

# **Introduction to Testing**

**Neelam Gupta**

**The University of Arizona  
Tucson, Arizona, USA**

# Testing

Why do we test?

- judge quality
- discover problems

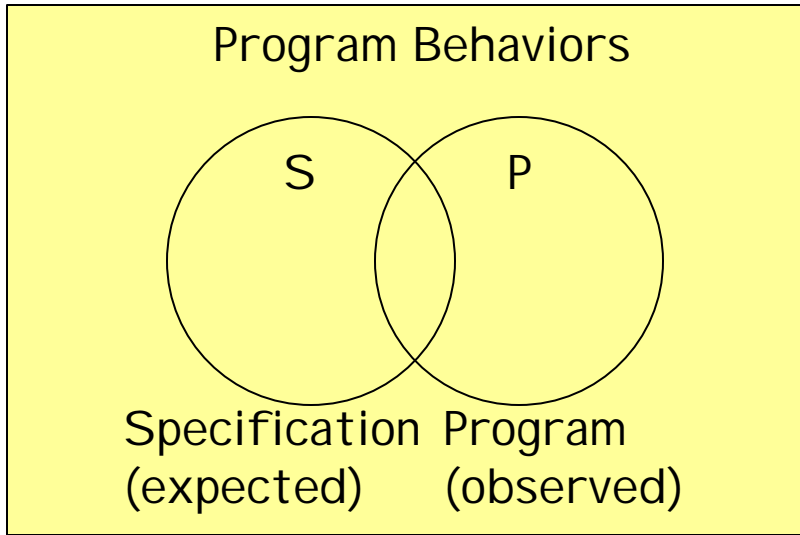
## Terminology

- Error: mistake
- Fault: result of an error
  - Commission
  - Omission

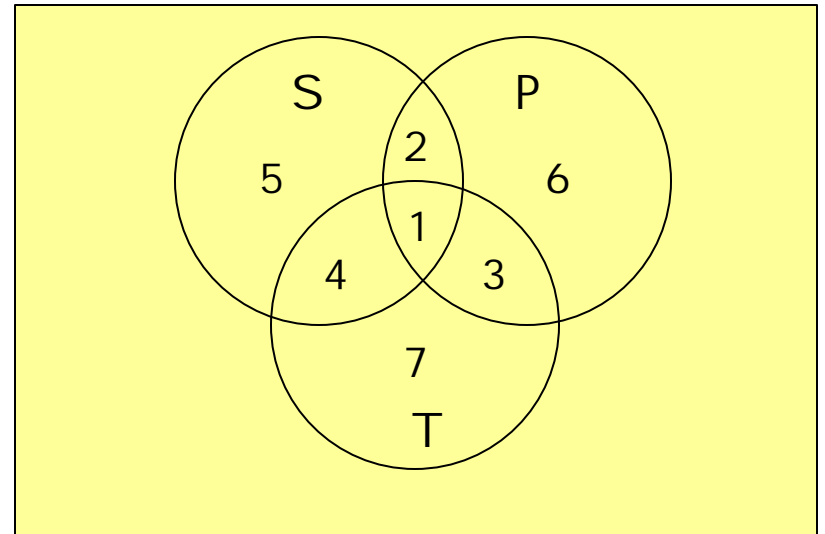
Failure: occurs when a fault executes.

Incident: symptom(s) associated with a failure.

Test cases: A set of inputs and a list of expected outputs for each input.



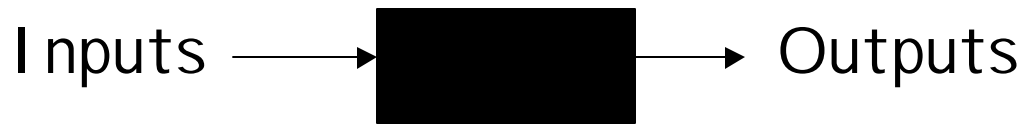
Specified and Implemented Program Behaviors



