FPT Software – FA.HN

Unit test

Agenda

- **Duration:** *2.5 Hours*
- **Purpose:** *Introduce unit test*
- **Audience:** *Fresher*
- **Content:**
  1. Why do we need to test software?
  2. What do you know about software testing profession?
  3. Unit Test Fundamentals: Answer the question of what, why, when doing the Unit Test
  4. Unit Test design techniques

Why do we need to test Software?

A True Story

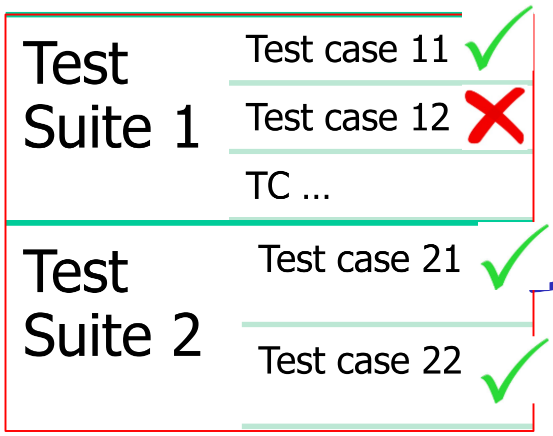
Power Steering ECU

Broken

Toyota recalls 50,000 vehicles in USA

If the testing is not careful >> has leakage >> Serious consequences

What do you know about software testing profession?



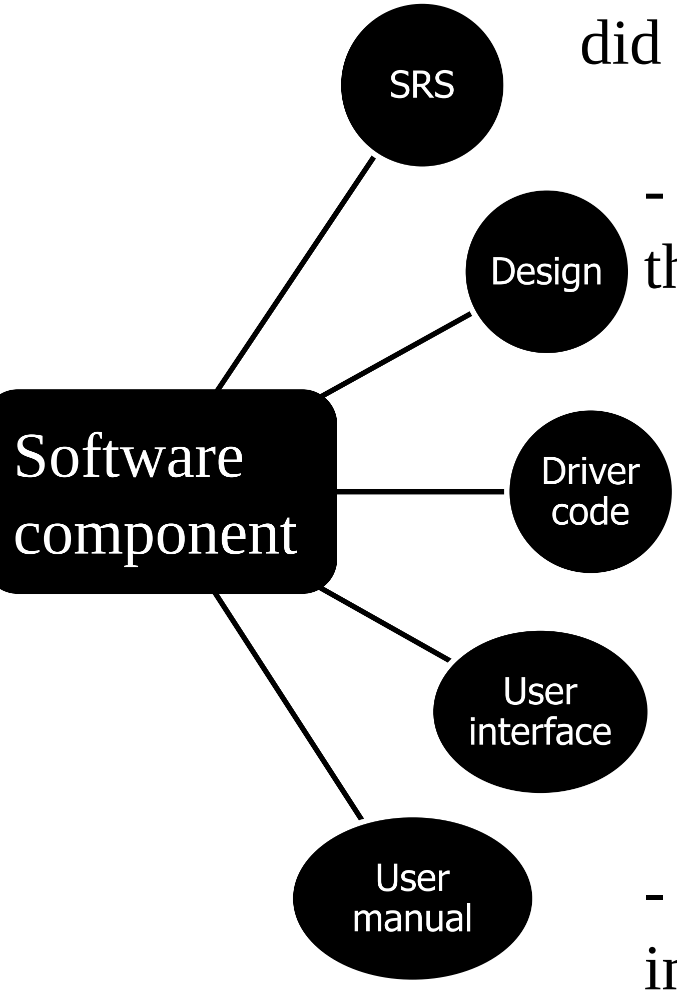
Test Suite 1	Test case 11	✓
	Test case 12	✗
	TC …	
Test Suite 2	Test case 21	✓
	Test case 22	✓

1.Check the Test Case list with available scenario

2. Pure testing only on the interface
based on requirements
(eg input information and check the monitor)

3. Mostly manual test and less automatic test

In fact: Software testing in embedded.



- SRS analysis (review) > Log ticket if SRS did not clear and difficult for testing

