Boolean Algebra

Boolean Algebra

- Boolean algebra provides the operations and the rules for working with the set **{0, 1}.**
- These are the rules that underlie **electronic circuits**, and the methods **VLSI design**.

Boolean Operations

- The **complement** is denoted by a bar (on the slides, we will use a minus sign). It is defined by
- $-0 = 1$ and $-1 = 0$.

- The **Boolean sum**, denoted by $+$ or by OR, has the following values:
- $1 + 1 = 1, \quad 1 + 0 = 1, \quad 0 + 1 = 1, \quad 0 + 0 = 0$

- The **Boolean product**, denoted by $\cdot$ or by AND, has the following values:
- $1 \cdot 1 = 1, \quad 1 \cdot 0 = 0, \quad 0 \cdot 1 = 0, \quad 0 \cdot 0 = 0$

Boolean Functions and Expressions

- **Example:** Give a Boolean expression for the Boolean function F(x, y) as defined by the following table:

x	y	F(x, y)
0	0	0
0	1	1
1	0	0
1	1	0

Possible solution: $F(x, y) = (\neg x) \cdot y$

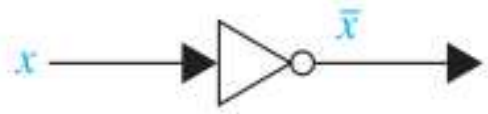
Example: Find the values of the Boolean function represented by F (x, y, z) = xy + z

- Sol:

TABLE					
x	y	z	xy	$\bar{z}$	$F(x, y, z) = xy + \bar{z}$
1	1	1	1	0	1
1	1	0	1	1	1
1	0	1	0	0	0
1	0	0	0	1	1
0	1	1	0	0	0
0	1	0	0	1	1
0	0	1	0	0	0
0	0	0	0	1	1

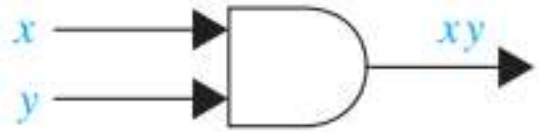
Logic Gates

Basic Types of Gates:



(a) Inverter

(b) OR gate



(c) AND gate

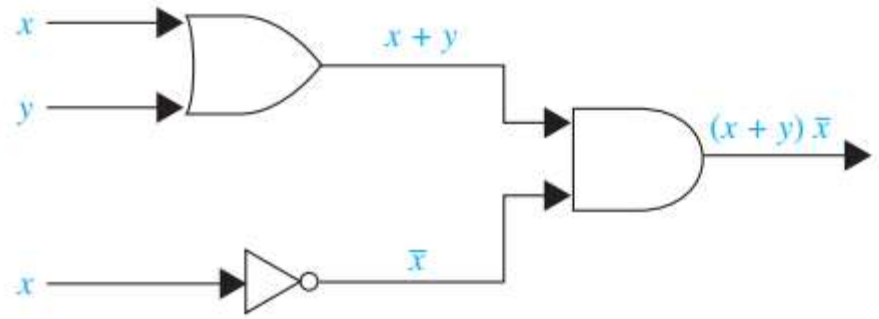
Combinations of Gates

Constructed using a combination of inverters, OR gates, and AND gates.

