

# Introduction to Software Testing

## Chapter 2.3

### Graph Coverage for Source Code

**Paul Ammann & Jeff Offutt**

[www.introsoftwaretesting.com](http://www.introsoftwaretesting.com)

**This version is abbreviated.  
Data flow coverage is omitted.**

## Overview

- The most usual application of graph criteria is to **program source**
- **Graph** : Usually the control flow graph (CFG)
- **Node coverage** : execute every **statement**
- **Edge coverage** : execute every **branch**
- **Loops** : looping structures such as for loops, while loops, etc.
- **Data flow coverage** : augment the CFG
  - defs are statements that assign values to variables
  - uses are statements that use variables

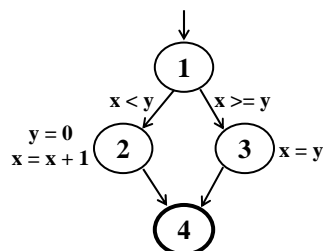
**Data flow not covered in undergraduate course**

# Control Flow Graphs

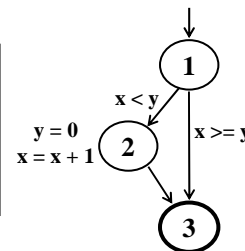
- A CFG models all executions of a method by describing control structures
- **Nodes** : statements or sequences of statements (basic blocks)
- **Edges** : transfers of control
- **Basic Block** : A sequence of statements such that if the first statement is executed, all statements will be (no branches)
- CFGs are sometimes annotated with extra information
  - branch predicates
  - defs
  - uses
- Rules for translating statements into graphs ...

## CFG : The if Statement

```
if (x < y)
{
  y = 0;
  x = x + 1;
}
else
{
  x = y;
}
```

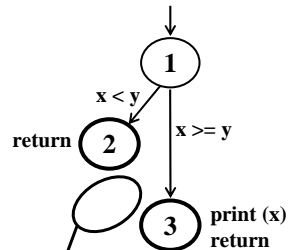


```
if (x < y)
{
  y = 0;
  x = x + 1;
}
```



## CFG : The if-Return Statement

```
if (x < y)
{
  return;
}
print (x);
return;
```

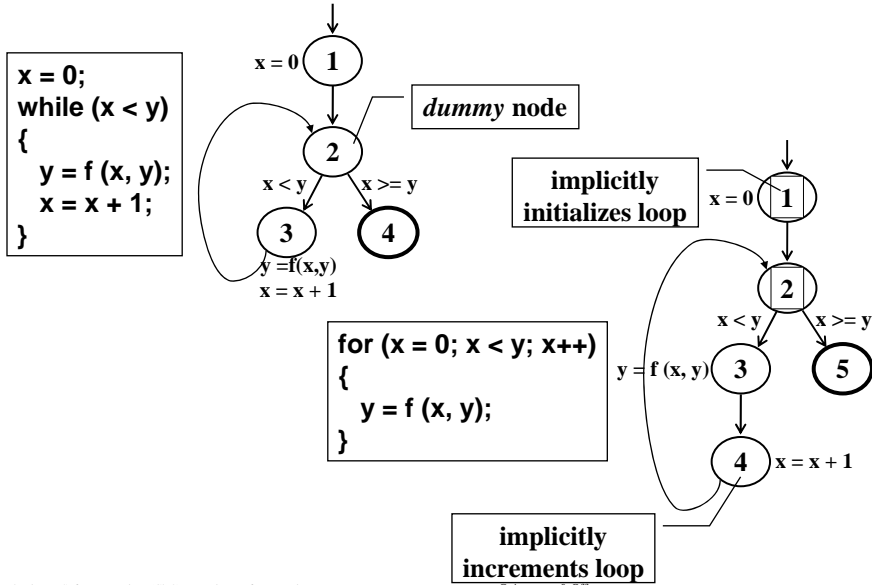


**NO edge from node 2 to 3.  
The return nodes must be distinct.**

## Loops

- Loops require “*extra*” nodes to be added
- Nodes that do not represent statements or basic blocks

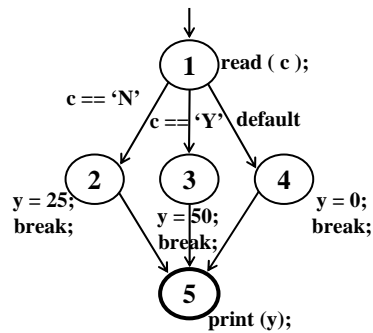
## CFG : while and for Loops



## CFG : The case (switch) Structure

```

read ( c );
switch ( c )
{
  case 'N':
    y = 25;
    break;
  case 'Y':
    y = 50;
    break;
  default:
    y = 0;
    break;
}
print (y);
        
```



