White Box Testing

CYCLOMATIC COMPLEXITY

- ***Cyclomatic complexity*** is a software metric that provides a quantitative measure of the logical complexity of a program.

- Cyclomatic complexity **defines number of independent paths** which can be further used in development of test cases.

Complexity is computed in one of three ways:

1. The **number of regions** of the flow graph corresponds to the cyclomatic complexity.

2. Cyclomatic complexity *V*(*G*) for a flow graph *G* is defined as

$$V(G) = E - N + 2$$ (**McCabe Complexity Measure**)

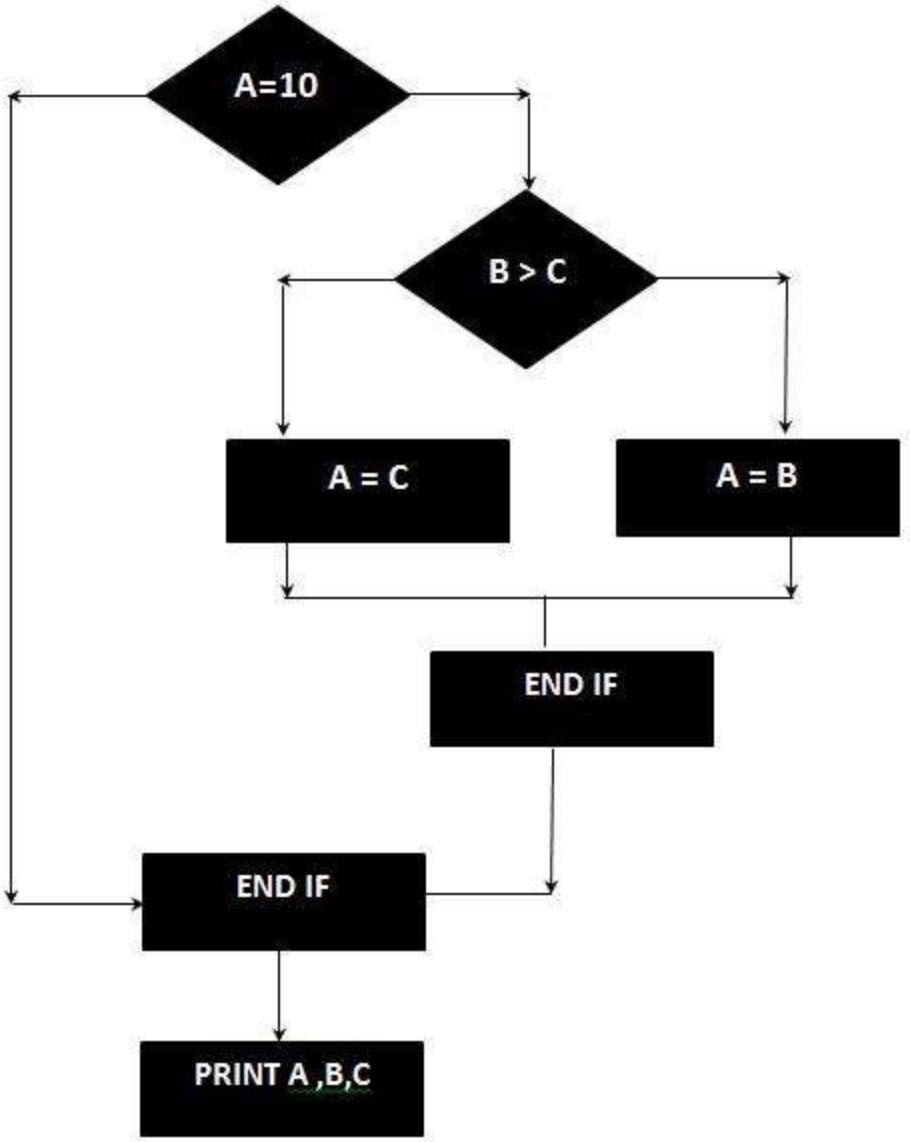
where *E* is the number of flow graph edges and *N* is the number of flow graph nodes.

