Cpt S 422: Software Engineering Principles II

White-box testing

Dr. Venera Arnaoudova

WASHINGTON STATE UNIVERSITY

Outline

- Control flow coverage
  - Statement, Edge, Condition, Path coverage
- Data flow coverage
  - Definitions-Usages of data
- Analyzing coverage data
- Integration testing
  - Coupling-based criteria
- Conclusions
  - Generating test data, Marick's Recommendations

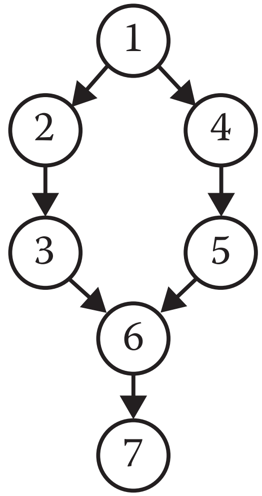
Control Flow Graph (CFG) - Basic Definitions

- Directed graph
- Nodes are blocks of sequential statements
- Edges are transfers of control
- Edges may be labeled with predicate representing the condition of control transfer
- Directed Graphs can also be represented with an adjacency matrix

Basics of CFG

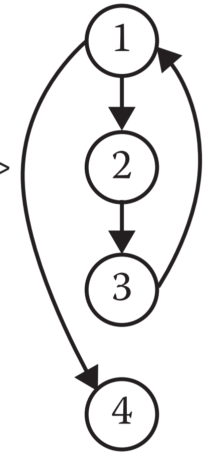
If–Then–Else

```
If <condition>
   Then
      <then statements>
   Else
      <else statements>
End If
<next statement>
```



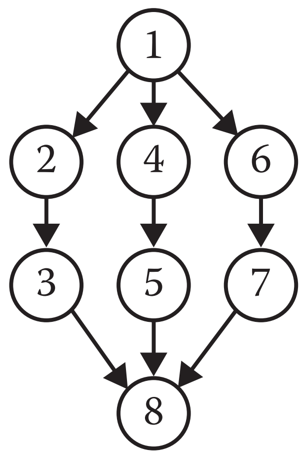
Pretest loop

```
While <condition>
    <repeated body>
End While
<next statement>
```



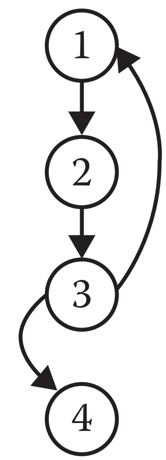
Case/Switch

```
Case n of 3
   n=1:
      <case 1  statements>
   n=2:
      <case 2  statements>
   n=3:
      <case 3  statements>
End Case
```