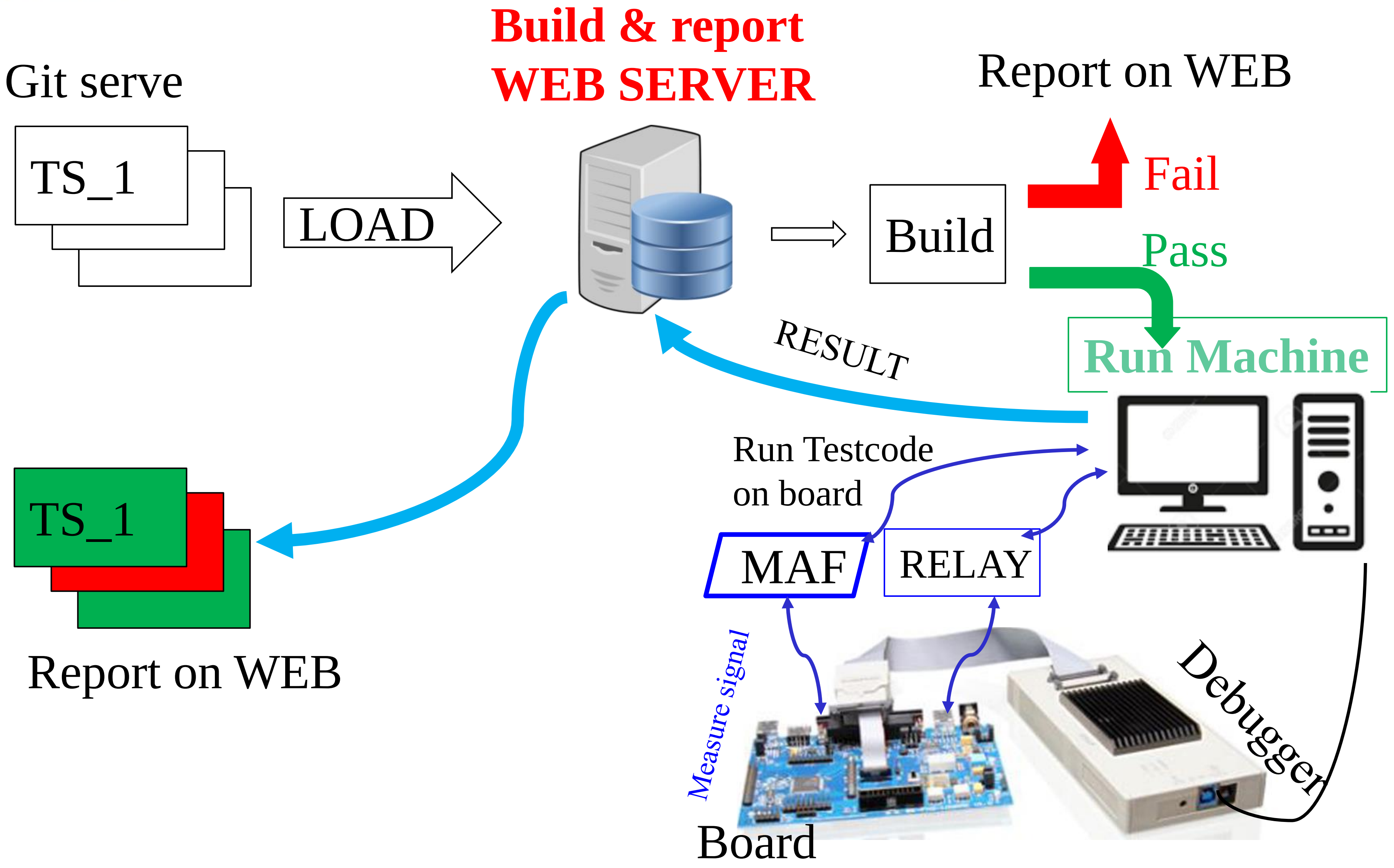
- Read Reference manual, datasheet to understand the usercase, design >> Create the Test design

- Write, Run TestCode, review driver code > find trouble > log issue

- Write, Run TestCode, review user interface configuration > find trouble > log issue

- Review, request adding known limitation, important note of module.

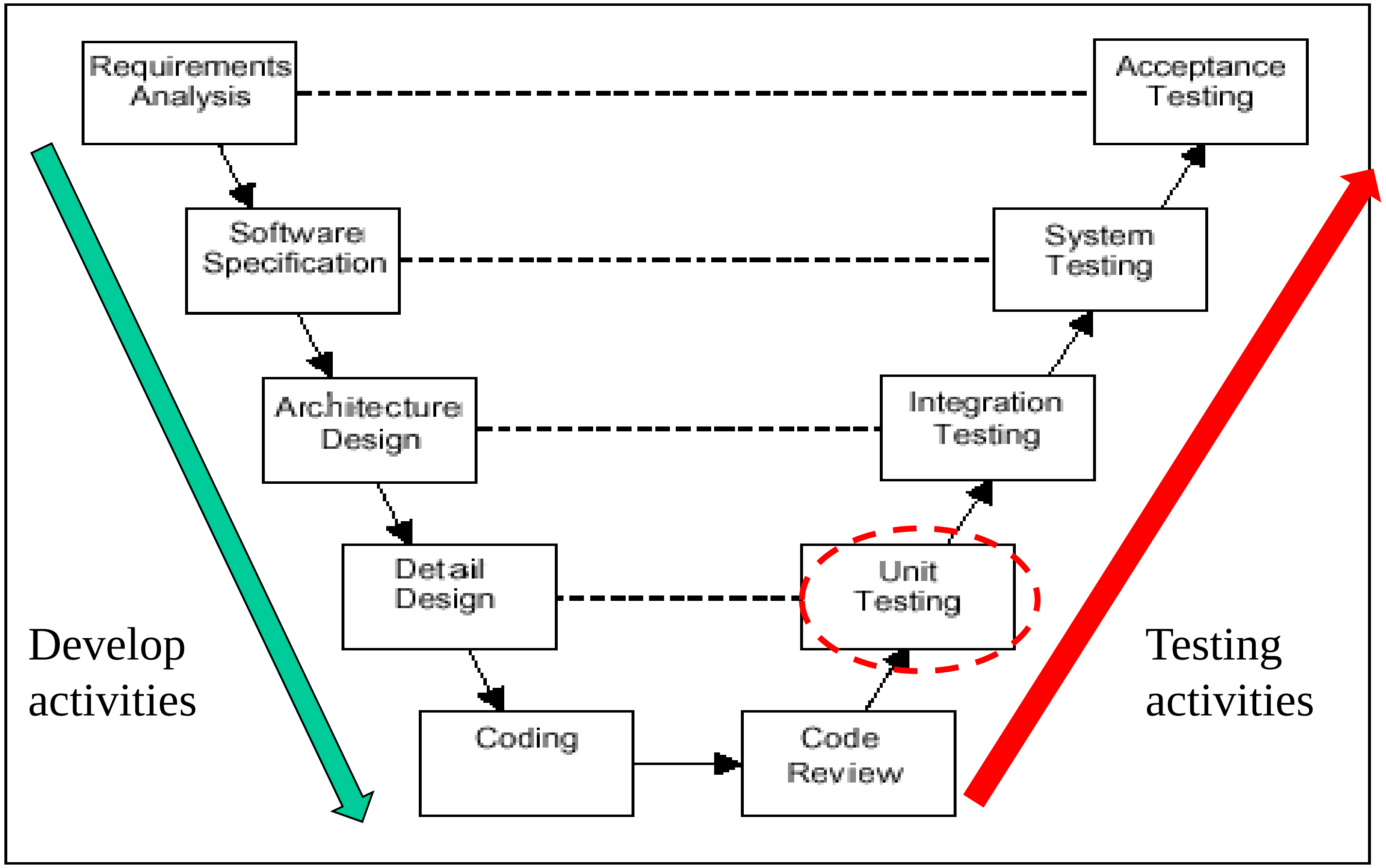
All most Test case were run manual ?



