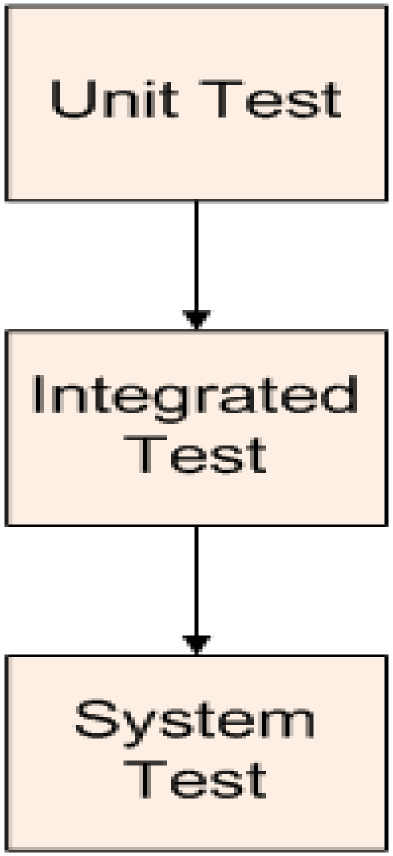
Software Testing

is critical to producing good quality software. Takes up to 40% of the total project effort in the development.

Objective in testing:

- To detect or uncover errors in the system, and consequently, correct them by evaluating the existing system against its original specifications

3 Types and Procedures of Tests



- Testing one module at a time
- Testing in the program unit(linking the modules)
- Testing the whole system

4 Unit test

- is a quality test for modules (smallest units within a program). In a test of the entire program it can be difficult to identify the cause of an error, so unit test are performed for each module as a unit. In unit tests, we perform function test and structure tests. Since modules do not function by themselves, we prepare drivers and stubs.
  - **Drivers – simulating the function of upper-level modules**
  - **Stubs – simulating the functions of lower-level modules.**

5 Integration Test

- With multiple modules linked together, we test these linked programs(load modules) in integrated tests. The main goal here is to examine the interface between modules as well as the input and output.

- Methods for integration tests include top-down tests, bottom-up tests, big bang tests and sandwich tests.