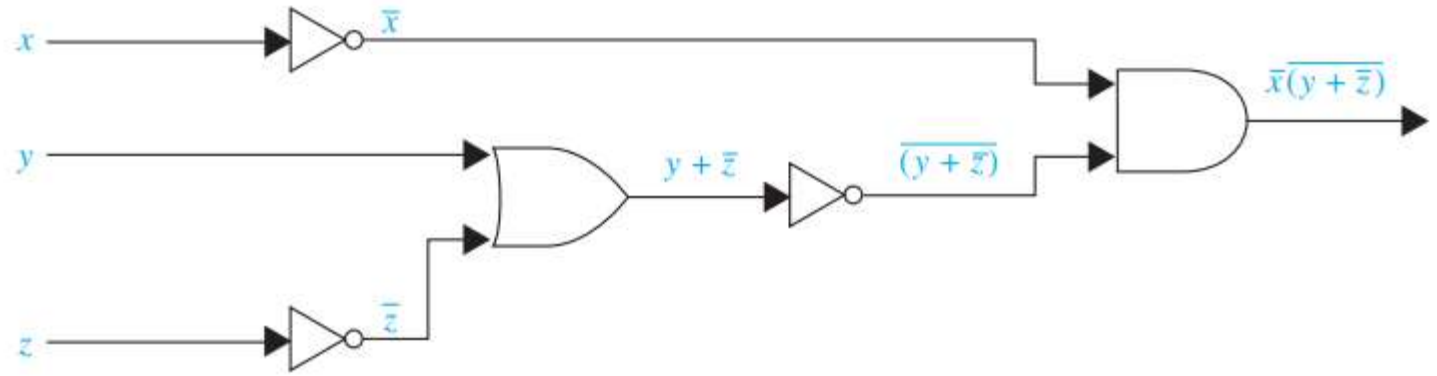
Example: Construct circuits that produce the following outputs:

(a) $(x + y)\overline{x}$, (b) $\overline{x}\ \overline{(y + \overline{z})}$, and (c) $(x + y + z)(\overline{x}\ \overline{y}\ \overline{z})$.

(a)

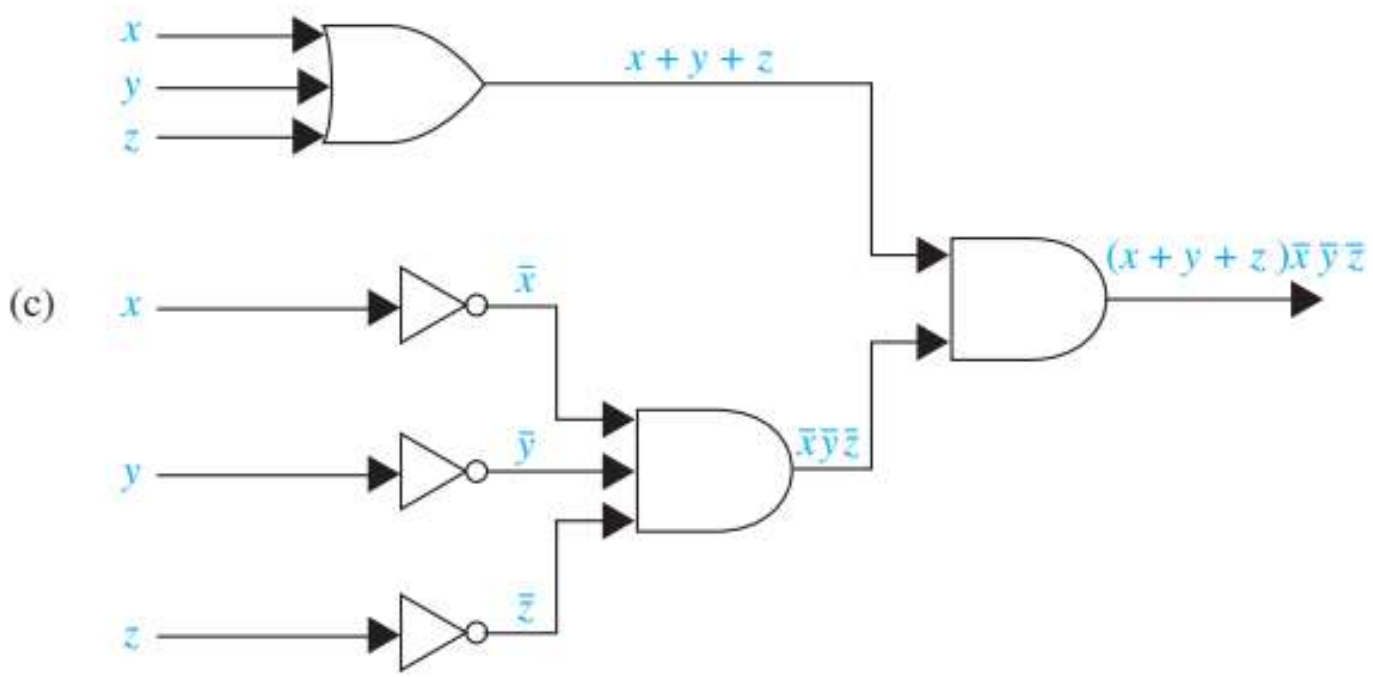


(b)



