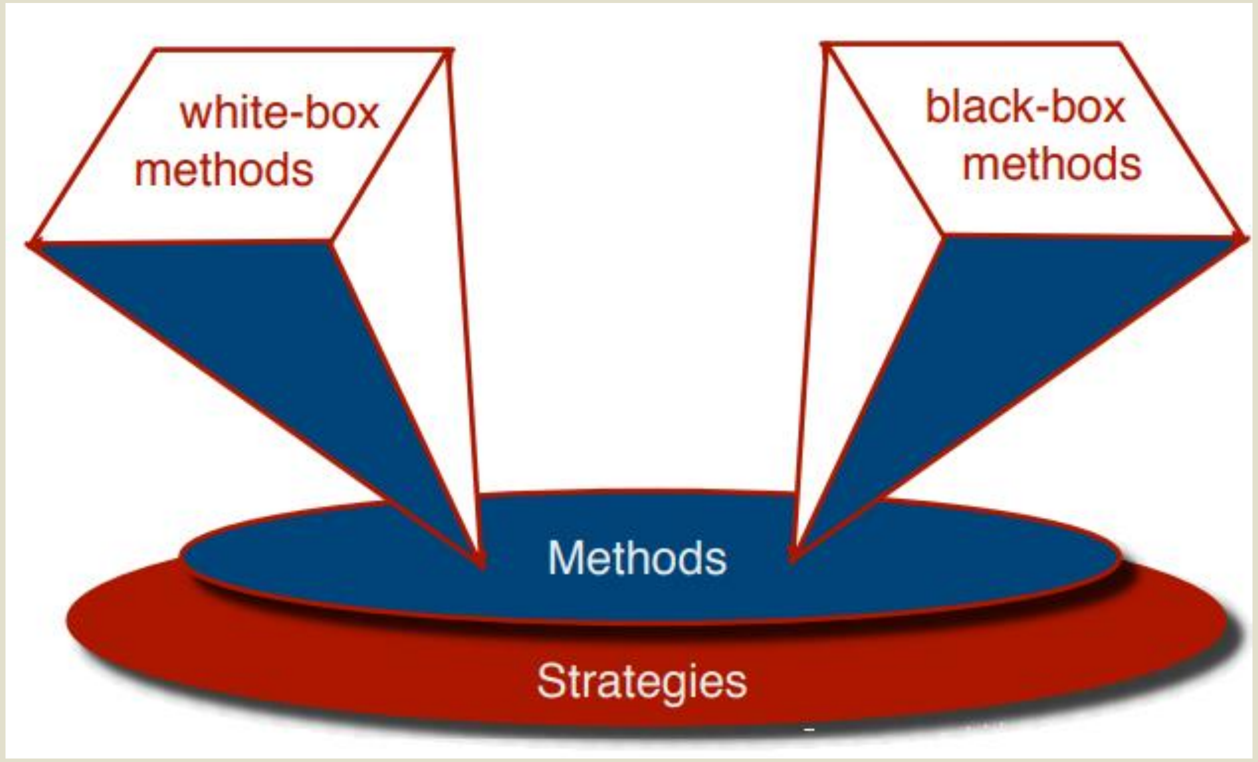
SOFTWARE TESTING FUNDAMENTALS

LECTURE # 32

Lecture by Engr.Sidra

SOFTWARE TESTING TECHNIQUES



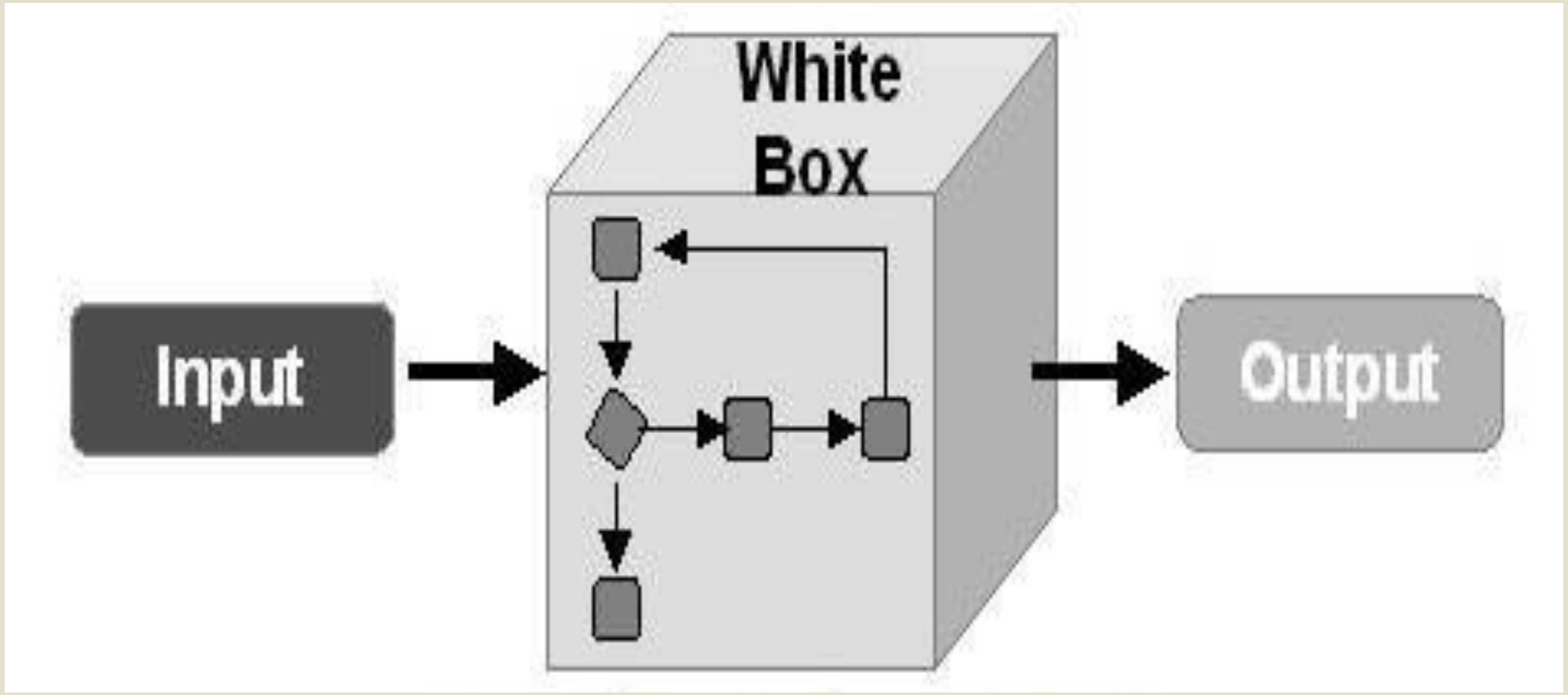
BLACK BOX TESTING V/S WHITE BOX TESTING

Black box Testing	White box Testing
It is a way of software testing in which the internal structure or the program or the code is hidden and nothing is known about it.	It is a way of testing the software in which the tester has knowledge about the internal structure or the code or the program of the software.
It is mostly done by software testers.	It is mostly done by software developers.
No knowledge of implementation is needed.	Knowledge of implementation is required.
It can be referred as outer or external software testing.	It is the inner or the internal software testing.
It is functional test of the software.	It is structural test of the software.
This testing can be initiated on the basis of requirement specifications document.	This type of testing of software is started after detail design document.