3. Cyclomatic complexity *V*(*G*) for a flow graph *G* is also defined as

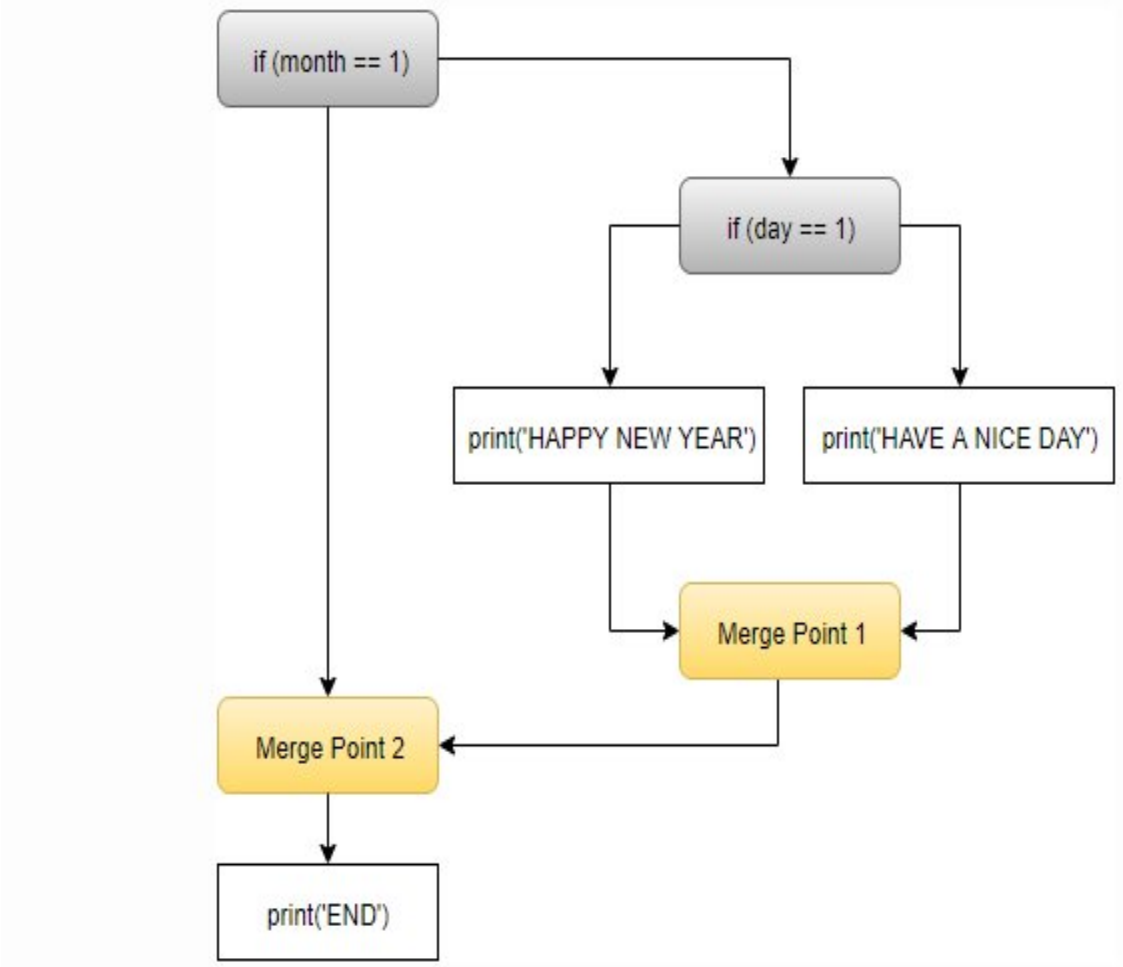
$$V(G) = P + 1$$

where *P* is the number of predicate nodes contained in the flow graph *G*.