## Example Control Flow – Stats

```

public static void computeStats (int [ ] numbers)
{
    int length = numbers.length;
    double med, var, sd, mean, sum, varsum;

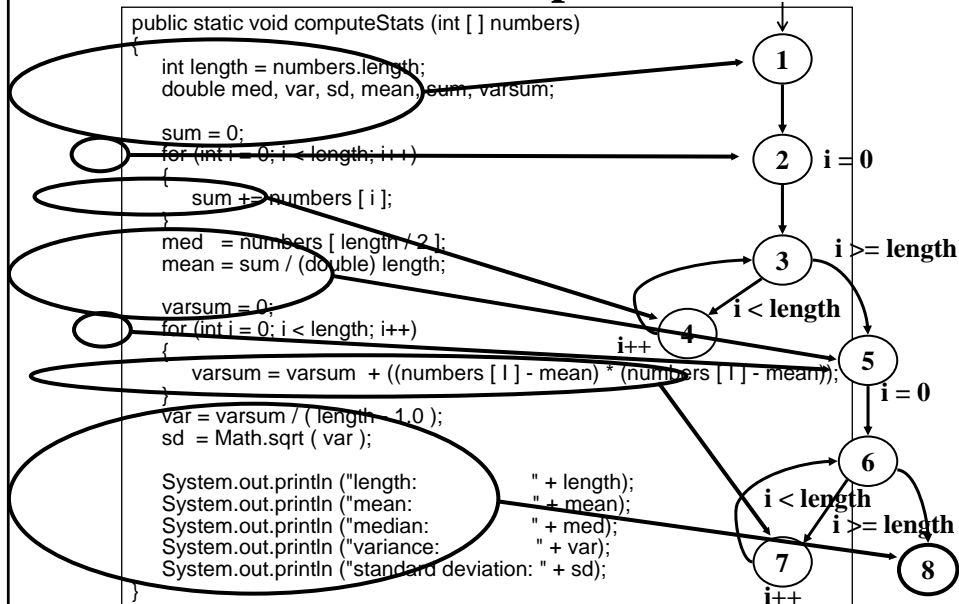
    sum = 0;
    for (int i = 0; i < length; i++)
    {
        sum += numbers [ i ];
    }
    med = numbers [ length / 2 ];
    mean = sum / (double) length;

    varsum = 0;
    for (int i = 0; i < length; i++)
    {
        varsum = varsum + ((numbers [ i ] - mean) * (numbers [ i ] - mean));
    }
    var = varsum / ( length - 1.0 );
    sd = Math.sqrt ( var );

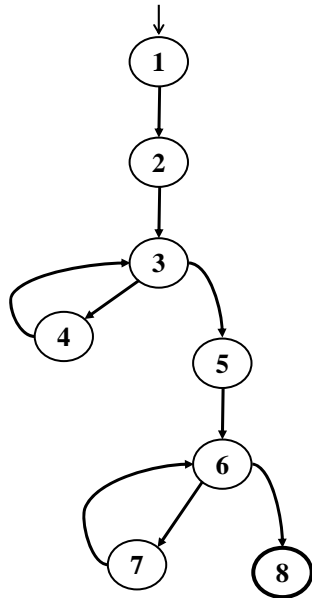
    System.out.println ("length:          " + length);
    System.out.println ("mean:           " + mean);
    System.out.println ("median:        " + med);
    System.out.println ("variance:      " + var);
    System.out.println ("standard deviation: " + sd);
}

```

## Control Flow Graph for Stats

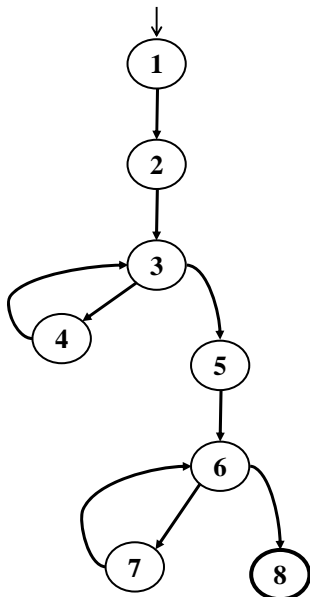


## Control Flow TRs and Test Paths – EC



Edge Coverage	
TR	Test Path
A. [ 1, 2 ]	[ 1, 2, 3, 4, 3, 5, 6, 7, 6, 8 ]
B. [ 2, 3 ]	
C. [ 3, 4 ]	
D. [ 3, 5 ]	
E. [ 4, 3 ]	
F. [ 5, 6 ]	
G. [ 6, 7 ]	
H. [ 6, 8 ]	
I. [ 7, 6 ]	