(c) $(x + y + z)(\bar{x}\ \bar{y}\ \bar{z})$

Sol:



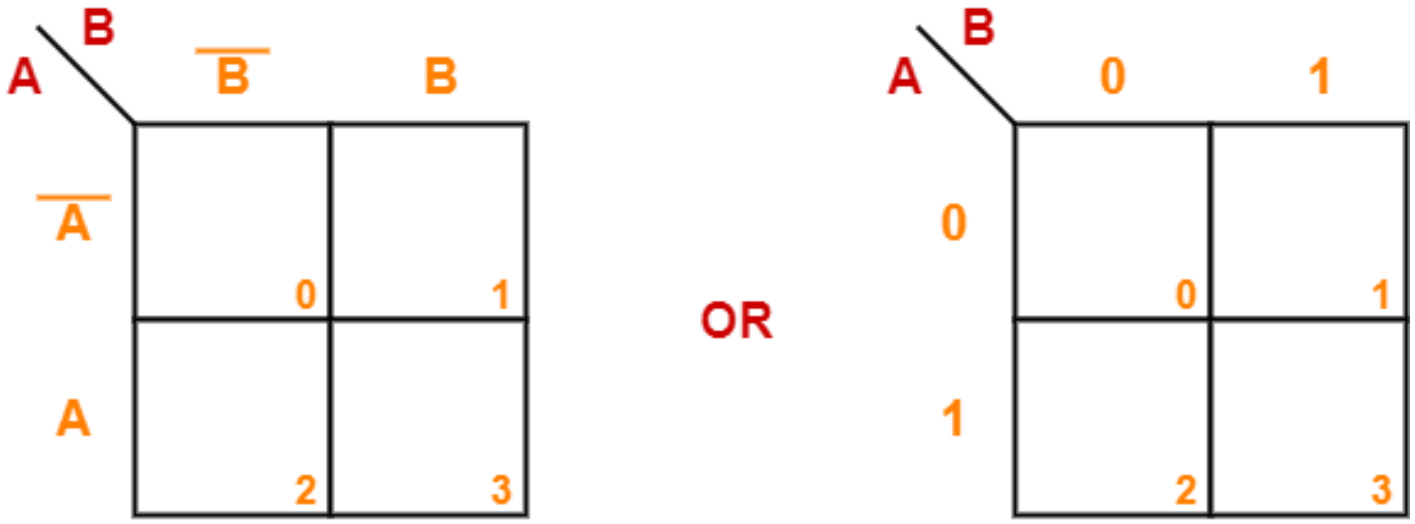
BOOLEAN FUNCTION MINIMIZATION

Karnaugh map or K-map:

- For a boolean expression consisting of n-variables, number of cells required in K Map = 2^n cells

Two Variable K Map:

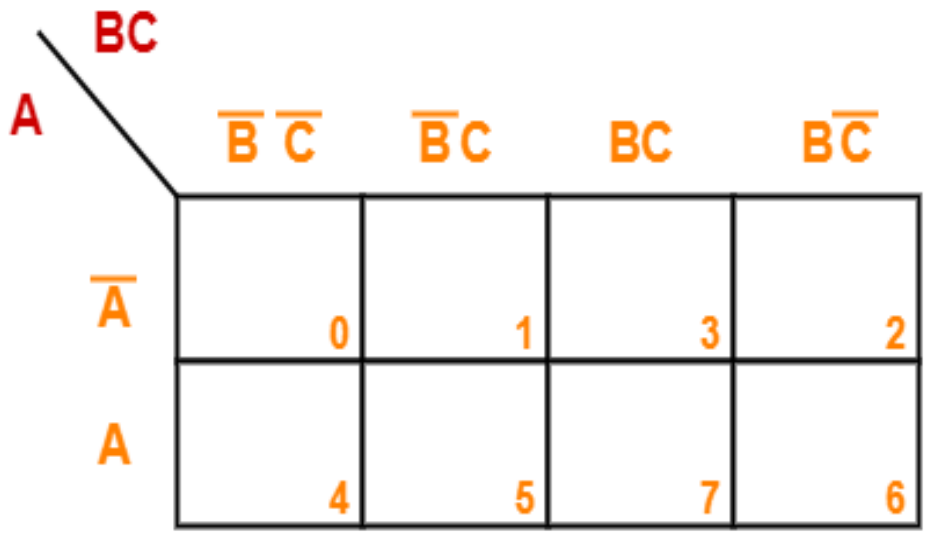
- The number of cells present in two variable K Map = 2^2 = 4 cells.
- So, for two variables, we draw a 2 x 2 K Map.
- Here, A and B are the two variables of the given boolean function.



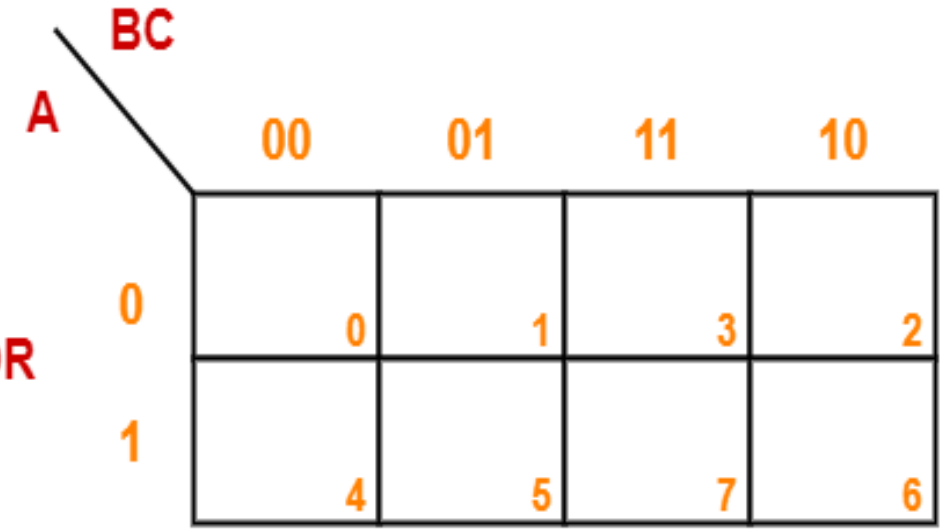
Two Variable K Map

Three Variable K Map:

- The number of cells present in three variable **K Map = 2^3 = 8 cells**.
- So, for three variables, we draw a **2 x 4 K Map**.
- Here, A, B and C are the three variables of the given boolean function.