What is UnitTest ?

- “Unit testing” refers to testing software code at the smallest testable unit (method or function) and based on detail design
- Exception testing
    - Range of feasible input
- Functional testing
    - Black Box Testing - conform to specification
    - White Box testing
- Regression testing
    - Conducted after a change
    - To find new fault
- Confirmation testing
    - Test to confirm that the bugs was fixed correctly

When do Unit Test?



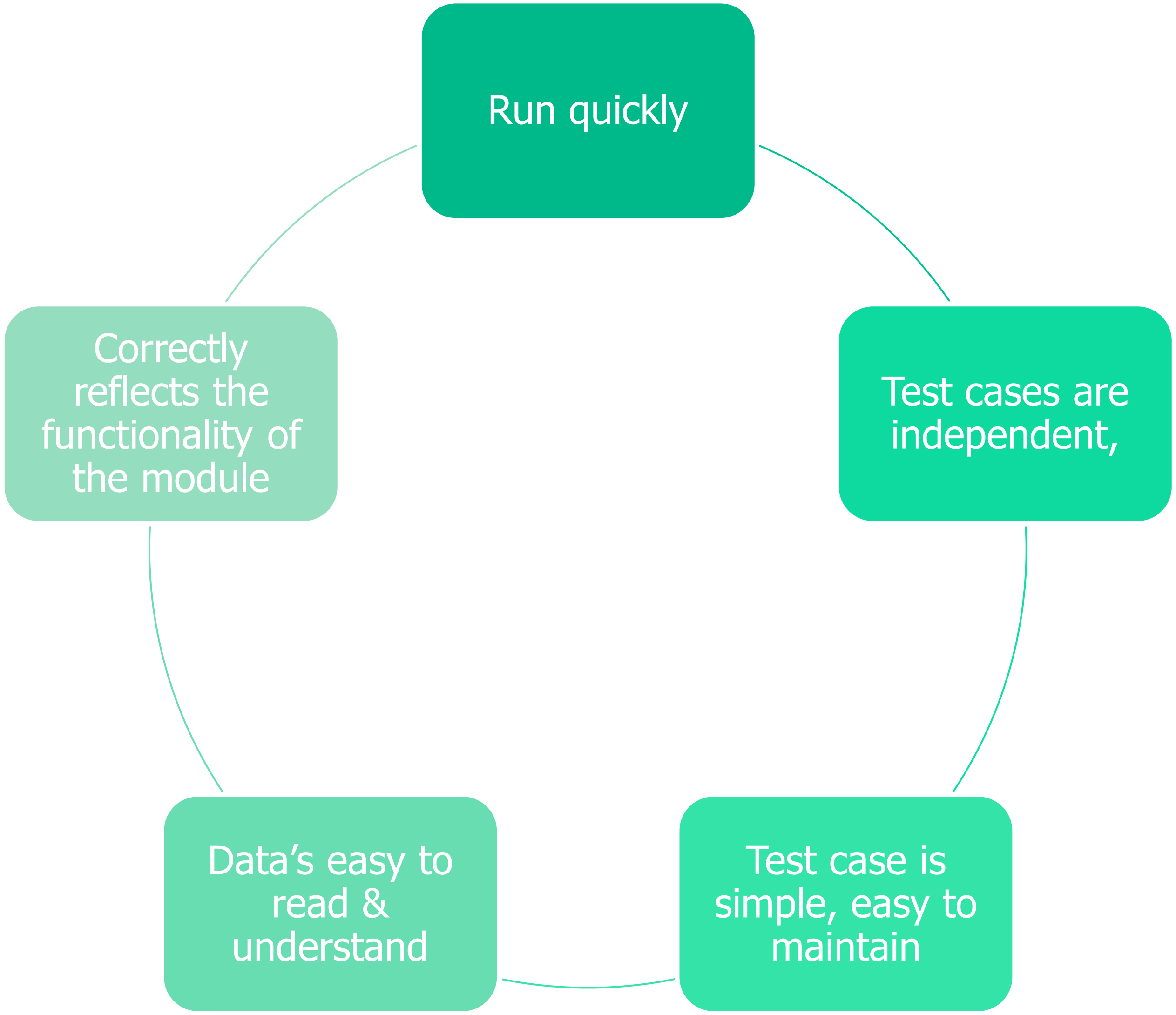
V model

Developer or tester does the Unit Test?

Why do Unit Test?

- Ensure quality of software unit
- Can test each component of project without waiting for other components complete
- Easy detect defects, issue early
- Reduce quality effort and correction cost

Good Unit Test


Run quickly
Test cases are independent,
Test case is simple, easy to maintain
Data’s easy to read & understand
Correctly reflects the functionality of the module