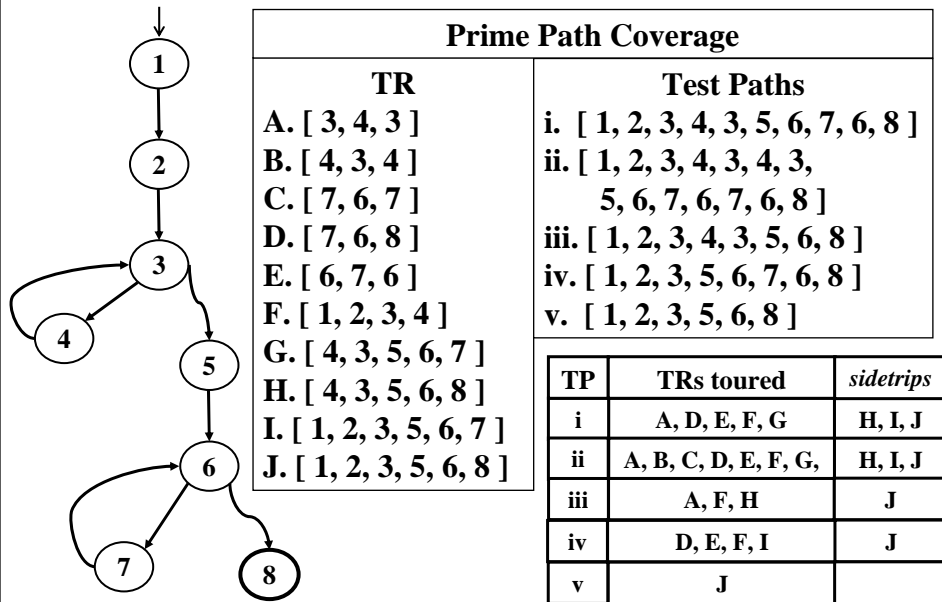
## Control Flow TRs and Test Paths – EPC



Edge-Pair Coverage		
TR	Test Paths	
A. [ 1, 2, 3 ]	i. [ 1, 2, 3, 4, 3, 5, 6, 7, 6, 8 ]	
B. [ 2, 3, 4 ]	ii. [ 1, 2, 3, 5, 6, 8 ]	
C. [ 2, 3, 5 ]	iii. [ 1, 2, 3, 4, 3, 4, 3, 5, 6, 7, 6, 7, 6, 8 ]	
D. [ 3, 4, 3 ]		
E. [ 3, 5, 6 ]		
F. [ 4, 3, 5 ]		
G. [ 5, 6, 7 ]		
H. [ 5, 6, 8 ]		
I. [ 6, 7, 6 ]		
J. [ 7, 6, 8 ]		
K. [ 4, 3, 4 ]		
L. [ 7, 6, 7 ]		

TP	TRs toured	sidetrips
i	A, B, D, E, F, G, I, J	C, H
ii	A, C, E, H	
iii	A, B, C, D, E, F, G, I, J, K, L	H

## Control Flow TRs and Test Paths – PPC



Prime Path Coverage	
TR	Test Paths
A. [ 3, 4, 3 ]	i. [ 1, 2, 3, 4, 3, 5, 6, 7, 6, 8 ]
B. [ 4, 3, 4 ]	ii. [ 1, 2, 3, 4, 3, 4, 3,
C. [ 7, 6, 7 ]	5, 6, 7, 6, 7, 6, 8 ]
D. [ 7, 6, 8 ]	iii. [ 1, 2, 3, 4, 3, 5, 6, 8 ]
E. [ 6, 7, 6 ]	iv. [ 1, 2, 3, 5, 6, 7, 6, 8 ]
F. [ 1, 2, 3, 4 ]	v. [ 1, 2, 3, 5, 6, 8 ]
G. [ 4, 3, 5, 6, 7 ]	
H. [ 4, 3, 5, 6, 8 ]	
I. [ 1, 2, 3, 5, 6, 7 ]	
J. [ 1, 2, 3, 5, 6, 8 ]	

TP	TRs toured	sidetrips
i	A, D, E, F, G	H, I, J
ii	A, B, C, D, E, F, G,	H, I, J
iii	A, F, H	J
iv	D, E, F, I	J
v	J	

## Summary

- **Applying the graph test criteria to control flow graphs is relatively straightforward**
  - Most of the developmental research work was done with CFGs
- **A few subtle decisions must be made to translate control structures into the graph**
- **Some tools will assign each statement to a unique node**
  - These slides and the book uses basic blocks
  - Coverage is the same, although the bookkeeping will differ