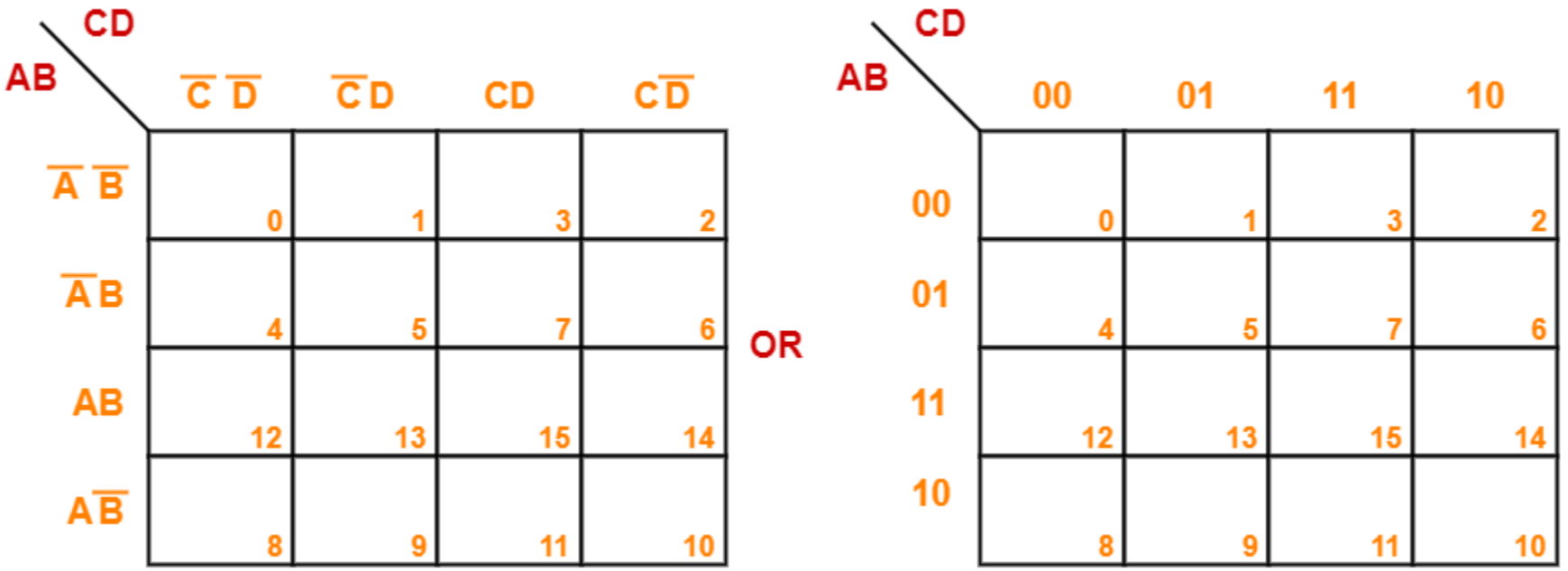


OR



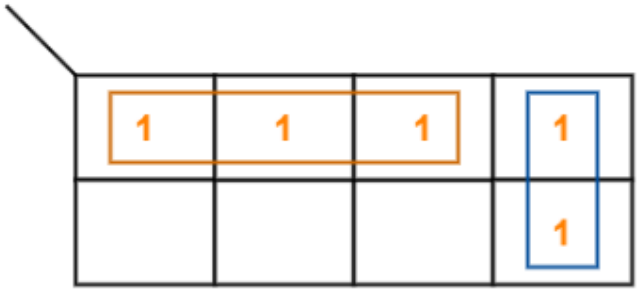
Four Variable K Map:

- The number of cells present in four variable **K Map = 2^4 = 16 cells**.
- So, for four variables, we draw a **4 x 4 K Map**.
- Here, A, B, C and D are the four variables of the given boolean function.

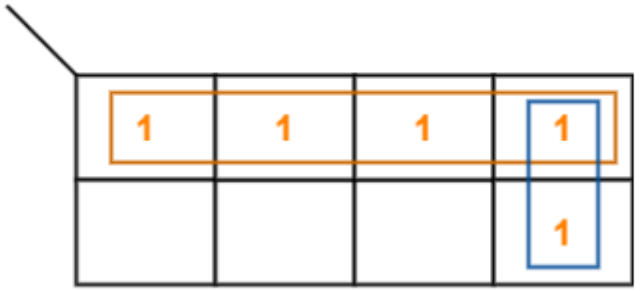


Karnaugh Map Simplification Rules:

- Draw a K Map and Fill k map with 0's and 1's according to its function.
- We can either group 0's with 0's or 1's with 1's but we can not group 0's and 1's together.
- Groups may overlap each other.
- Only create a group whose number of cells can be represented in the power of 2 that is 2^n i.e. 1, 2, 4, 8, 16 and so on number of cells.