Specification, Implemented and Tested Behaviors

# Two Approaches to Test case Identification

## 1. Functional Testing (Black Box Testing)

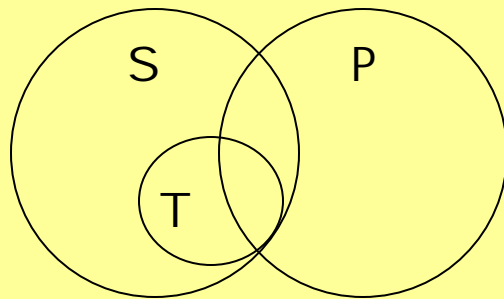


### Advantages:

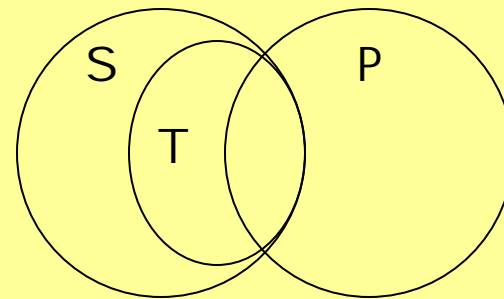
- a) Independent of implementation.
- b) Test case development in parallel with implementation.

### Disadvantages:

- a) Redundancies among test cases.
- b) Possibilities of gaps of untested software.



Method A



Method B

Comparing functional test identification methods.

## 2. Structural Testing (White Box Testing)

Program implementation is used to derive test cases. Test coverage criteria is used to measure the extent of coverage.

Different structural coverage criteria:

- i. All statement coverage
- ii. All branches coverage
- iii. All path coverage
- iv. All definition-use coverage

All Statement Coverage: generate a set  $s$  of inputs such that each statement in the given program is exercised by at least one input in the set.

Equivalently for each statement  $S$  in the program, there exists at least one input  $I$  in the set  $S$ , such that statement  $S$  is executed when the program is executed with input  $I$ .

Similarly, for all branch coverage criteria, the goal is to generate a test suite  $S$  of inputs, such that each branch outcome in the control flow graph of the program is exercised by some input in the test suite.

If a test suite satisfies all statement coverage criteria, it does not necessarily imply that it satisfies the all branch coverage criteria.

Consider the following program:

```
x = 2  
if y>0 then  
    x=5  
endif  
x = 10
```

A test suite satisfying all statement coverage criteria may not cover the false outcome of the branch corresponding to the if statement.

All branch coverage criteria is stronger than all statements coverage criteria. A test suite satisfying all branch coverage criteria always satisfies all statement coverage criteria.

All path coverage criteria is stronger than all branch coverage criteria since it considers execution of all possible combinations of branch outcomes.

Definition-use coverage criteria is concerned with generating input that exercises a given definition of a variable and its corresponding use by a statement executed later.

Definition - Each occurrence of a variable on the left hand side of an assignment is a definition of the variable as it updates the value of the variable.

e.g.,  $x = y + z$  is a definition of  $x$ .

Use - Each occurrence of a variable on the right hand side of an assignment statement or in a boolean predicate is called a use of a variable.

e.g., (a) there is a use of variable  $y$  in  $x=y+z$

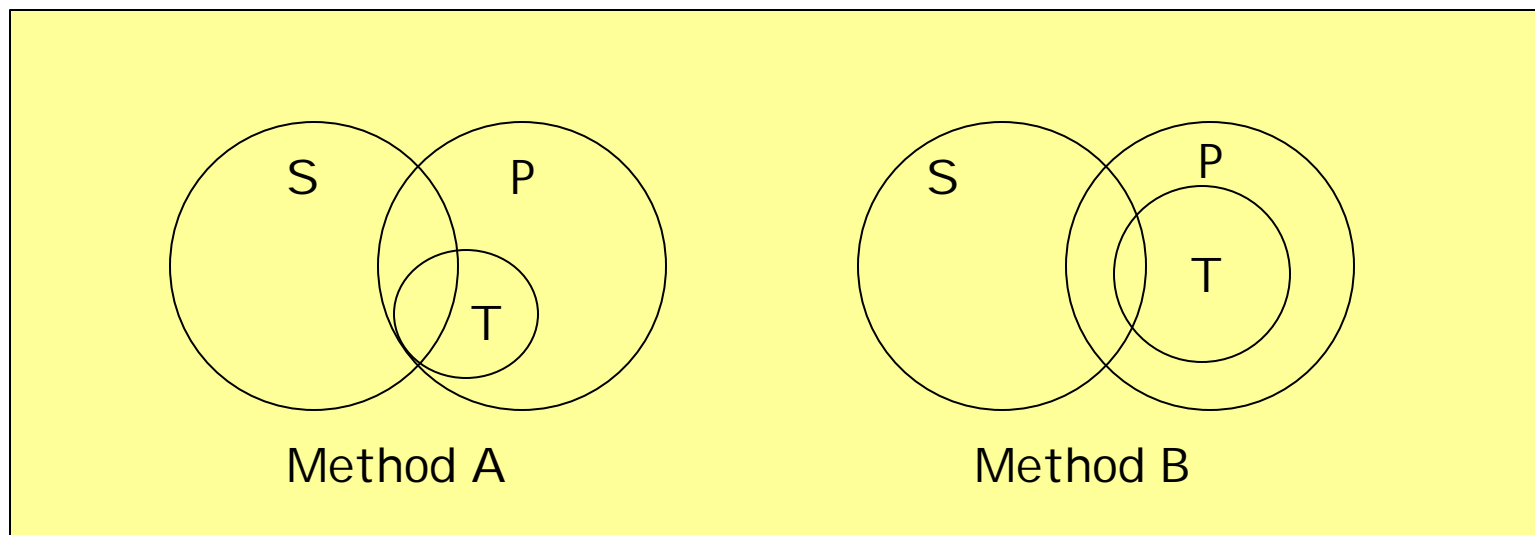
(b) there is a use of  $z$  in the boolean predicate  
if  $(z+x)>10$  then