Posttest loop

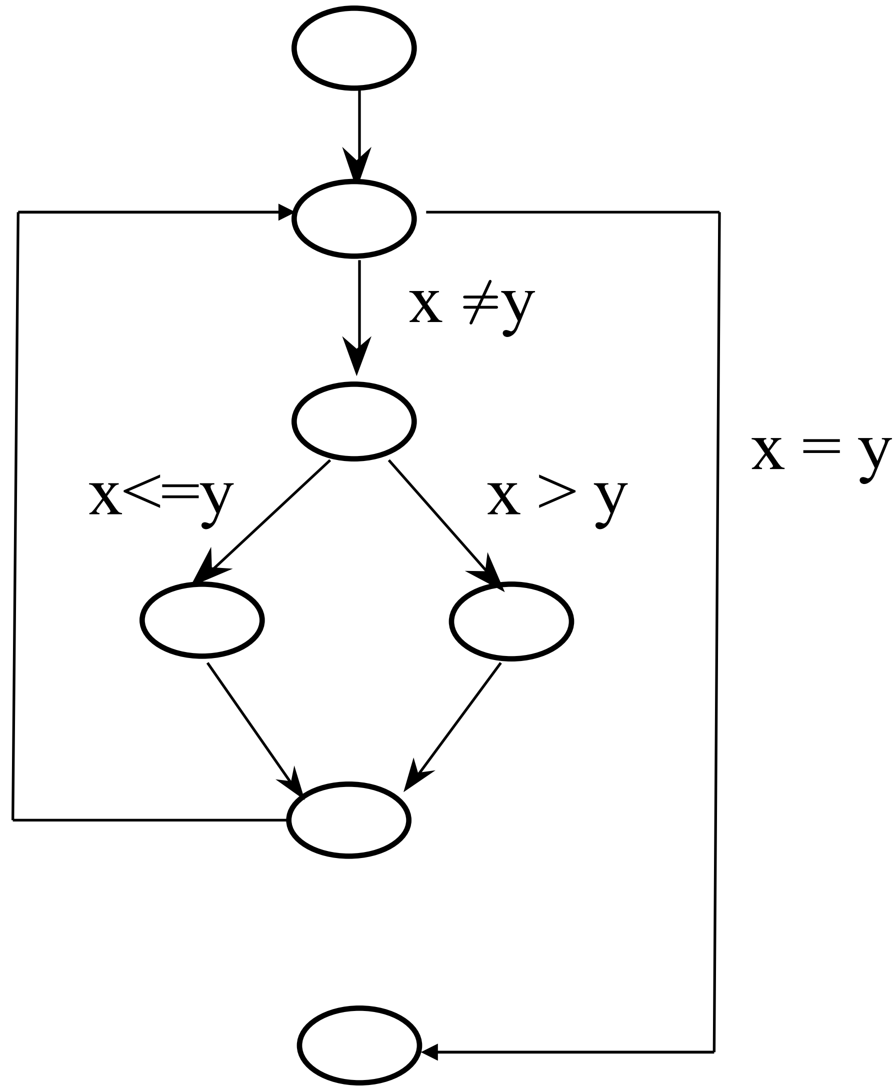
```
Do
    <repeated body>
Until <condition>
<next statement>
```



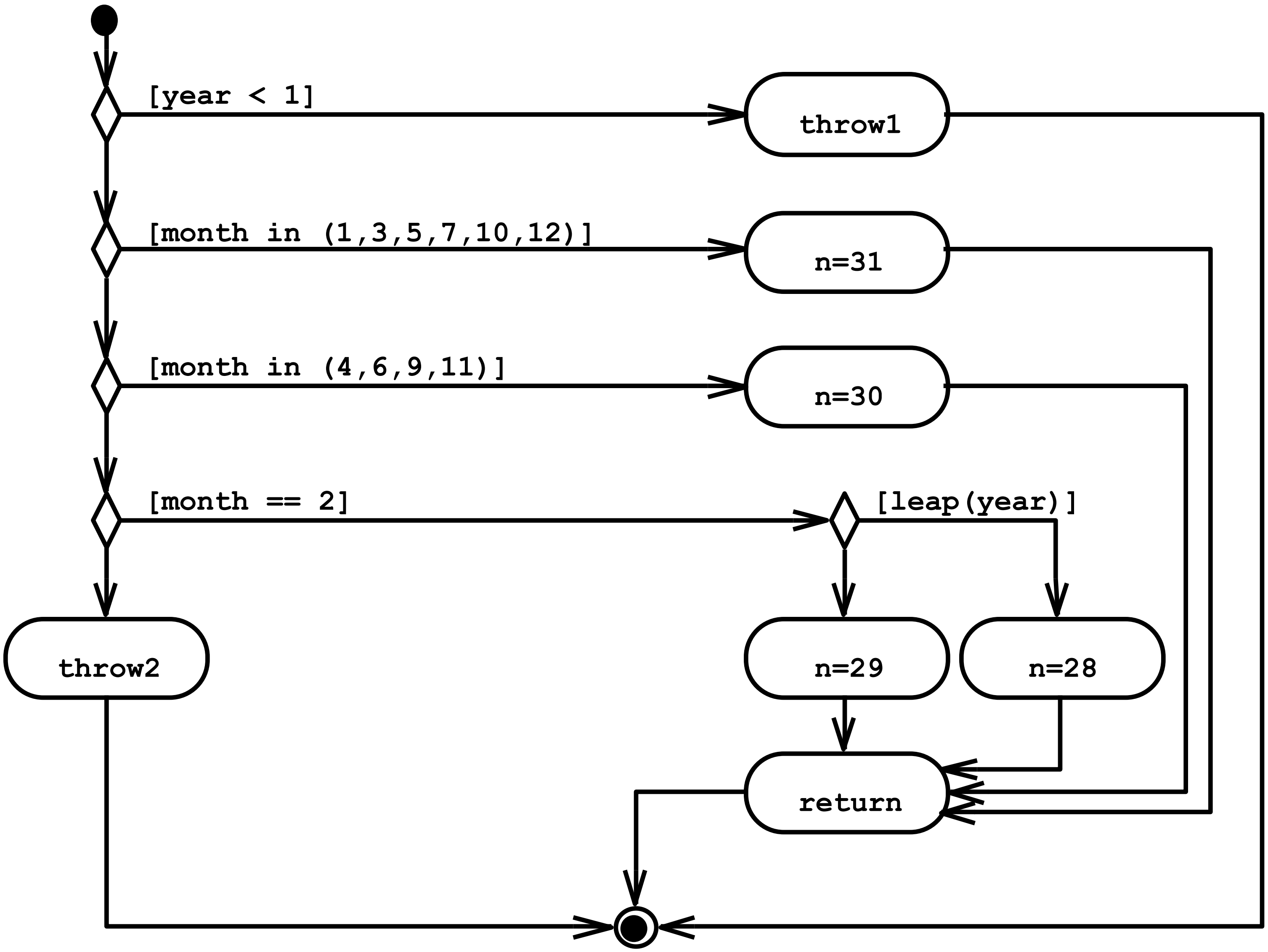
Note that some of those nodes can be grouped