Categories of Test Design Techniques

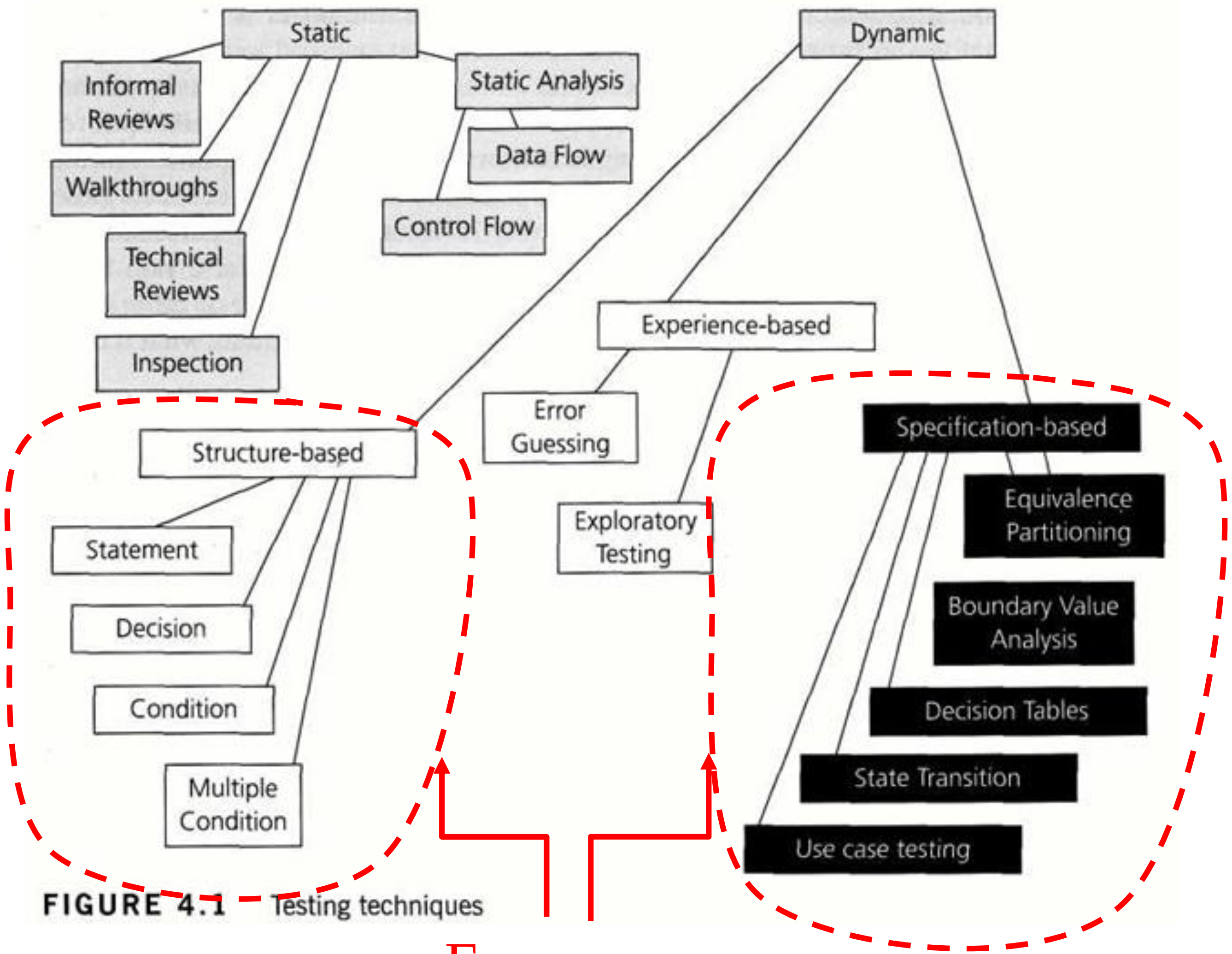
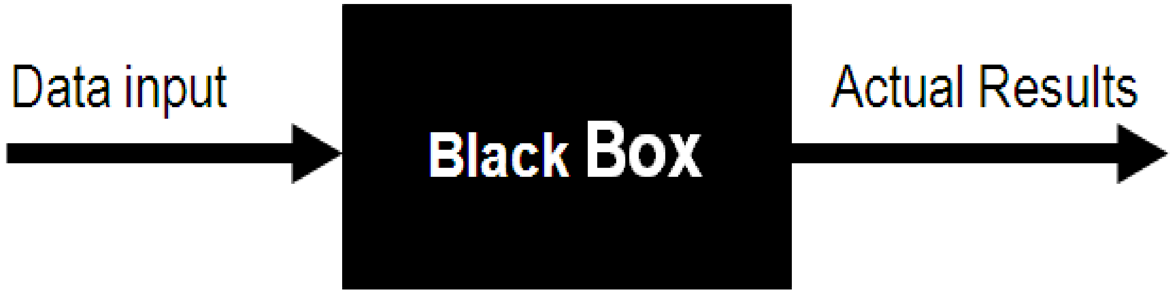


FIGURE 4.1 Testing techniques

Focus on

Black Box Testing

- Focuses on WHAT a system does not HOW it does it
- Focuses on the functional capabilities of the system
- It also know as functional testing
- Common test design techniques:
  - Equivalence partitioning
  - Boundary-value analysis
  - Combination strategy