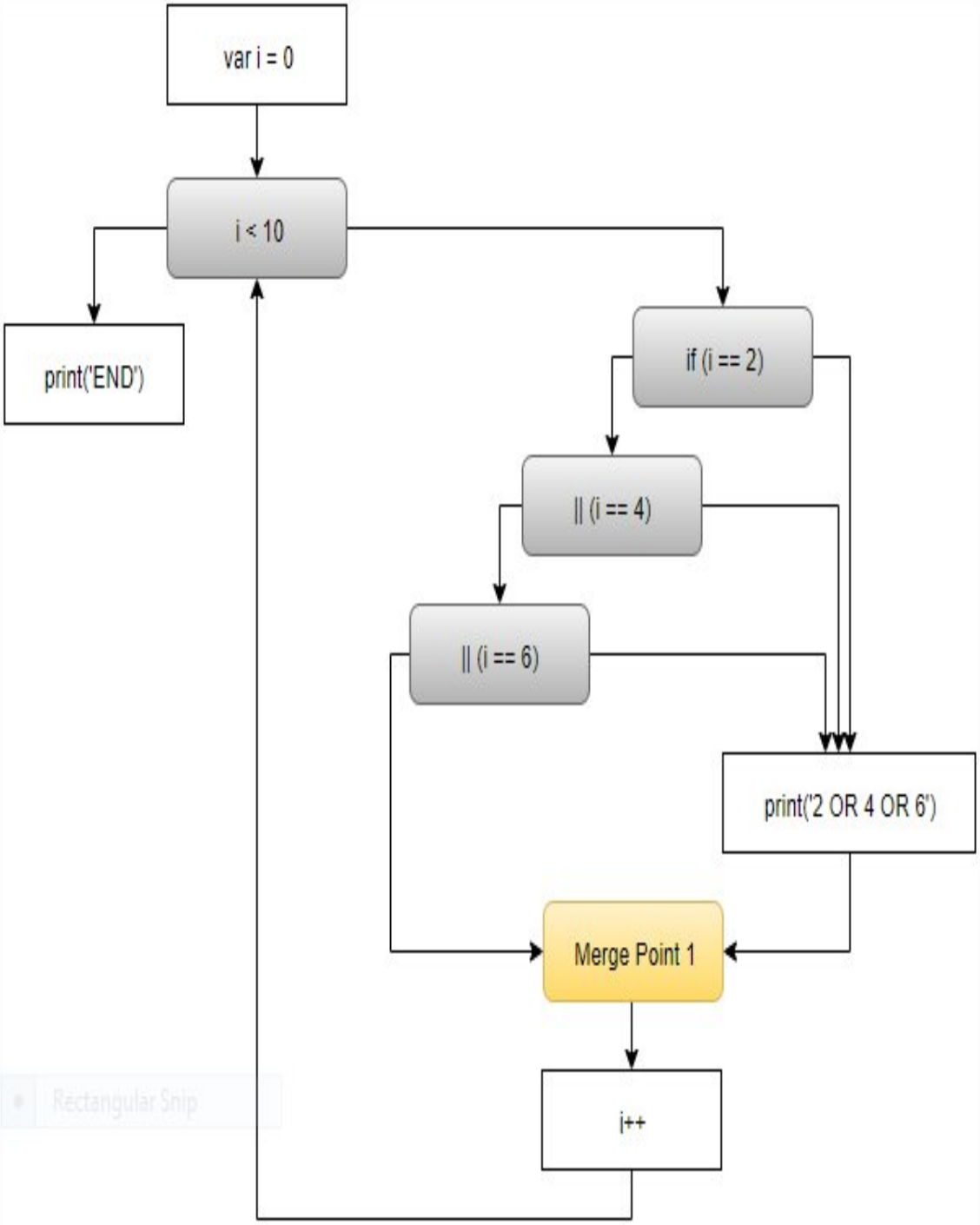
```
IF A = 10 THEN
IF B > C  THEN
A = B
ELSE
A = C
ENDIF
ENDIF
Print A
Print B
Print C
```



