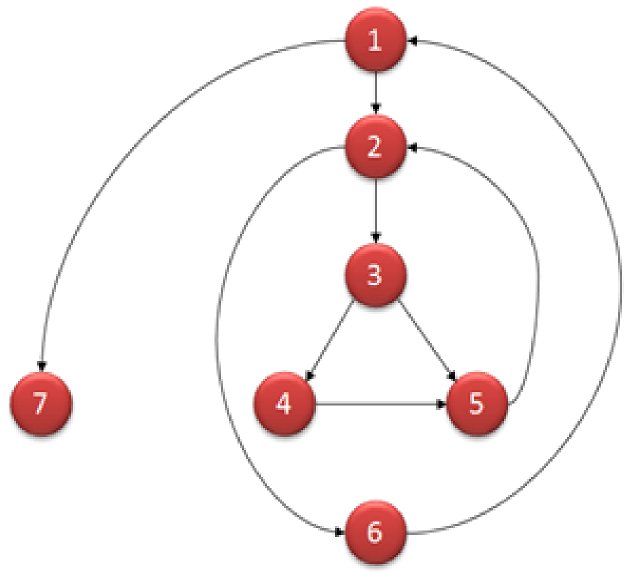
Unit- III
Cyclomatic Complexity

- Dr. Shivani Budhkar

Cyclomatic Complexity Example-1

```
1.  i = 0;
2.  n=4; //N-Number of nodes present in the graph

3.  while (i<n-1) do
4.  j = i + 1;

5.  while (j<n) do

6.  if A[i]<A[j] then
7.  swap(A[i], A[j]);

8.  end do;
9.  i=i+1;

10. end do;
```

Flow Graph for the given program



- **Computing mathematically,**
- V(G) = E-N+2
- V(G) = 9 - 7 + 2 = 4
- V(G) = 3 + 1 = 4 (Condition nodes are 1,2 and 3 nodes)
- Basis Set - A set of possible execution path of a program
- 1, 7
- 1, 2, 6, 1, 7
- 1, 2, 3, 4, 5, 2, 6, 1, 7
- 1, 2, 3, 5, 2, 6, 1, 7

Properties of Cyclomatic complexity:

- Following are the 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
- Minimize complexity to 10

How this metric is useful for software testing?

- Basis Path testing is one of White box technique and it guarantees to execute atleast one statement during testing. It checks each linearly independent path through the program, which **means number test cases, will be equivalent to the cyclomatic complexity of the program.**
- This metric is useful because of properties of Cyclomatic complexity (M) -
- M can be number of test cases to achieve branch coverage (Upper Bound)
- M can be number of paths through the graphs. (Lower Bound)
- **Steps to be followed:**
- The following steps should be followed for computing Cyclomatic complexity and test cases design.

  Step 1 - Construction of graph with nodes and edges from the code

  Step 2 - Identification of independent paths

  Step 3 - Cyclomatic Complexity Calculation

  Step 4 - Design of Test Cases
- Once the basic set is formed, TEST CASES should be written to execute all the paths.

More on V (G):

- Cyclomatic complexity can be calculated manually if the program is small. Automated tools need to be used if the program is very complex as this involves more flow graphs. Based on complexity number, team can conclude on the actions that need to be taken for measure.
- Following table gives overview on the complexity number and corresponding meaning of v (G):
-

Complexity Number	Meaning
1-10	Structured and well written code High Testability Cost and Effort is less
10-20	Complex Code Medium Testability Cost and effort is Medium
20-40	Very complex Code Low Testability Cost and Effort are high
>40	Not at all testable Very high Cost and Effort

Tools for Cyclomatic Complexity calculation:

- Many tools are available for determining the complexity of the application. Some complexity calculation tools are used for specific technologies. Complexity can be found by the number of decision points in a program. The decision points are if, for, for-each, while, do, catch, case statements in a source code.
- Examples of tools are
- OCLint - Static code analyzer for C and Related Languages
- Reflector Add In - Code metrics for .NET assemblies
- GMetrics - Find metrics in Java related applications

Uses of Cyclomatic Complexity:

- Cyclomatic Complexity can prove to be very helpful in
  - Helps developers and testers to determine independent path executions
  - Developers can assure that all the paths have been tested atleast once
  - Helps us to focus more on the uncovered paths
  - Improve code coverage in Software Engineering
  - Evaluate the risk associated with the application or program
  - These metrics being used earlier in the program helps in reducing the risks.

- Cyclomatic Complexity is software metric useful for structured or White Box Testing. It is mainly used to evaluate complexity of a program. If the decision points are more, then complexity of the program is more. If program has high complexity number, then probability of error is high with increased time for maintenance and trouble shoot.

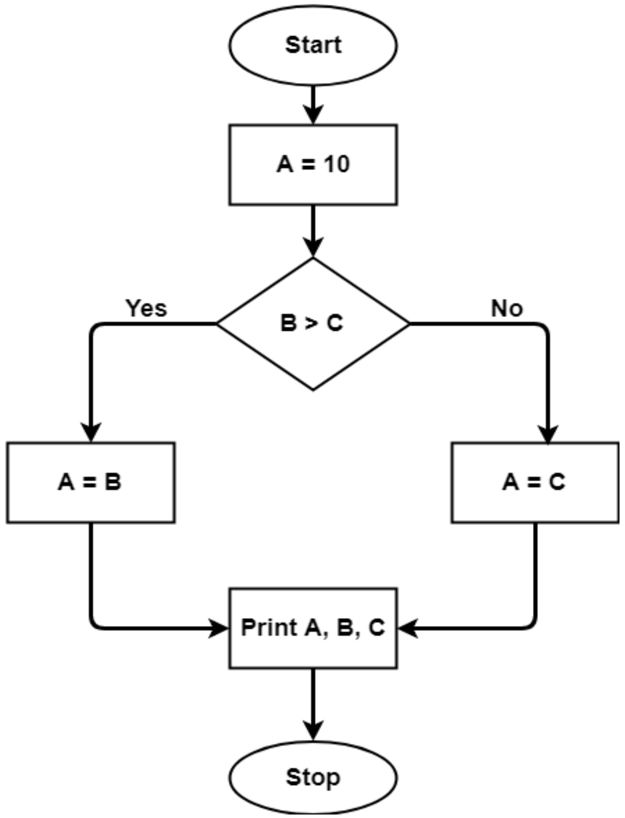
Advantages & Disadvantages of Cyclomatic Complexity