Consider the program (1)  $x=z+y$ ; (2)  $z=\sin(x)$

$x(1,2)$  forms a definition-use pair since there is a definition of  $x$  in statement (1) that is used in statement (2).

The goal of all definition-use pairs coverage criteria is to generate inputs such that each definition-use pair in a program is executed by some input.

We will discuss each of these structural coverage criteria in detail later.



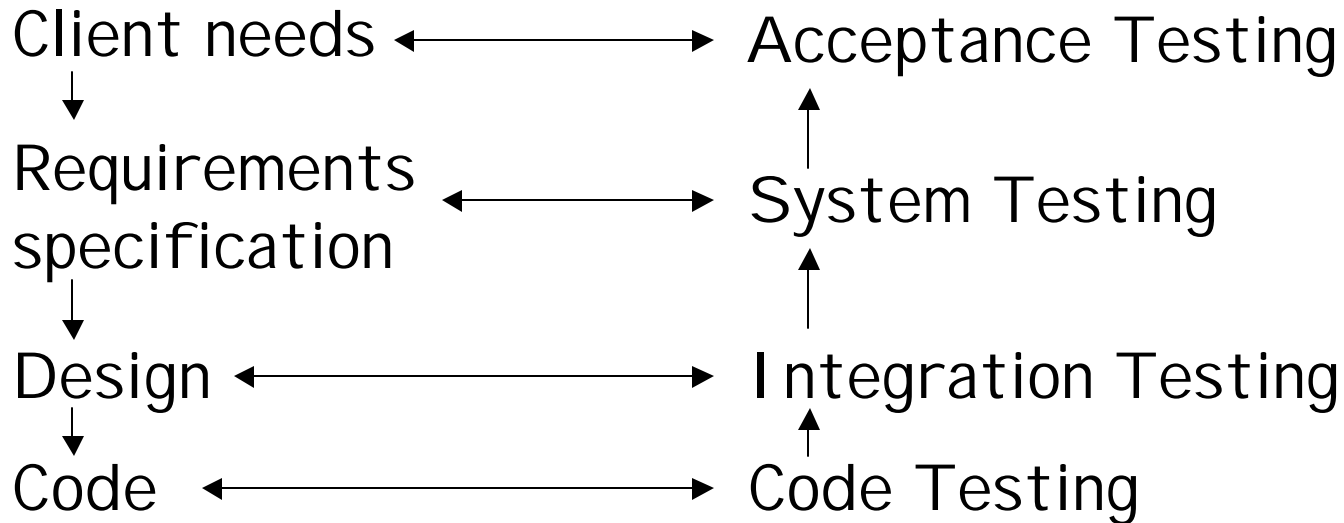
Comparing structural test coverage methods.

# Comparison of Structural and Functional Testing Methods

Structural test cases cannot recognize specified behaviors that are not implemented.

Functional test cases cannot reveal implemented program behaviors that have not been specified.

# Testing at Different Levels of Software Development



From the textbook, read sections 6.1, 6.2, 6.3.1, 6.3.2, 6.3.3, 6.3.4.1.