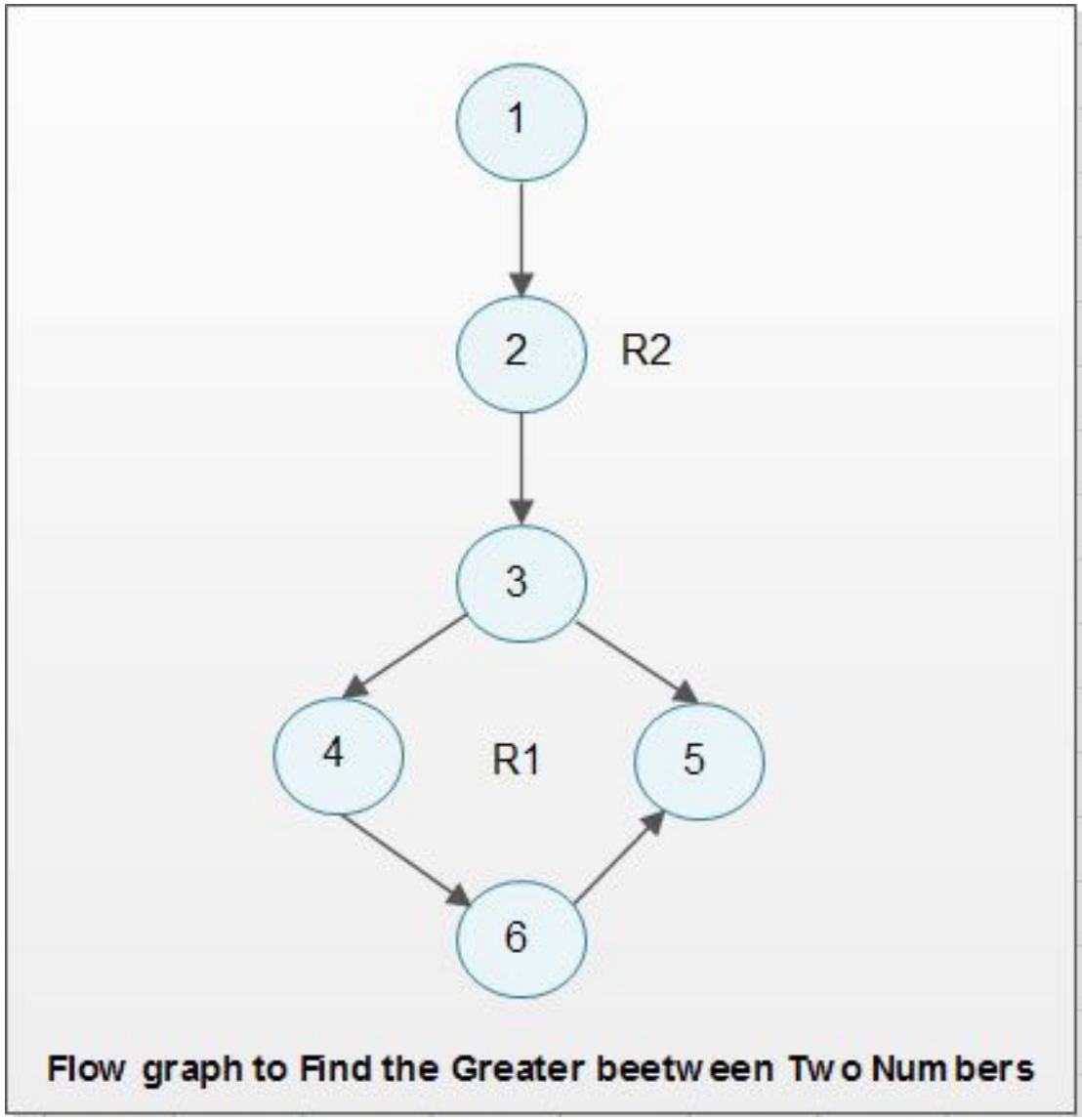
```
if (month == 1)
 {
 if (day == 1)
{
 print('HAPPY NEW YEAR');
}
 else
 {
 print('HAVE A NICE DAY');
}
 }
 print('END');
```



```
for (var i = 0; i < 10; i++)
 {
 if (i == 2 || i == 4 || i == 6)
{
 print('2 OR 4 OR 6');
 }
 }
 print('END');
```



```
procedure greater;
integer: x, y, z = 0;
enter the value of x;
enter the value of y;
if x > y then
z = x;
else
z = y;
end greater
```



Flow graph to Find the Greater beetween Two Numbers

Determine all independent paths through the program

For the flow graph depicted the independent paths are listed below.

P1: 1-2-3-4-6

P2: 1-2-3-5-6

Compute the cyclomatic complexity

CC = 2 as there two regions R1 and R2

or

CC 6 edges - 6 nodes + 2 =2

or

CC 1 predicate node + 1 = 2.