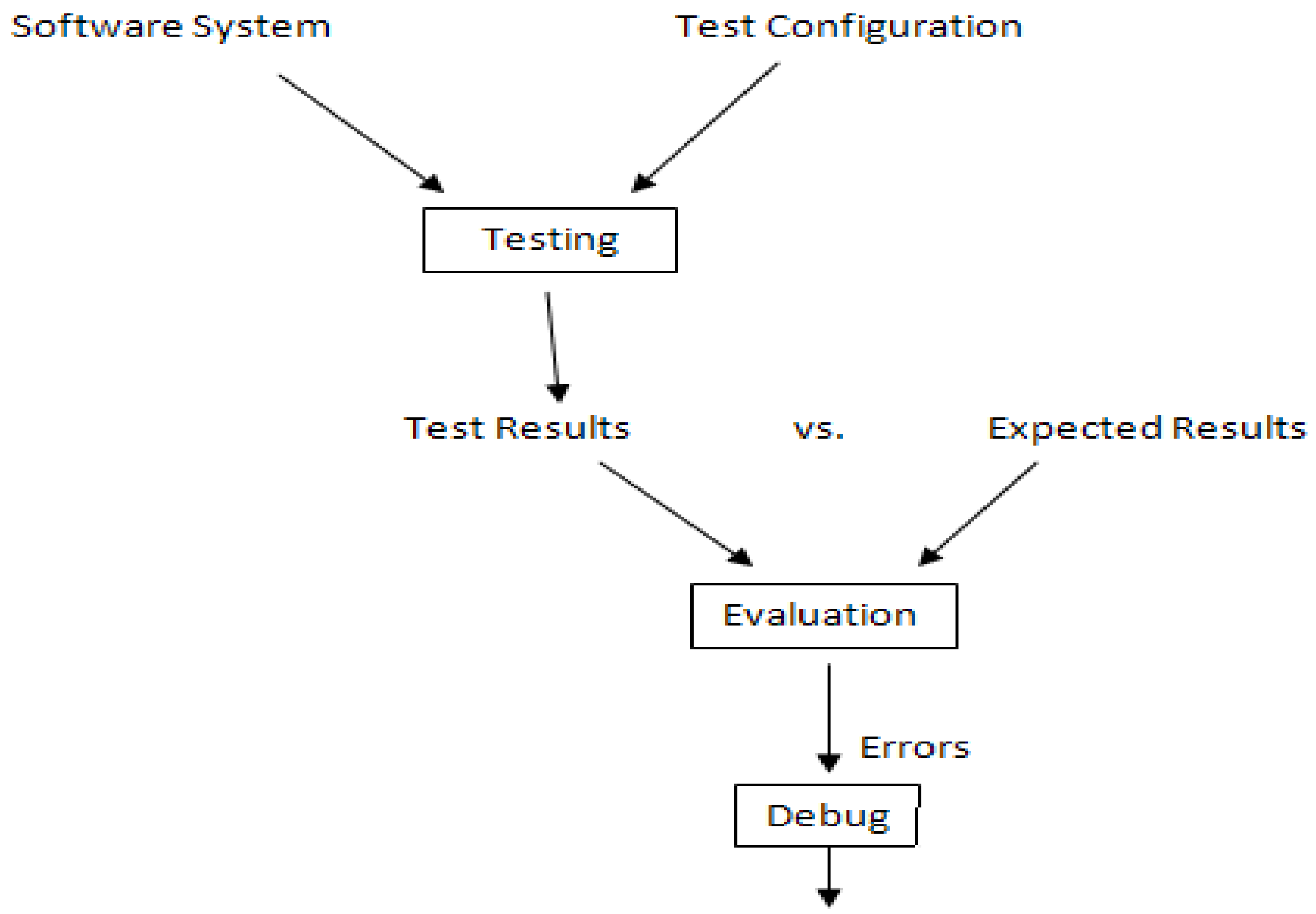
6 Methods for Integration Testing

- **Function Test – it is a validation test, based on module specification, to verify that all functions that the module is supposed to have are satisfied.**
- **Structure Test – It is a validation test, based on module specifications and source program, to verify that the logic of the module is sound.**
- **Big-bang Test – it is a test wherein all the modules that have completed the unit tests are linked all at once and tested. If the program is small scale, this could reduce the number of testing procedures, however, if an error occurs, it is difficult to identify where the error has occurred.**
- **Sandwich Test – It is a test where lower-level modules are tested bottom-up and higher-level modules are tested top-down. This is the most realistic type of testing.**

"Software testing shows the presence of bugs, but not their absence."

…thus while software testing has its limitations, software verification addresses these limitations.

General Procedure for Software Testing



- **After errors are detected and corrected, we must address two main issues:**
  - ***What is the assurance that the debugging really fixed the error?***
  - ***What is the assurance that the debugging did not introduce more errors and possibly more complicated ones?***

- **What if software testing does not detect errors in the system?**
  - **The software has no errors (Fairley, 1985).** ***The number of errors detected is proportional to the number of remaining errors in the system.***
  - **The test are inadequate to uncover the errors (Pressman, 1992)** ***Point of view : we need to re-evaluate the design of our test configurations and make sure that they are adequate to catch the errors.***

