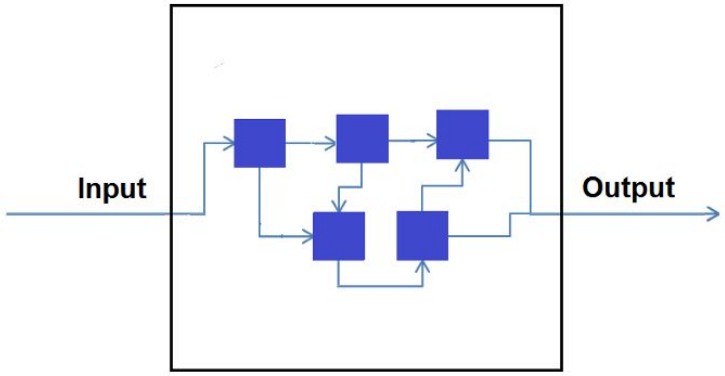
BASIS PATH TESTING

INTRODUCTION

- The white box test is a test case design method that uses the procedural design control structure to derive the test cases. White box tests checks that:

  - All independent paths of each module are executed at least once.
  - Decisions are used in their true part and in their false part.
  - All loops are executed at their limits.
  - All internal data structures are used.



BASIS PATH TESTING

- Basis path testing is a type of white box testing.
- Basis path testing is a hybrid between path testing and branch testing:
- ***Path Testing:*** Testing designed to execute all or selected paths through a computer program
- ***Branch Testing:*** Testing designed to execute each outcome of each decision point in a computer program
- ***Basis Path Testing:*** Testing that fulfills the requirements of branch testing & also tests all of the independent paths that could be used to construct any arbitrary path through the computer program

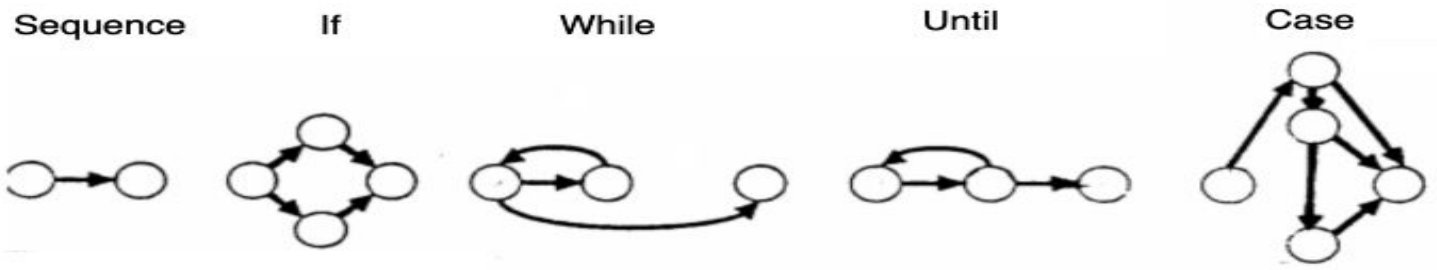
BASIS PATH TESTING STEPS

1. Using the design or code, draw the corresponding flow graph.
2. Determine the cyclomatic complexity of the flow graph.
3. Determine a basis set of independent paths.
4. Prepare test cases that will force execution of each path in the basis set.

