani Campus

Topic 5.4: Case Study

Contacts Application

- Create a Contact
- Retrieve a Contact
- Update a Contact
- Delete a Contact
- Share Contact
  - Bluetooth
  - Email
  - WhatsApp
- Fields in a contact

Software Testing Methodologies

BITS Pilani

Prashant Joshi