More Numericals

- Black Box Testing

Guidelines for Equivalence Class Partitioning

- An input condition specifies a range [a, b]
  - one equivalence class for a < X < b, and
  - two other classes for X < a and X > b to test the system with invalid inputs
- An input condition specifies a set of values
  - one equivalence class for each element of the set $\{M_1\}$, $\{M_2\}$, ..., $\{M_N\}$, and
  - one equivalence class for elements outside the set $\{M_1, M_2, ..., M_N\}$
- Input condition specified for each individual value
  - If the system handles each valid input differently then create one equivalence class for each valid input

Guidelines for Equivalence Class Partitioning

- An input condition specifies the number of valid values (Say " N ")
  - Create **one** equivalence class for **the correct number of inputs**
  - Two equivalence classes for invalid inputs – **one for zero values and one for more than N values**
- An input condition specifies a "must be" value
  - Create one equivalence class for a "must be" value, and
  - one equivalence class for something that is not a "must be" value

Identification of Test Cases

Test cases for each equivalence class can be identified by:

- Assign a unique number to each EC
- For each EC with **valid input** that has not been covered by test cases yet, write a new test case covering **as many** uncovered EC as possible
- For each EC with **invalid input** that has not been covered by test cases, write a new test case that covers **one and only one** of the uncovered EC

Example-Adjusted Gross Income

Consider a software system that computes income tax based on adjusted gross income (AGI) according to the following rules:

- If AGI is between $1 and $29,500, the tax due is 22% of AGI.
- If AGI is between $29,501 and $58,500, the tax due is 27% of AGI.
- If AGI is between $58,501 and $100 billion, the tax due is 36% of AGI.

Example

Condition 1

$1 ≤ AGI ≤ $29,500, to derive two ECs:

EC1: $1 ≤ AGI ≤ $29,500; valid input.

EC2: AGI < 1; invalid input.

Condition 2

$29,501 ≤ AGI ≤ $58,500, to derive one EC:

EC3: $29,501 ≤ AGI ≤ $58,500; valid input.

Condition 3

$58,501 ≤ AGI ≤ $100 billion, to derive two ECs:

EC4: $58,501 ≤ AGI ≤ $100 billion; valid input.

EC5: AGI > $100 billion; invalid input.

Example

TABLE 9.10 Generated Test Cases to Cover Each Equivalence Class

Test Case Number	Test Value	Expected Result	Equivalence Class Being Tested
TC_1	$22,000	$4,840	EC1
TC_2	$46,000	$12,420	EC3
TC_3	$68,000	$24,480	EC4
TC_4	$-20,000	Rejected with an error message	EC2
TC_5	$150 billion	Rejected with an error message	EC5

Boundary Value Analysis (BVA)

- Select test data near the boundary of a data domain so that data both **within and outside** an equivalence class are selected
- Extension and refinement of the equivalence class partitioning technique
- **"Boundary conditions for each of the equivalence class are analyzed in order to generate test cases"**

Guidelines for Boundary Value Analysis

Equivalence class specifies a range

- Construct test cases by considering the **boundary points of the range** and points just beyond the boundaries of the range Eg: -10.0<=X<=10.0

{-9.9,-10.0,-10.1}and {9.9,10.0,10.1}

Equivalence class specifies an ordered set

- Eg:-linear list, table, or a sequential file, then focus attention on the **first and last elements of the set.**

Guidelines for Boundary Value Analysis

- **Equivalence class specifies a number of values**
  - Construct test cases for the **minimum and the maximum value of the number**
  - In addition, select a value smaller than the minimum and a value larger than the maximum value.
  - Eg: Student hostel where a room can be shared by 1 to 4 students.

Test cases for 1 ,4 ,0 ,5 students developed

Solve for BVA

Consider a software system that computes income tax based on adjusted gross income (AGI) according to the following rules:

- If AGI is between $1 and $29,500, the tax due is 22% of AGI.
- If AGI is between $29,501 and $58,500, the tax due is 27% of AGI.
- If AGI is between $58,501 and $100 billion, the tax due is 36% of AGI.

EC1: $1≤ AGI ≤ $29,500;

This would result in values of $1, $0, $–1, $1.50 and $29,499.50, $29,500, $29,500.50.

EC2: AGI < 1;

This would result in values of $1, $0, $–1, $–100 billion.

EC3: $29,501 ≤ AGI ≤ $58,500;

This would result in values of $29,500, $29,500.50, $29,501, $58,499, $58,500, $58,500.50, $58,501.

EC4: $58,501≤AGI≤$100 billion;

This would result in values of $58,500, $58,500.50, $58,501, $100 billion, $101 billion

EC5: AGI > $100 billion;

This would result in $100 billion, $101 billion, $10000 billion.