CFG - Example

```
read(x);
read(y);
while x ≠y loop
  if x>y then
          x := x - y;
  else
          y := y - x;
  end if;
end loop;
gcd := x;
```



UML Activity Diagram



Statement/Node Coverage

- Statement coverage: faults cannot be discovered if the parts containing them are not executed
- Equivalent to covering all nodes in CFG
- Executing a statement is a weak guarantee of correctness, but easy to achieve
- In practice, several inputs will execute the same statements

Statement coverage - Incompleteness

- Statement coverage may lead to incompleteness

```
if x < 0 then
     x := -x;
end if
z := x;
```

A negative x would result in the coverage of all statements.
But not exercising x >= 0 would not cover all cases.
The program behavior for x >= 0 may turn out to be wrong and need to be tested.

Edge Coverage

- **Edge (branch) coverage criterion:** Select a test set T such that, by executing P for each t in T, each edge of P's control flow graph is traversed at least once

- Exercise all conditions that govern control flow of the program with true and false values

- To satisfy the edge coverage criterion for the example on the previous slide we would need to have a test with x<0 and a test with x>=0

Another Example - Search Algorithm

```
counter:= 0;
found := false;                                                  ≤
if number_of_items ≠ 0 then counter :=1;
   while (not found) and counter < number_of_items loop
      if elementAt(counter) = desired_element then
            found := true;
      end if;
      counter := counter + 1;
   end loop;
end if;
if found then write (“the desired element exists”);
else write (“the desired element does not exists”);
end if;
```

- ❑ Provide a test set to satisfies the Edge Coverage criterion

Test Set

- We can choose a test set as follows (|T| = 2):
  - t1: A collection with 0 items
  - t2: A collection with 3 items, the second being the desired one
- With t1, we cover the else statements of the first and last ifs
- With t2, the body of loop is executed twice, once executing the **then** branch and finding the element.
- The edge coverage criterion is fulfilled but the error is not discovered by the test set as not all possible values of the *constituents* of the **while loop** condition have been exercised

Condition Coverage

- Further strengthen edge coverage

- **Condition Coverage Criterion:** Select a test set T such that, by executing P for each element in T, each edge of P's control flow graph is traversed, and all possible values of the constituents of compound conditions are exercised at least once

- **Compound conditions:** C1 **and** C2 **or** C3 … where Ci's are relational expressions or boolean variables (atomic conditions)

- **Modified Condition Decision Coverage (MCDC)**: Only combinations of values such that every Ci drives the overall condition truth value twice (true and false). (Almost identical to ACC)

Uncovering Hidden Edges

```
if c1 and c2 then
   st;
else
   sf;
end if;
```

```
if c1 then
   if c2 then
        st;
   else
        sf;
   end if;
else
   sf;
end if;
```

- Two equivalent programs but chances are that you would write the left one

- Edge coverage
  - would not compulsorily cover the "hidden" edges,
  - Ex.: C2 = false might not be covered

- Condition coverage would!