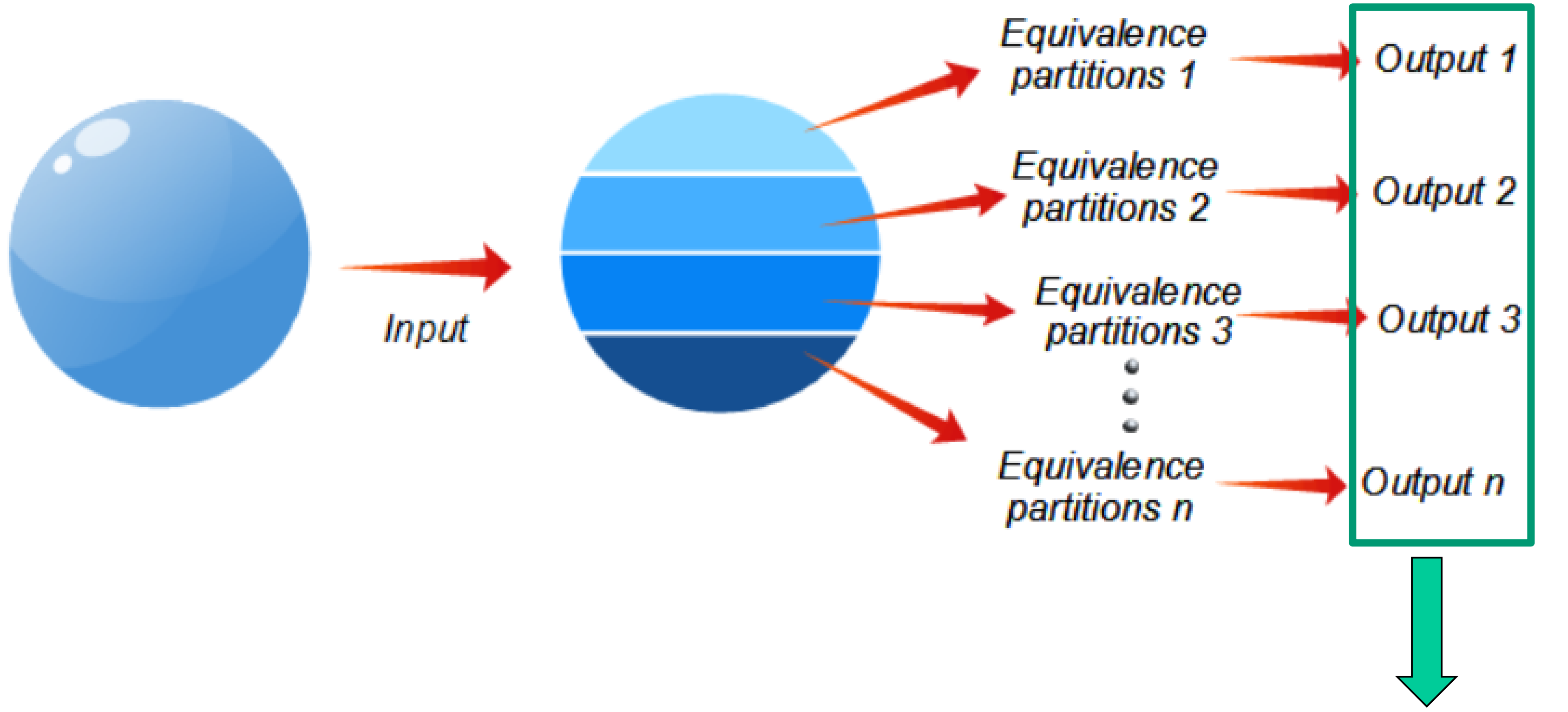


If the actual results are as expected then the test has passed

Equivalence Partitioning - Definition

- A technique in black box testing.
- It is designed to minimize the number of test cases by dividing tests in such a way that the system is expected to act the same way for all tests of each equivalence partition. Test inputs are selected from each class. Every possible input belongs to one and only one equivalence partition

Equivalence Partitioning - How To?



Number test case

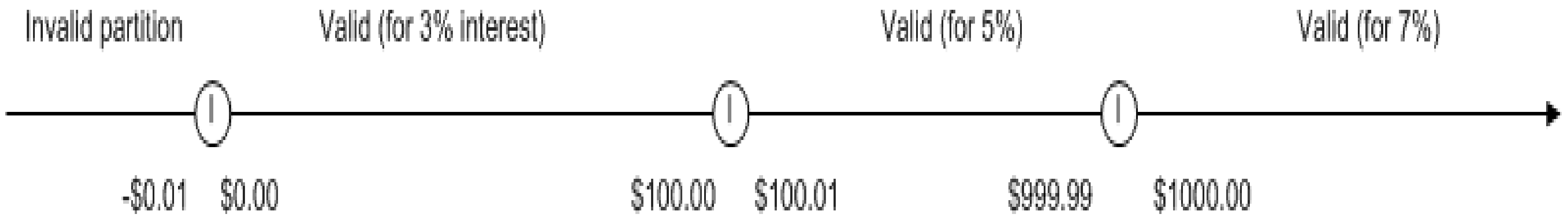
Equivalence Partitioning - How To?

- There are 2 major steps we need to do in order to use equivalence class partitioning:
  - Identify the equivalence classes of input or output. Take each input's or output's condition that is described in the specification and derive at least 2 classes for it:
    - One class that satisfies the condition – the **valid class**.
    - Second class that doesn't satisfy the condition – the **invalid class**.
  - Design test cases based on the equivalence classes.

Example

- A savings account in a bank earns a different rate of interest depending on the balance in the account

  If a balance in the range $0 up to $100 has a 3% interest rate, a balance over $100 and up to $1000 has a 5% interest rate, and balances of $1000 and over have a 7% interest rate, we would initially identify three valid equivalence partitions and one invalid partition



Case Study

In a system designed to work out the tax to be paid:

An employee has £4000 of salary tax free.
The next £1500 is taxed at 10%.
The next £28000 after that is taxed at 22%.
Any further amount is taxed at 40%.

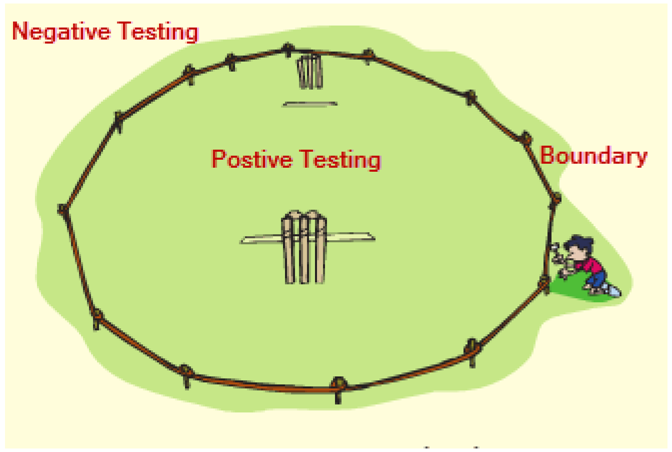
To the nearest whole pound, which of these groups of numbers fall into three DIFFERENT equivalence classes?

a) £4000; £5000; £5500
b) £32001; £34000; £36500
c) £28000; £28001; £32001
d) £4000; £4200; £5600

Time: 5’

Boundary Value Analysis - Definition

- A technique in black box testing.
- Is the process of selecting test cases (or test data) by understanding boundaries that differentiate between valid and invalid conditions. Tests are run to check the inside and outside edges of these boundaries, in addition to the actual boundary points.