BLACK BOX V/S WHITE BOX TESTING

Black box Testing	White box Testing
No knowledge of programming is required.	It is mandatory to have knowledge of programming.
It is the behaviour testing of the software.	It is the logic testing of the software.
It is applicable to the higher levels of testing of software.	It is generally applicable to the lower levels of software testing.
It is also called closed testing.	It is also called as clear box testing.
It is least time consuming.	It is most time consuming.
It is not suitable or preferred for algorithm testing.	It is suitable for algorithm testing.
Example: search something on google by using keywords	**Example:** by input to check and verify loops
Types of Black Box Testing: Functional Testing, Non-functional testing, Regression Testing	**Types of White Box Testing:** Path Testing, Loop Testing, Condition testing

WHITE BOX TESTING



BASIS PATH TESTING

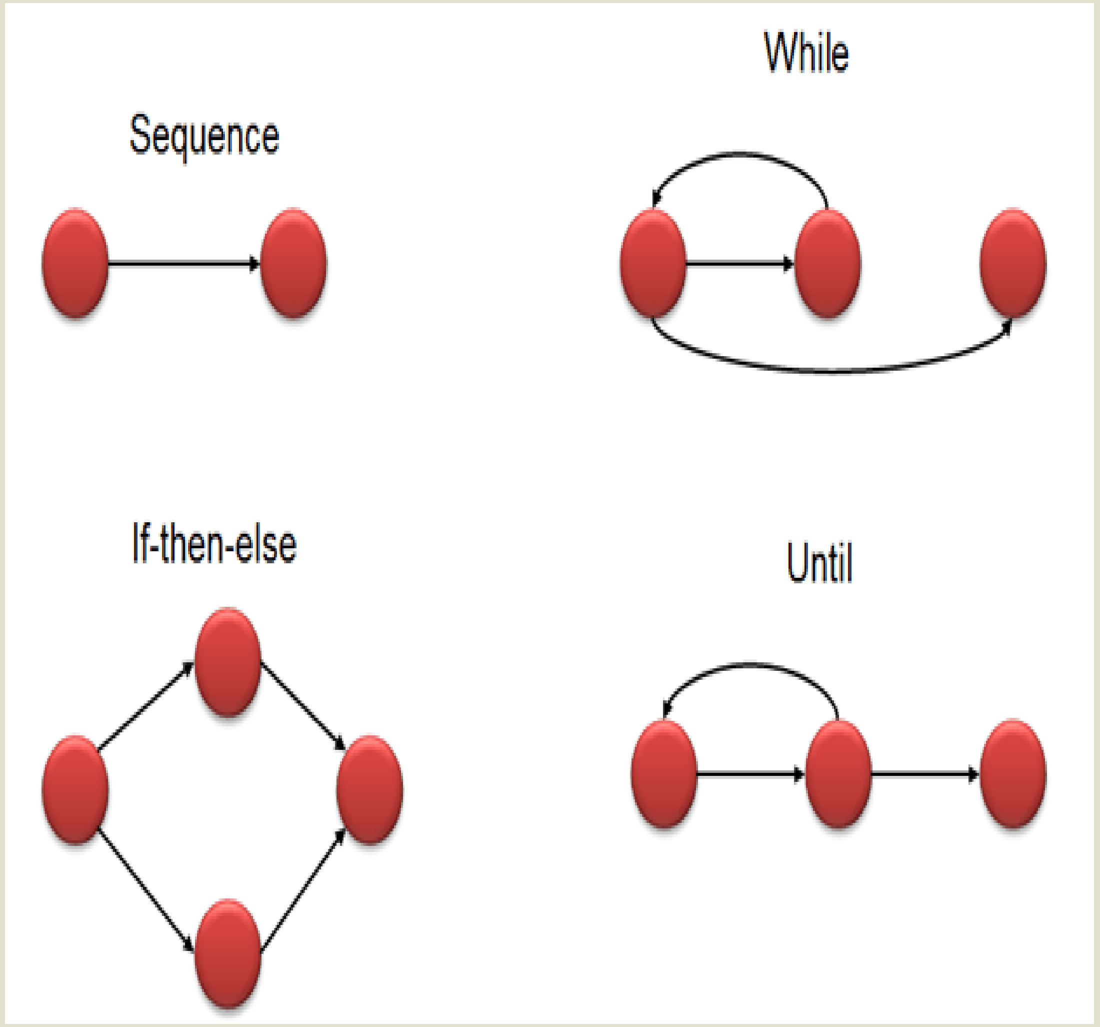
- It is based on a White Box Testing method, that defines test cases based on the flows or logical path that can be taken through the program. In software engineering, Basis path testing involves execution of all possible blocks in a program and achieves maximum path coverage with the least number of test cases.
- The objective behind basis path in software testing is that it defines the number of independent paths, thus the number of test cases needed can be defined explicitly (maximizes the coverage of each test case).