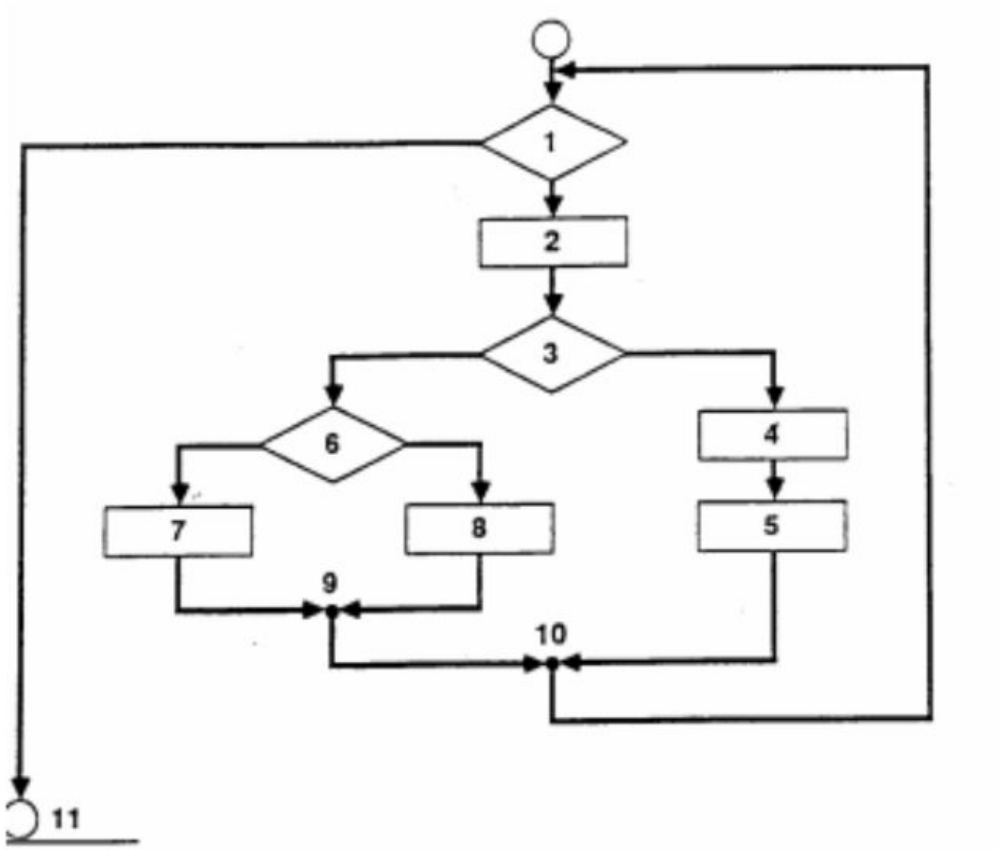
1. ***Using the design or code, draw the corresponding flow graph***

Notation of the flow graph or program graph

- Represents the logical control flow with the following notation:
- Any procedural design can be translated into a control flow graph:
  - Lines (or arrows) called *edges represent flow of control*
  - Circles called *nodes represent one or more actions*
  - Areas bounded by edges and nodes called *regions*
  - A *predicate node is a node containing a condition*



Here is an example based on a flow chart representing the control structure of the program.



2. *Determine the cyclomatic complexity of the flow graph.*

- McCabe's Cyclomatic Complexity is a measure of the number of linearly independent paths through the unit/component. It can therefore be used in structural testing to determine the minimum number of tests that must be executed for complete basis path coverage.
- The cyclomatic complexity, **V(G)**, of a flow graph G, is defined as
  - **V(G) = E – N + 2*P**