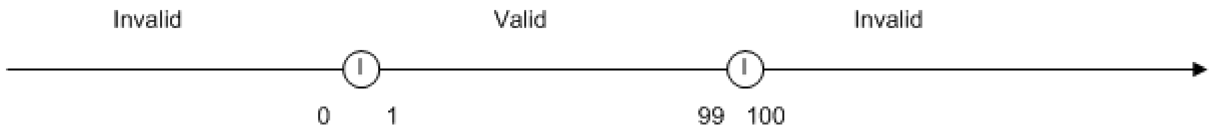
Boundary Value Analysis - How To?

- There are 2 major steps we need to do in order to use BVA:
  - Identify the boundary points (a, b)
  - Design test cases based on boundary points

Test case	Value	Expected result
1	a-1	Invalid
2	a	Valid
3	B	Valid
4	b+1	Invalid

Example

Sample: consider a printer that has an input option of



- To apply boundary value analysis, we will take the minimum and maximum (boundary) values from the valid partition (1 and 99 in this case) together with the number of copies to be made, from 1 to 99
- The first or last value respectively in each of the invalid partitions adjacent to the valid partition (0 and 100 in this case). In this example we would have three equivalence partitioning tests (one from each of the three partitions) and four boundary value tests.

Case Study

In a system designed to work out the tax to be paid:

- ✓ An employee has £4000 of salary tax free. The next £1500 is taxed at 10%
- ✓ The next £28000 is taxed at 22%
- ✓ Any further amount is taxed at 40%

To the nearest whole pound, which of these is a valid Boundary Value Analysis test case?

- A. £1500
- B. £32001
- C. £33501
- D. £28000

Time: 5’

Why Both EP and BVA?

- Because:
  - ★ Every boundary is in some partition, if you did only boundary value analysis you would also have tested every equivalence partition.
  - ★ If only testing boundaries we would probably not give the users much confidence as we are using extreme values rather than normal values

Stage Transition Testing

- State transition testing focuses on the testing of transitions from one state (e.g., open, closed) of an object (e.g., an account) to another state

- State Transition applies for finite state systems

- Use a state transition chart to identify state transitions that can occur in the real business world and state transitions that cannot occur

A state transition model

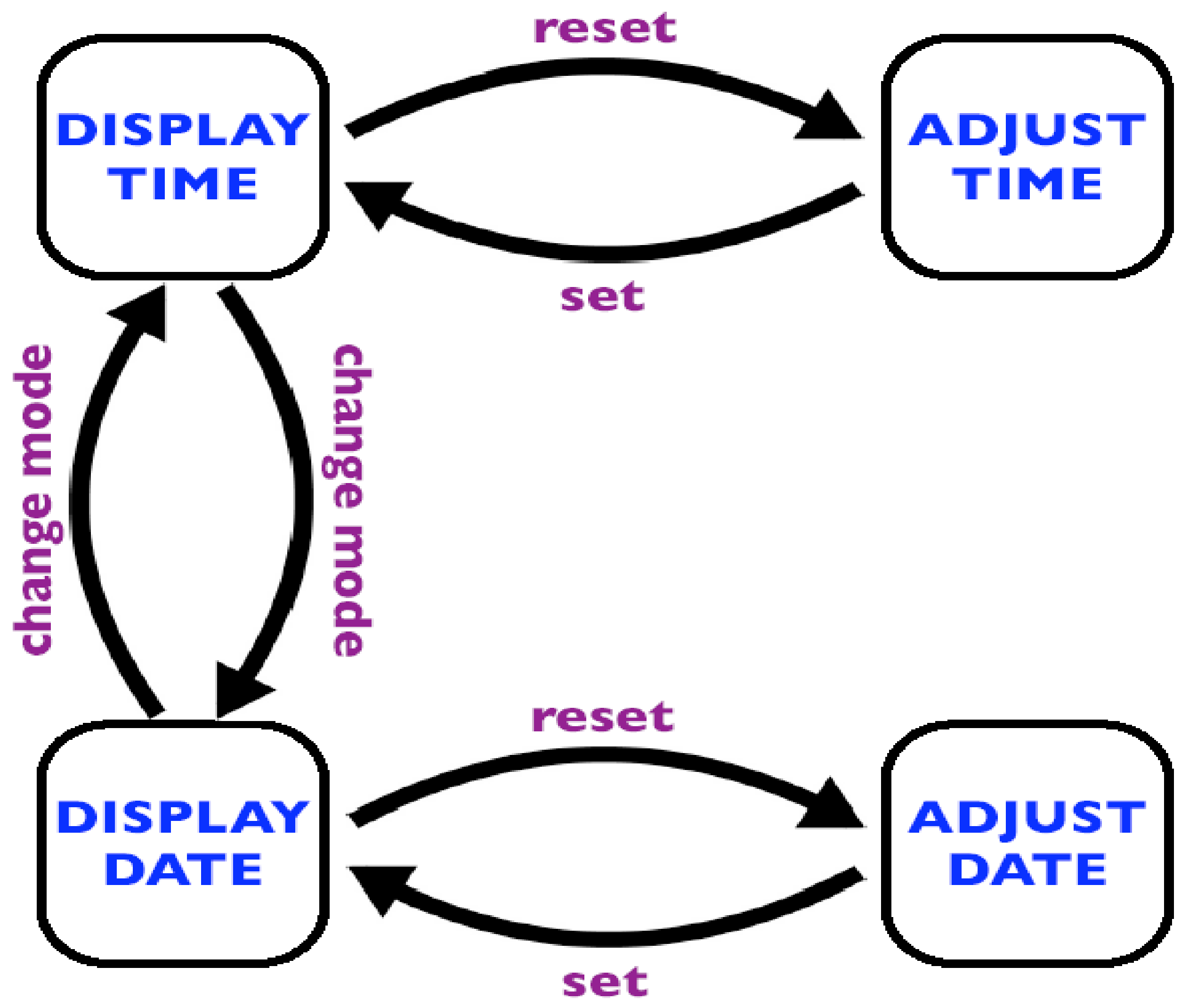
A state transition model has four basic parts

- The states that the software may occupy (open/closed or funded/insufficient funds);
- The transitions from one state to another (not all transitions are allowed);
- The events that cause a transition (closing a file or withdrawing money);
- The actions that result from a transition (an error message or being given your cash).