MCDC - Example

- International Standard DO-178B for airborne systems certification (1992)
- Example : $A \wedge (B \vee C)$

	ABC	Result
1	TTT	T
2	TTF	T
3	TFT	T
4	TFF	F
5	FTT	F
6	FTF	F
7	FFT	F
8	FFF	F

Pairs of points where each constituent is toggled and the change is visible in the outcome:

- A : (1,5), or (2,6), or (3,7)
- B : (2,4)
- C : (3,4)

Minimal set to satisfy MCDC:

- (2,3,4,6) or (2,3,4,7)

That is 4 test cases (instead of 8 for all possible combinations)

Path Coverage

- **Path Coverage Criterion:** Select a test T such that, by executing P for each t in T, all paths leading from the initial to the final node of P's control flow graph are traversed

- In practice, however, the number of paths is too large, if not infinite

- Some paths are infeasible

- It is key to determine "critical paths"

Example

```
if x ≠0 then
      y := 5;
else
      z := z - x;
end if;
if z > 1 then
      z := z / x;
else
      z := 0;
end if;
```

T1 = {<x=0, z =1>, <x =1, z=3>}:

Executes all edges but do not show risk of division by 0

T2 = {<x=0, z =3>, <x =1, z=1>}:

Would find the problem

T1 $\cup$ T2:

All paths are covered

Dealing with Loops

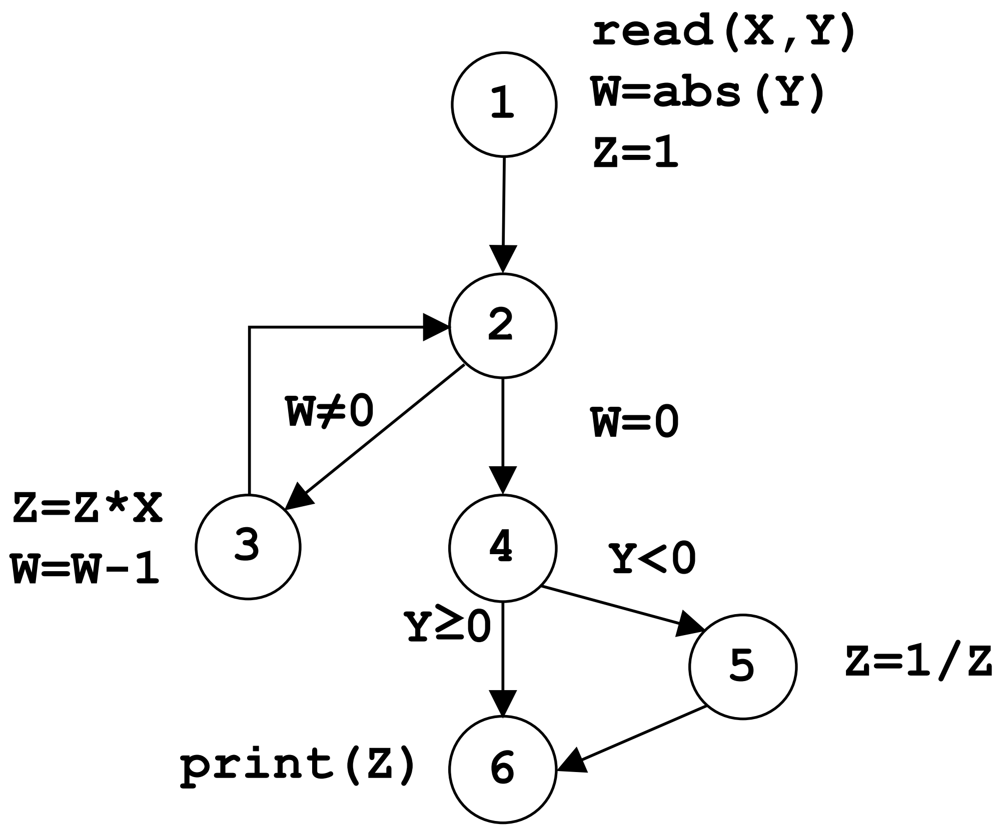
- Look for conditions that execute loops
  - Zero times
  - A maximum number of times
  - An average number of times (statistical criterion)