  Where

  E: Number of edges

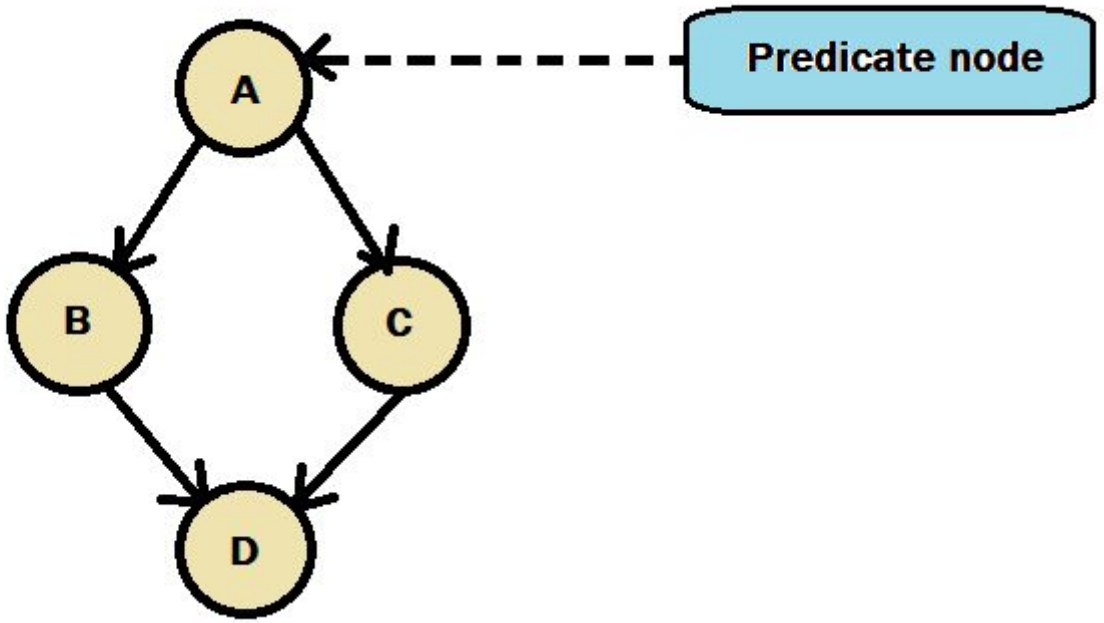
  N: Number of nodes

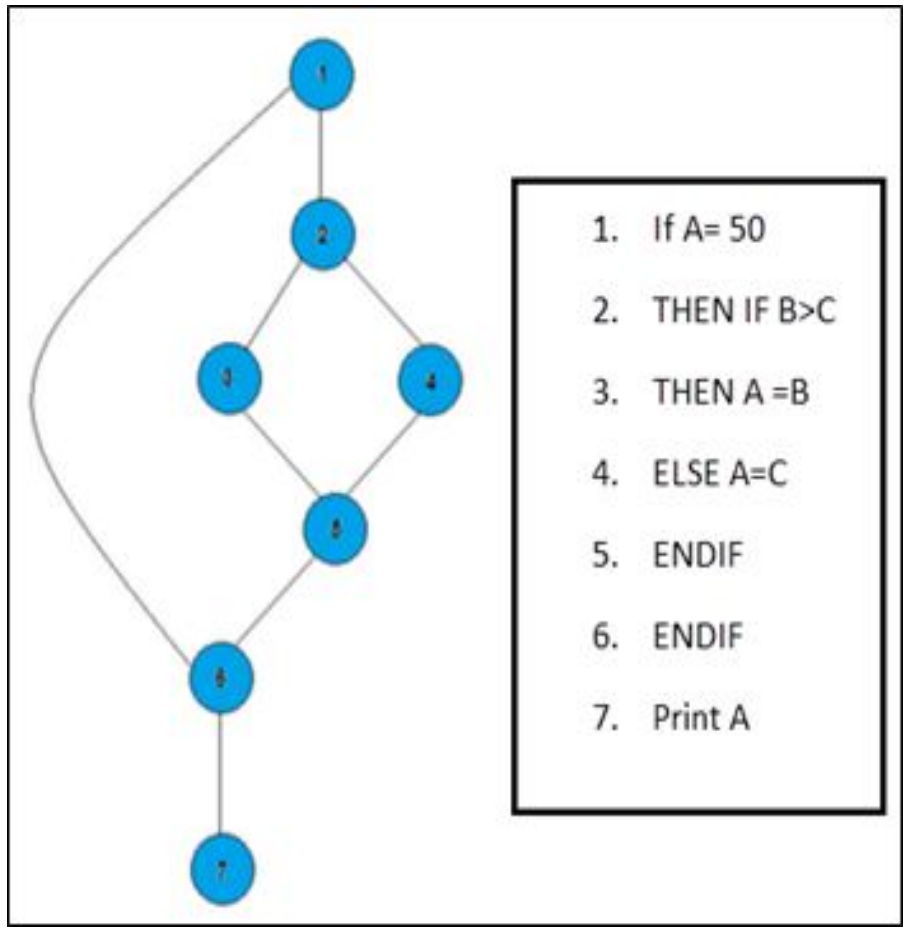
  P: Number of nodes that have exit points