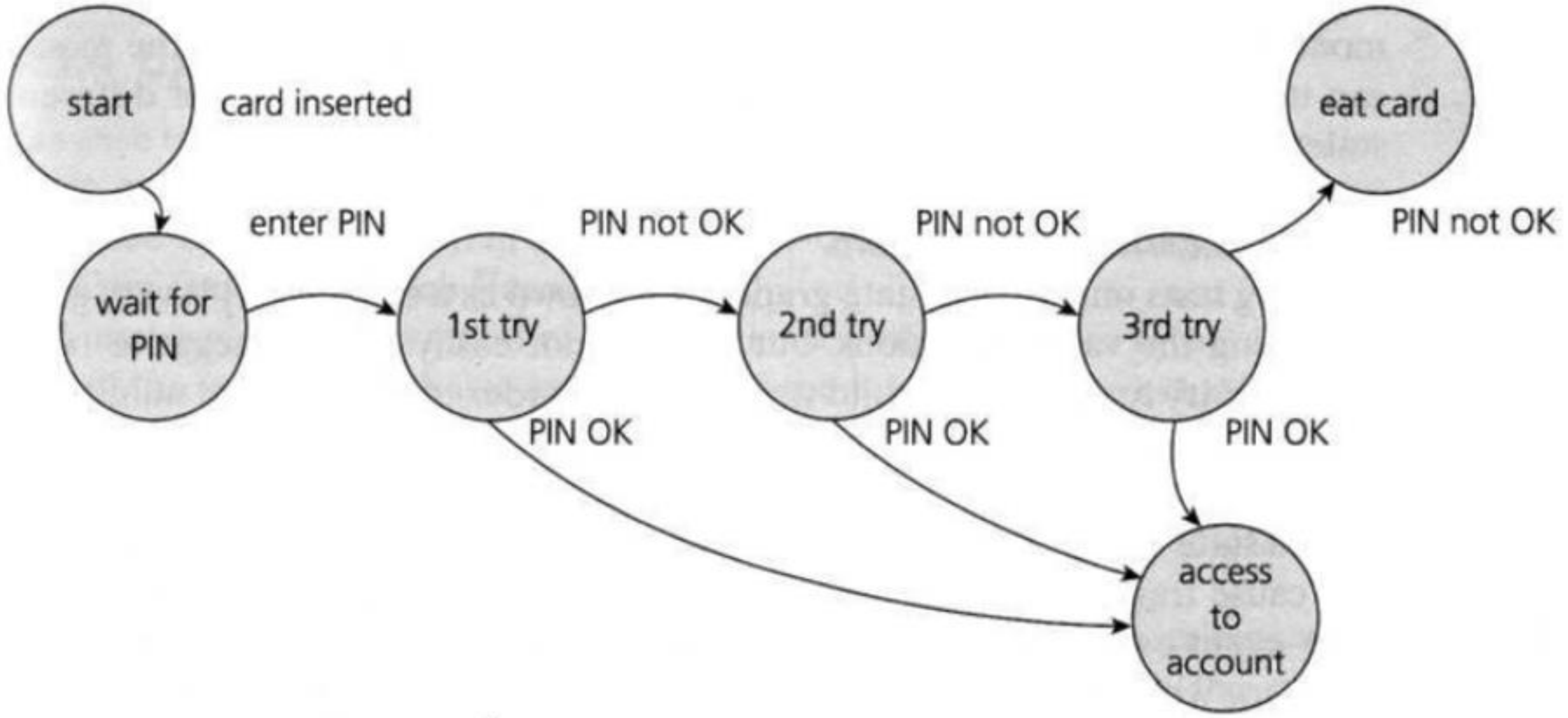
Electronic clock example

- A simple electronic clock has four modes, display time, change time, display date and change date
- The change mode button switches between display time and display date
- The reset button switches from display time to adjust time or display date to adjust date
- The set button returns from adjust time to display time or adjust date to display date

Sample: Draw a state transition diagram



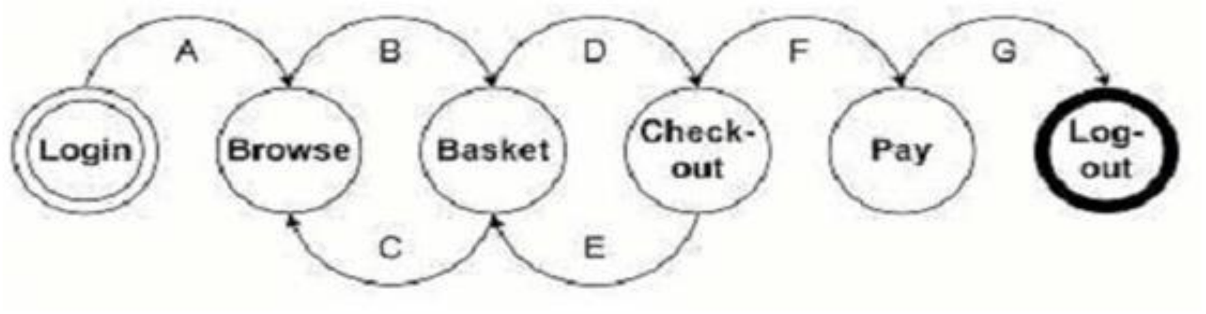
Sample: Draw a state transition diagram



Fpt Software

Case Study

Given the following state transition diagram, which of the following series of stat e transitions contains an INVALID transition which may indicate a fault in the system design?