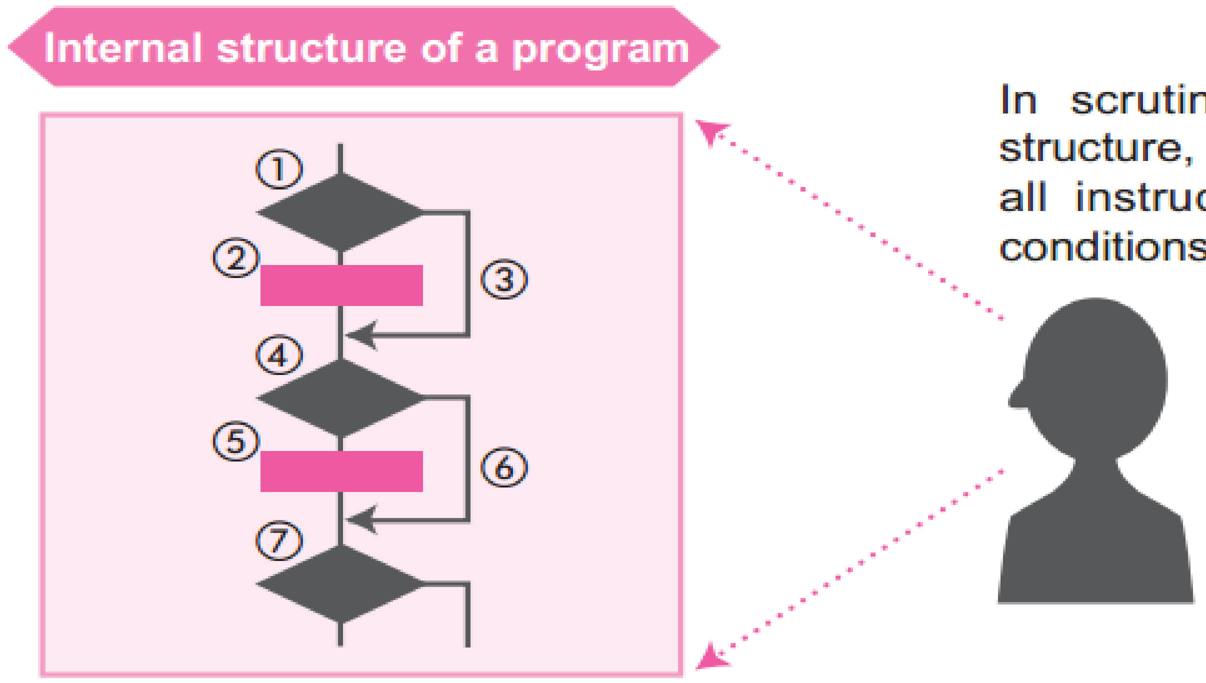
Different types of errors

Lamb (1988)	Fairley(1985)
➢logical ➢overload ➢timing ➢performance throughput ➢hardware/operating system errors	➢requirement ➢design ➢implementation

12 Software Testing Techniques

- **White box testing –** performs tests on internal workings of the software by close examination on the procedural level, using the internal logic of the procedural design of the software.
- **Black box testing –** tests the performance of specified functions on the interface of the software on the operational level

White Box Testing



In scrutinizing the internal structure, test cases to cover all instructions and branch conditions are considered.

① → ② → ④ → ⑤ → ⑦ ············ Statement coverage

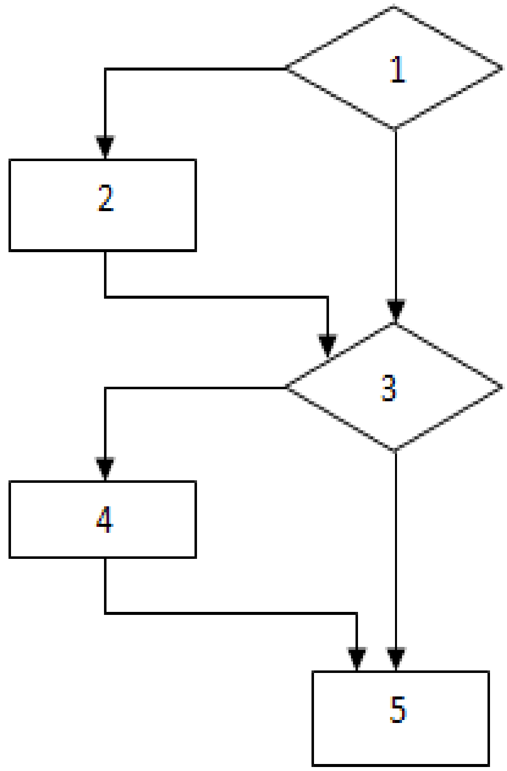
① → ③ → ④ → ⑤ → ⑦
① → ② → ④ → ⑤ → ⑦
① → ③ → ④ → ⑥ → ⑦
① → ② → ④ → ⑥ → ⑦

········· Condition coverage (All test cases are tested)