OR

- The cyclomatic complexity, **V(G)**, of a flow graph G, is also defined as
  - **V(G) = p + 1**

    Where p: Number of predicate nodes (nodes with condition)





In the example,

Number of edges= 8
Number of nodes= 7
Number of nodes with exit points= 1

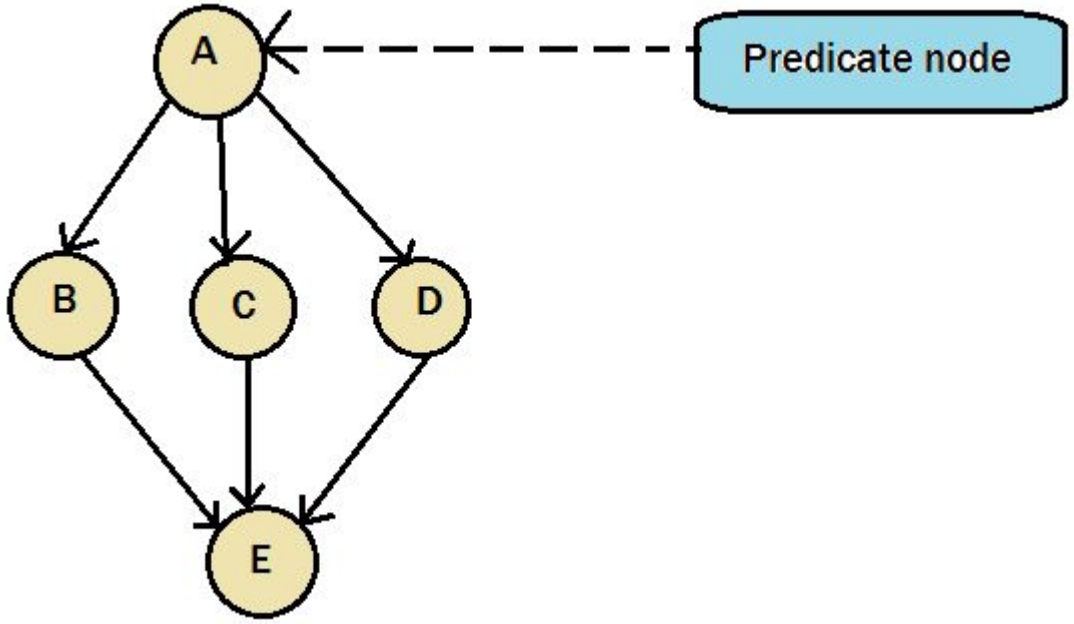
V(G) = E-N +2*P
= 8-7 +2*1
= 3

OR

Number of predicate nodes= 2

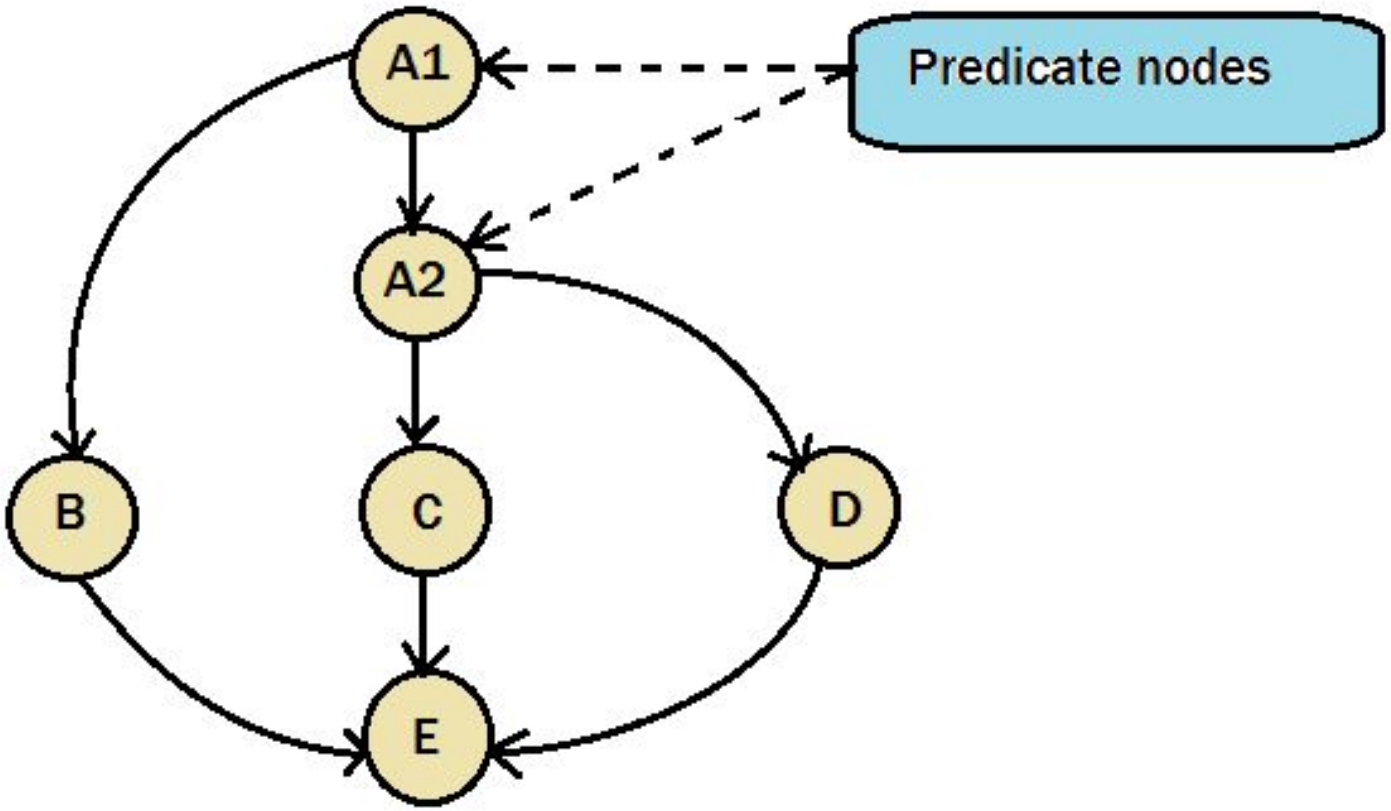
V(G) = p+1
2+1= 3

Limitation on the predicate nodes formula



Thus for this particular graph: V(G) = 1+1 = 2
BUT …….. V(G) = 6-5+2(1) = **3** ……

A limitation on the predicate nodes formula is that it assumes that there are only two outgoing flows for each of such nodes. To adjust for this, split the predicate node into two sub-nodes:



V(G) = 2+1 = 3

3. ***Determine a basis set of independent paths***

- we can see there are few conditional statements that is executed depending on what condition it suffice. Here there are 3 paths or condition that need to be tested to get the output,

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

Properties of Cyclomatic complexity:

- V (G) is the maximum number of independent paths in the graph
- V (G) >=1
- G will have one path if V (G) = 1
- Cyclomatic complexity can be used to:
    - Tell us how many paths to look for in basis path testing
    - Help pinpoint areas of potential instability
    - Indicate a unit/component's testability & understandability (maintainability)
    - Provide a quantitative indication of unit/component's control flow complexity
    - Indicate the effort required to test a unit/component