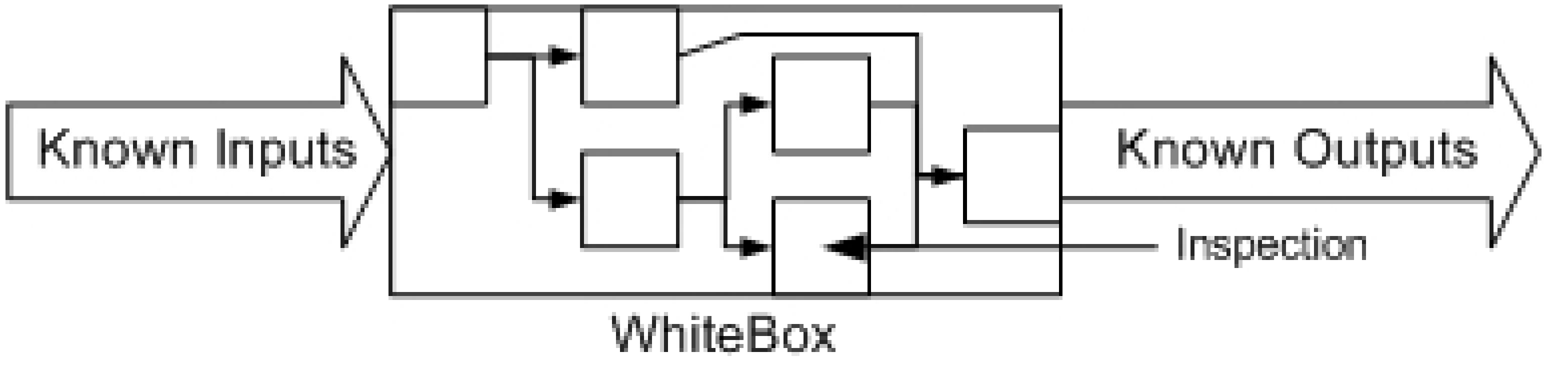


- A. Login Browse Basket Checkout Basket Checkout Pay Logout.
- B. Login Browse Basket Checkout Pay Logout.
- C. Login Browse Basket Checkout Basket Logout.
- D. Login Browse Basket Browse Basket Checkout Pay Logout

Time: 5’

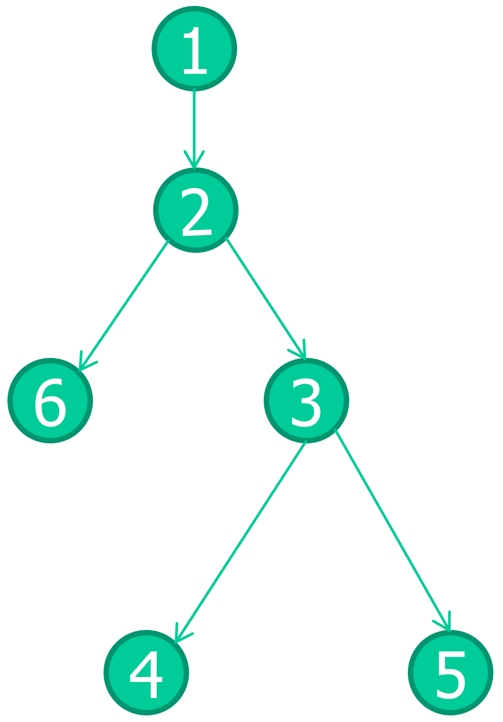
White box testing (1)

- Based on internal behavior of unit
- Code coverage based testing
- Criteria
  - Statement coverage
  - Decision coverage
  - Path coverage



White box testing (2)

```
• int Func(int a,int b)
1 {
2   if (a > 0)
3       if( b > 0)
4           return(a+b);
        else
5           return(a-b);
    else
6       return 0;
  }
```



6 statements: 1,2,3,4,5,6

4 decisions(branchs):2→6, 2→3, 3→4, 3→5

3 paths: 1 – 2 – 6, 1 – 2 – 3 – 4,1 – 2 – 3 – 5

Ex:Test case: a =1, b = 1 has SC = 4/6, PC = 1/3, DC = 2/4

Stubs and drivers

- Stubs and drivers are usually used in UT to replace missing components, software
- Stub: is called by software component testing
- Driver: call software component

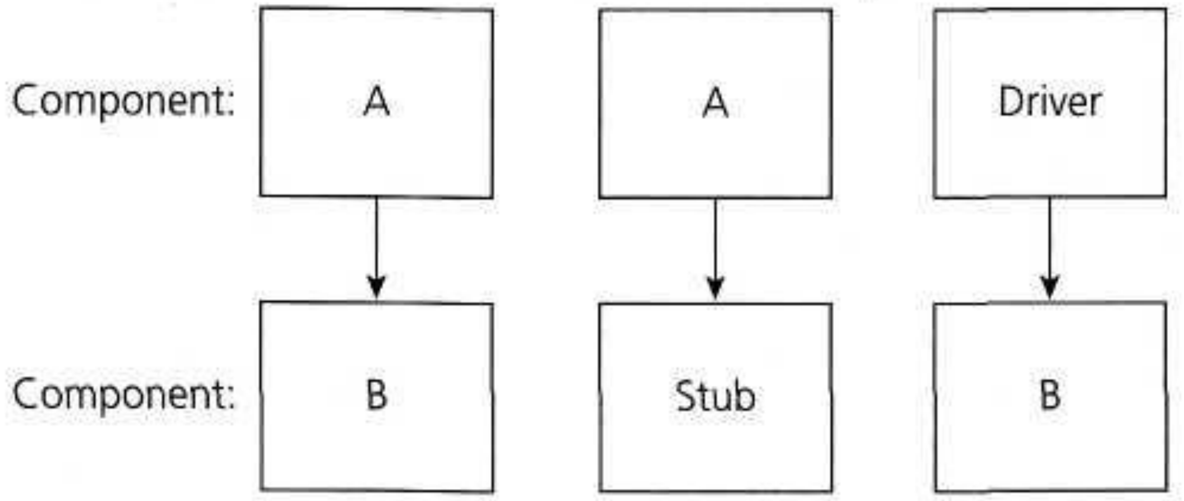


FIGURE 2.5 Stubs and drivers

QUESTIONS AND ANSWERS