- **Advantages of Cyclomatic Complexity:**.
  - It can be used as a quality metric, gives relative complexity of various designs.
  - It is able to compute faster than the Halstead's metrics.
  - It is used to measure the minimum effort and best areas of concentration for testing.
  - It is able to guide the testing process.
  - It is easy to apply.
- **Disadvantages of Cyclomatic Complexity:**
  - It is the measure of the programs's control complexity and not the data the data complexity.
  - In this, nested conditional structures are harder to understand than non-nested structures.
  - In case of simple comparisons and decision strucures, it may give a misleading figure.

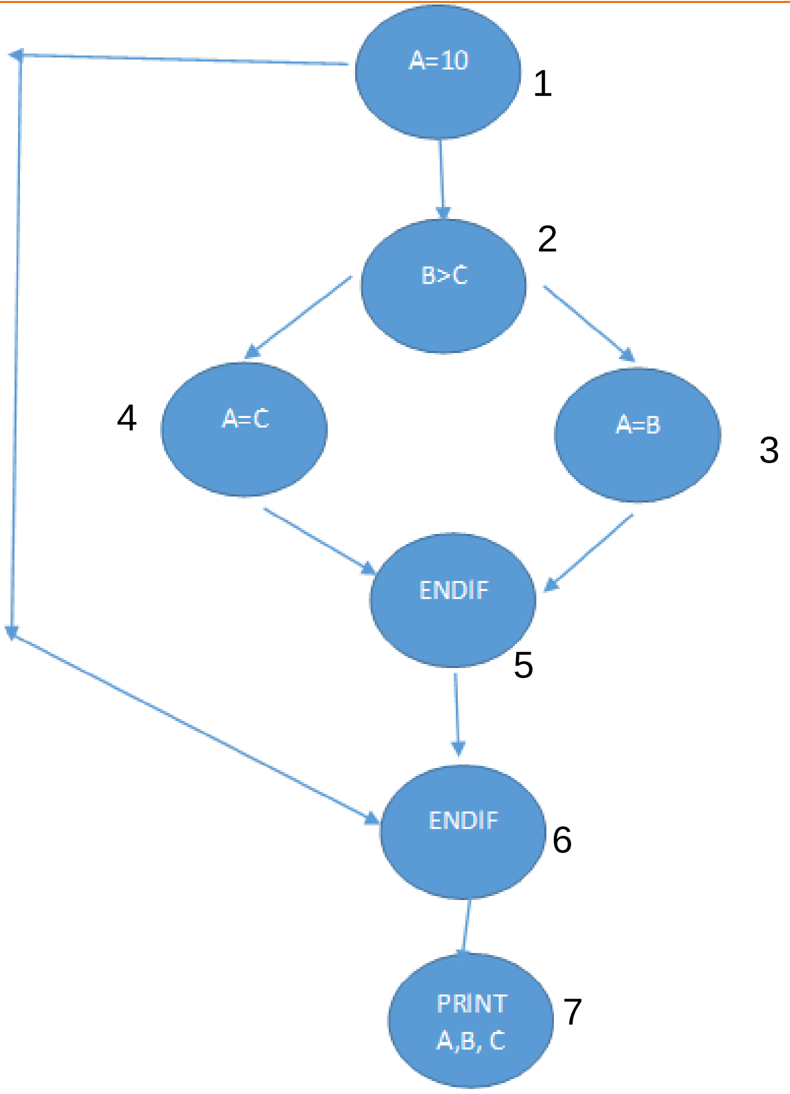
Cyclomatic Complexity Example-2

```
1.  A = 10
2.      IF B > C THEN
3.          A = B
4.      ELSE
5.          A = C
6.      ENDIF
7.      ENDIF
8.  Print A
9.  Print B
10. Print C
```

Flow Chart



Control Flow Graph



Calculate Cyclomatic Complexity

No. of nodes = 7

No. of edges = 8

Cyclomatic Complexity V(G) = E-N+2

= 8-7+2

= 3

Linear Independent Path:

1-2-5-6-7

1-2-4-5-6-7

1-7

So 3 test cases need to be designed

Test Cases

Test Case ID	Test Case Description	Type of input	Steps	Test Data	Expected Output	Actual Output	Status
TC_001	Print value of A,B,C for A=10,B>C	Valid Input	Read A,B,C such that A=10,B>C	A=10,B=15, C=12	Print A=10,B=15 ,C=12	A=10,B=15, C=12	Pass
TC_002	Print value of A,B,C for A=10,B<C	Valid Input	Read A,B,C such that A=10,B<C	A=10,B=12, C=15	Print A=10,B=12 ,C=15	A=10,B=12, C=15	Pass
TC_003	Print value of A,B,C for A=10,B=C	Valid Input	Read A,B,C such that A=10,B=C	A=10,B=12, C=12	Print A=10,B=12 ,C=12	A=10,B=12, C=12	Pass