Test case design methods for white box testing

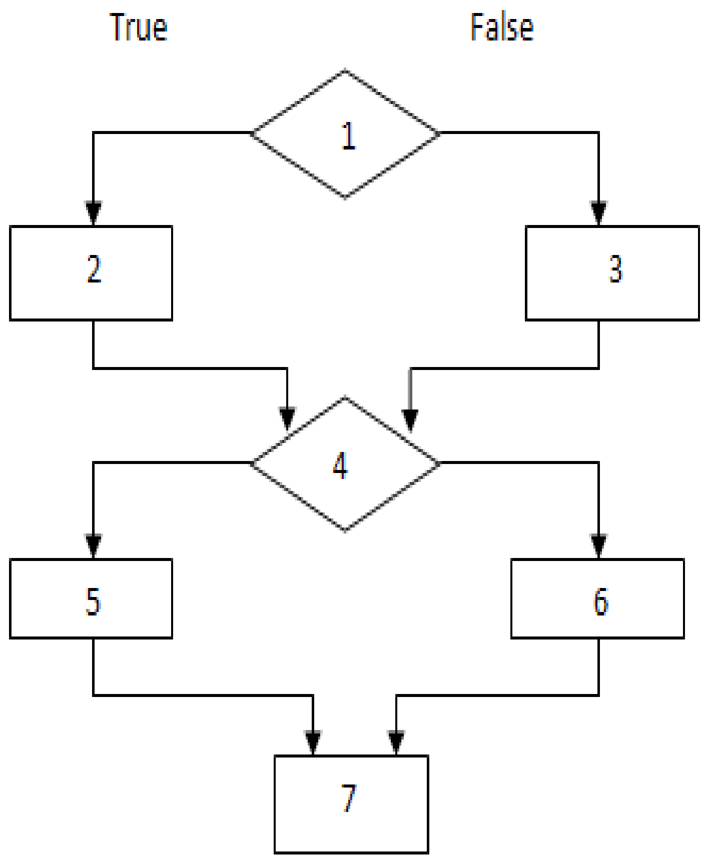
- Statement testing - all statements in the program are executed at least once.
- Condition testing – for every condition, all branches are chosen at least once.
- Path testing – all distinct paths are executed at least once.

Statement Testing



- **Logical complexity value – ** measures the upper bound on the number of test cases needed to be designed and executed to guarantee that all statements in the program are executed at least once.
- The basis set contains only one path 1,2,3,4,5

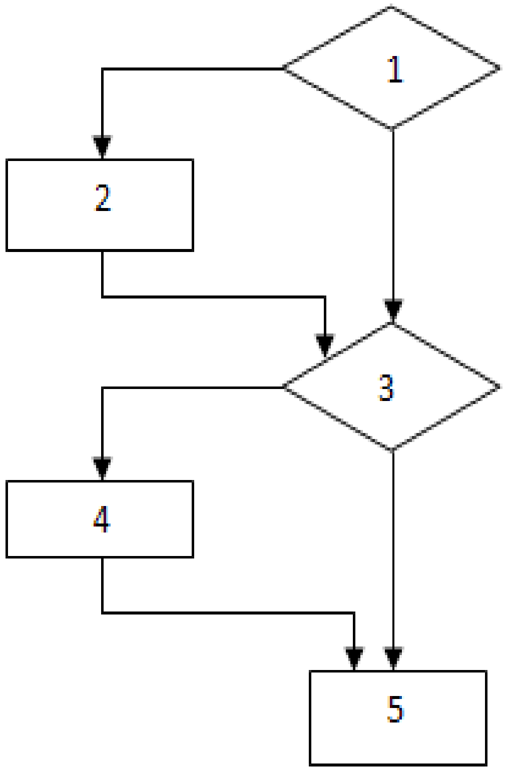
Basis set may not be unique:



- **Basis set 1:**
  - **Path 1: 1,2,4,5,7**
  - **Path 2: 1,3,4,6,7**
- **Basis set 2:**
  - Path 1: 1,2,4,6,7
  - Path 2: 1,3,4,5,7