STEPS FOR BASIS PATH TESTING

- The basic steps involved in basis path testing include
  - Draw a control-flow graph (to determine different program paths)
  - Calculate Cyclomatic complexity (metrics to determine the number of independent paths)
  - Find a basis set of paths
  - Generate test cases to exercise each path

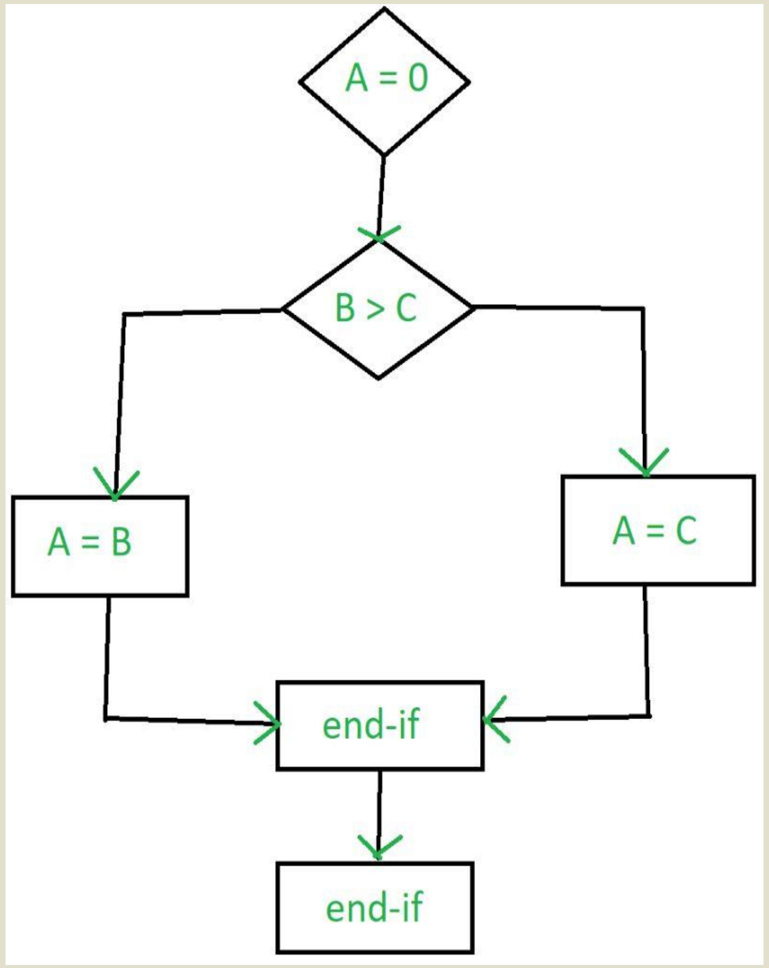
CONTROL-FLOW GRAPH

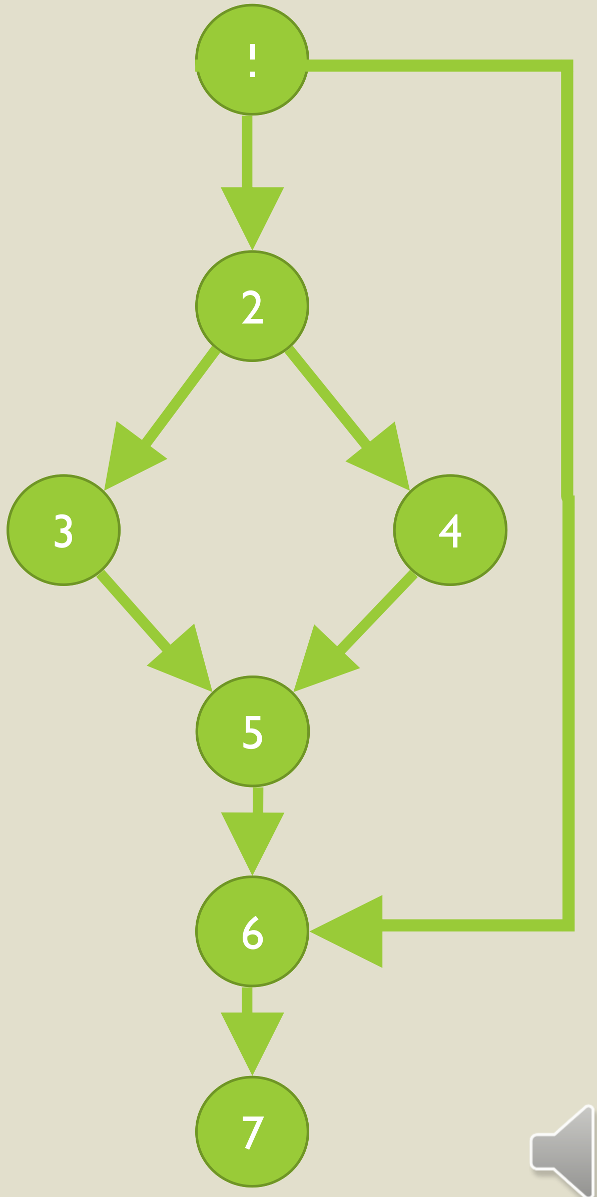
- A **Control Flow Graph (CFG)** is the graphical representation of control flow or computation during the execution of programs or applications.
- Control flow graphs are mostly used in static analysis as well as compiler applications, as they can accurately represent the flow inside of a program unit.



EXAMPLE OF FLOW GRAPH

```
1. if A = 10
     then
2.          if B > C
3.               A = B
4.    else A = C
5.          end if
6. end if
7. print A, B, C
```





CYCLOMATIC COMPLEXITY

- It is used to measure the complexity of a program.
- It is a quantitative measure of independent paths in the source code of the program.
- Independent path is defined as a path that has at least one edge which has not been traversed before in any other paths.
- Cyclomatic complexity can be calculated with respect to functions, modules, methods or classes within a program.

- $V(G) = E - N + 2$ where E - Number of edges and N - Number of Nodes
- In the above example E=8 and N = 7 so the cyclomatic complexity will be:

  $V(G) = 8 – 7 + 2$

  $V(G) = 3$

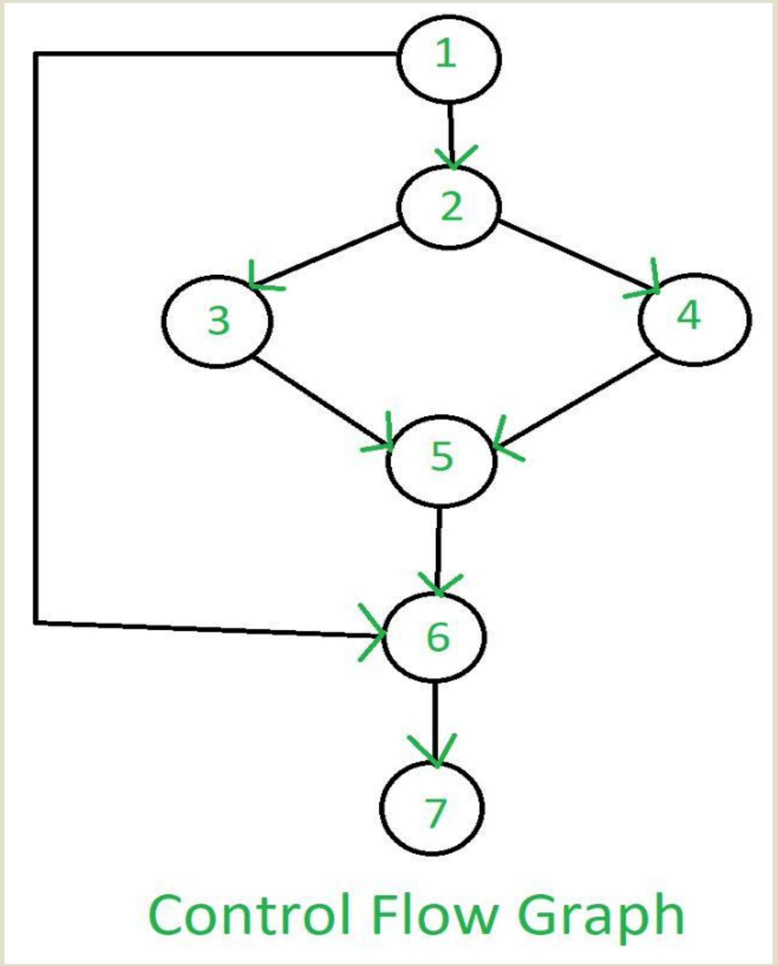
FIND THE BASIS PATHS

- Basis paths are independent paths.
- We have V(G) = 3
- So the paths will be

Path 1: 1-2-3-5-6-7

Path 2: 1-2-4-5-6-7

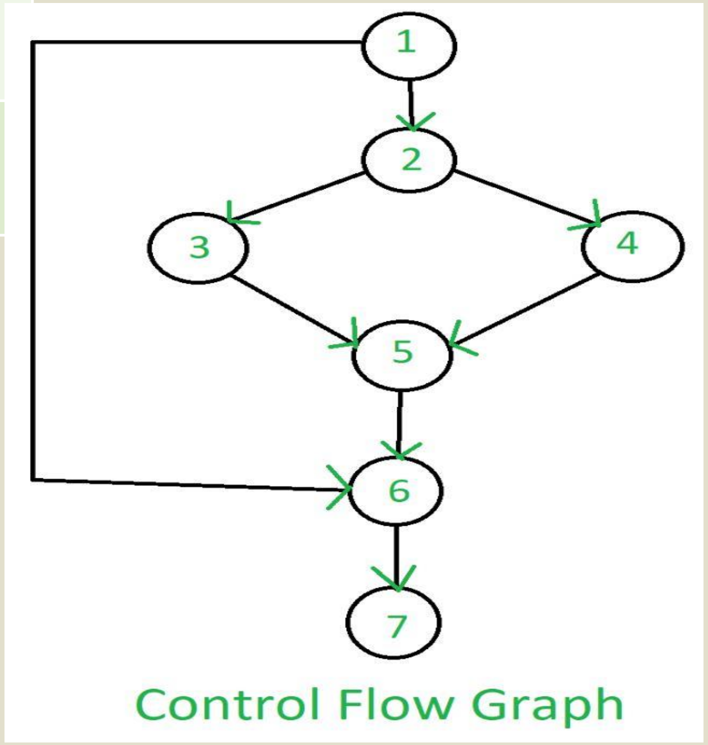
Path 3: 1-6-7



GENERATE TEST CASES

Test case ID	Paths	Test Data	Expected Result	Actual Result	Test Status
1	Path 1: 1-2-3-5-6-7	A = 10, B = 8 , C = 6	8, 8, 6		
2	Path 2: 1-2-4-5-6-7	A = 10, B = 6, C = 8	8, 6, 8		
3	Path 3: 1-6-7	A = 11, B = 6, C = 8	11, 6, 8		

```
if A = 10 then
   if B > C
      A = B
   else A = C
   end if
end if
print A, B, C
```


$$\begin{tikzcd}
 & 1 \arrow[d] \arrow[ddd, bend right=60] & \\
 & 2 \arrow[dl] \arrow[dr] & \\
3 \arrow[dr] & & 4 \arrow[dl] \\
 & 5 \arrow[d] & \\
 & 6 \arrow[d] & \\
 & 7 &
\end{tikzcd}$$

Control Flow Graph