- For example, in the search algorithm
  - Skipping the loop
  - Executing the loop once or twice and then finding the element
  - Searching without finding the desired element

Power Function Example

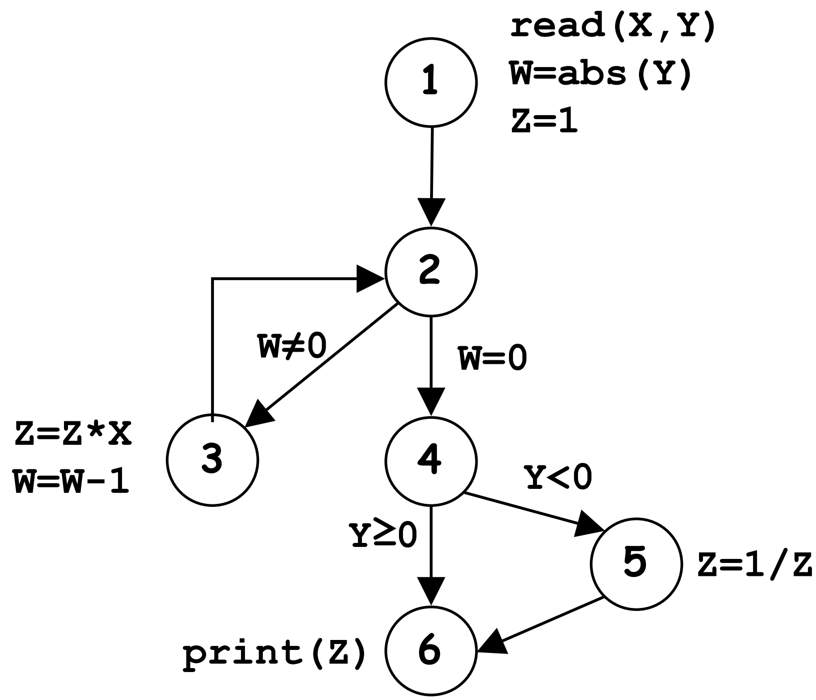
Program computing Z=X^Y

```
BEGIN
  read (X, Y) ;
  W = abs(Y) ;
  Z = 1 ;
  WHILE (W <> 0) DO
    Z = Z * X ;
    W = W - 1 ;
  END
  IF (Y < 0) THEN
    Z = 1 / Z ;
  END
  print (Z) ;
END
```



Power Function Example (cont.)

- All paths
  - Infeasible path:
    - ✓ 1 ->2 -> 4 -> 5-> 6
  - Infinite number of paths:
    - ✓ As many ways to iterate 2 -> (3 -> 2)* as values of abs(Y)
- All branches
  - Two test cases are enough:
    - ✓ Y<0 : 1 -> 2 -> (3 ->2)+ -> 4 -> 5 -> 6
    - ✓ Y≥0 : 1 -> 2 -> (3 -> 2)* -> 4 -> 6
- All statements
  - One test case is enough:
    - ✓ Y<0 : 1 -> 2 -> (3 ->2)+ ->4 -> 5 -> 6



Deriving Input Values

- Not all statements are reachable in real world programs
- It is not always possible to decide automatically if a statement is reachable and the percentage of reachable statements
- When one does not reach 100% coverage, it is therefore difficult to determine the reason
- Tools are needed to support this activity – and there exist good tools
- Research focuses on search algorithms to help automate coverage
- In general, control flow testing is, more applicable to testing in the small