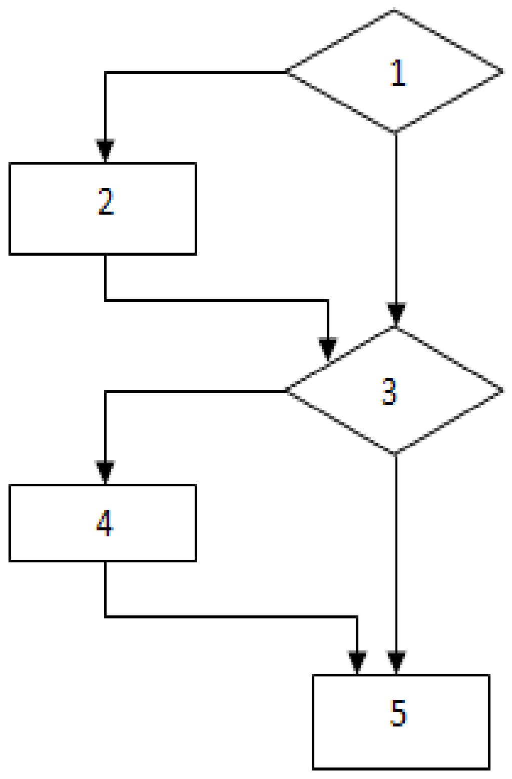
Logical complexity value = 2

Using Condition Testing:

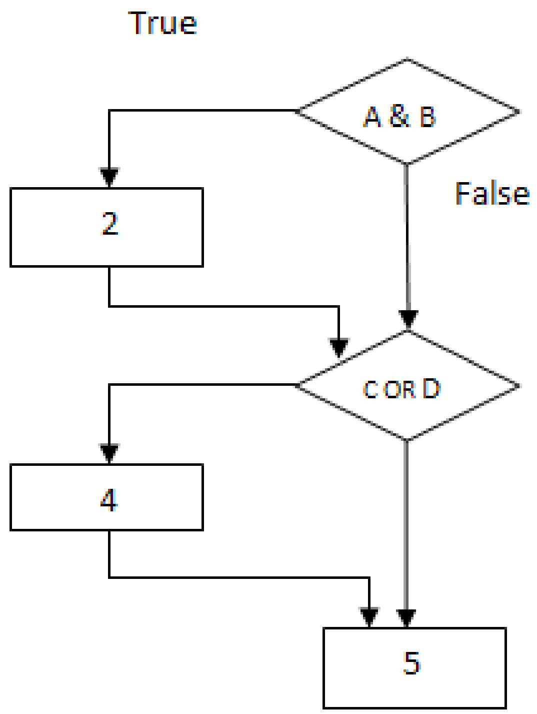


- **Condition testing** is a test case design method that focuses on testing each condition in the program.
- The approach is similar to statement testing, but instead uses the test criterion that for every condition all branches are executed at least once.
- In the example, test cases for both statement and condition testing are the same

Using Condition testing:

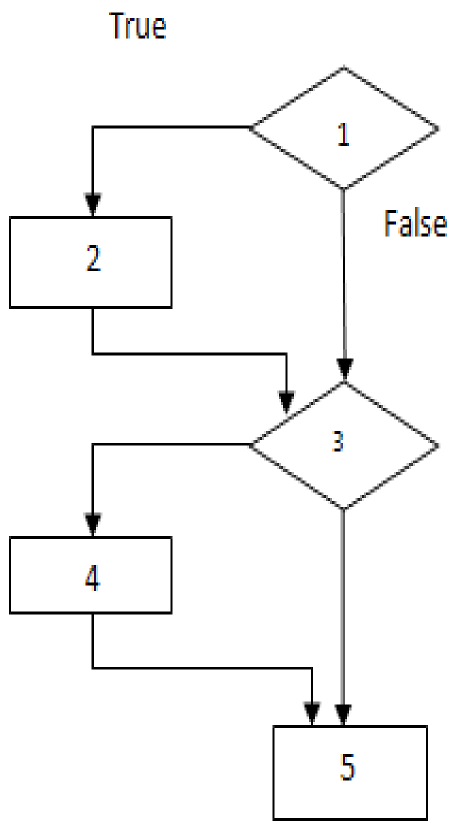


- Path 1 : 1,2,3,4,5
- Path 2 : 1,3,5



- Path 1 : A & B,2,C OR D, 4, 5
- Path 2 : A & B, C OR D, 5
    - *The sequence of the execution of operands in conditions affects the performance of the tests.*

20 PATH TESTING



- In *Path Testing*, test execution paths are derived using the following criterion: all distinct paths are executed at least once
  - Path 1: 1,2,3,4,5
  - Path 2: 1,2,3,5
  - Path 3: 1,3,4,5
  - Path 4: 1,3,5

Black box testing

- tests the performance of specified functions on the interface of the software on the operational level without bothering about the internal structure of the program. Three steps followed in black box testing:
  - establishing the input domain
  - partition these input domains into classes from which test cases can be derived.
  - define what these valid and invalid inputs will be (Boundary value analysis)

Example 1: Functional test

A payroll program will calculate the weekly gross pay given an input of the number of hours worked and current wages. A worker may not work more than 80 hours per week, and the maximum wage is $50.00 per hour. Construct functional tests.

Functional tests should include tests of normal pay and overtime and tests of the error conditions.

Hours	Wages	Expected Output
30	40.00	1200.00
60	50.00	3500.00 (assume overtime)
81	50.00	Invalid hours
20	60.00	Invalid wages

Identifying the conditions and constructing the test matrix:

Condition						
0<hours<=40	T	F	F	F	T	F
40 < hours < = 80	F	T	F	T	F	F
0< wages <= 50	T	T	T	F	F	F
Hours	30	50	90	50	30	-5
Wages	50	30	50	60	70	-5
Expected output	1500	1650	Error	Error	Error	Error

Example2: Functional test

A program accepts two times (in 12-hour format) and outputs the elapsed number of minutes. Construct functional tests.

The functional tests should include tests of less than 1 hour, more than 1 hour, one more than 12 hours, one that requires a carry of hours, and one with the times reversed.

Start Time	Stop Time	Expected Elapsed Time
10:00 am	10:40 am	0:40
9:57 pm	11:40 pm	1:43
3:00 am	9:15 pm	18:15
1:50 am	3:40 am	1:50
3:00 am	7:24 am	4:24
5:00 pm	4:00 am	Error

Combination Factors

White box testing method uncovers the errors in the logical level of the program, there is also the danger of paying too much attention into the code's internal structure. On the other hand, while the black box testing deals with the functional requirements of the software through the interface, it may not be sufficient to detect particular errors.

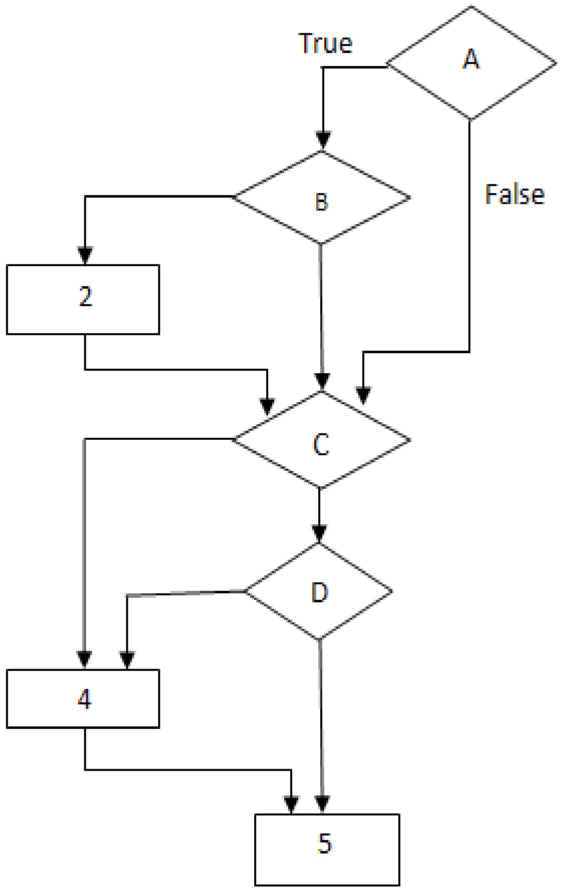
- **Nature of the project**
- **Complexity of the system**
- **Nature of the input and output data**
- **Amount of computation and logic involved**

26 Summary:

- Software testing is essential to the delivery of good quality software systems.
- It may ensure a certain degree of confidence on the quality of the system, it does not eliminate the presence of errors in the software.
- Can be performed by deriving test cases using the white box and black box test design methods.

“ Testing never ends, it just gets transferred from the developer to the client. Every time the client uses the software, a test is being conducted. ”

Seatwork:



- Given the structural flow of a program, identify the number of execution paths and the execution paths and the values for the operands using statement testing, condition testing and path testing.

Design Method	Number of Execution Paths	Execution Paths	Values of			
			A	B	C	D
Statement Testing						
Condition Testing						
Path Testing						