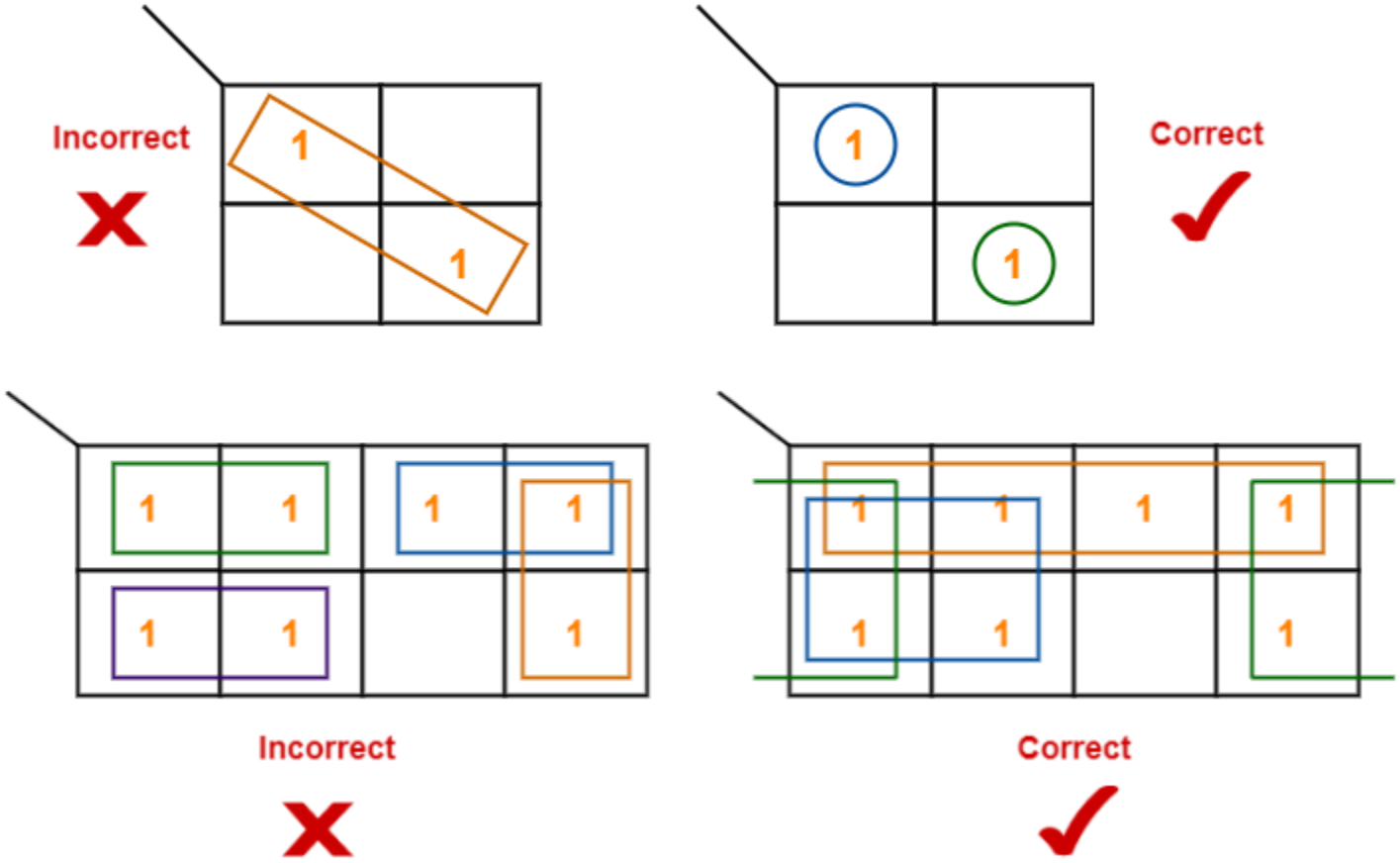


Incorrect

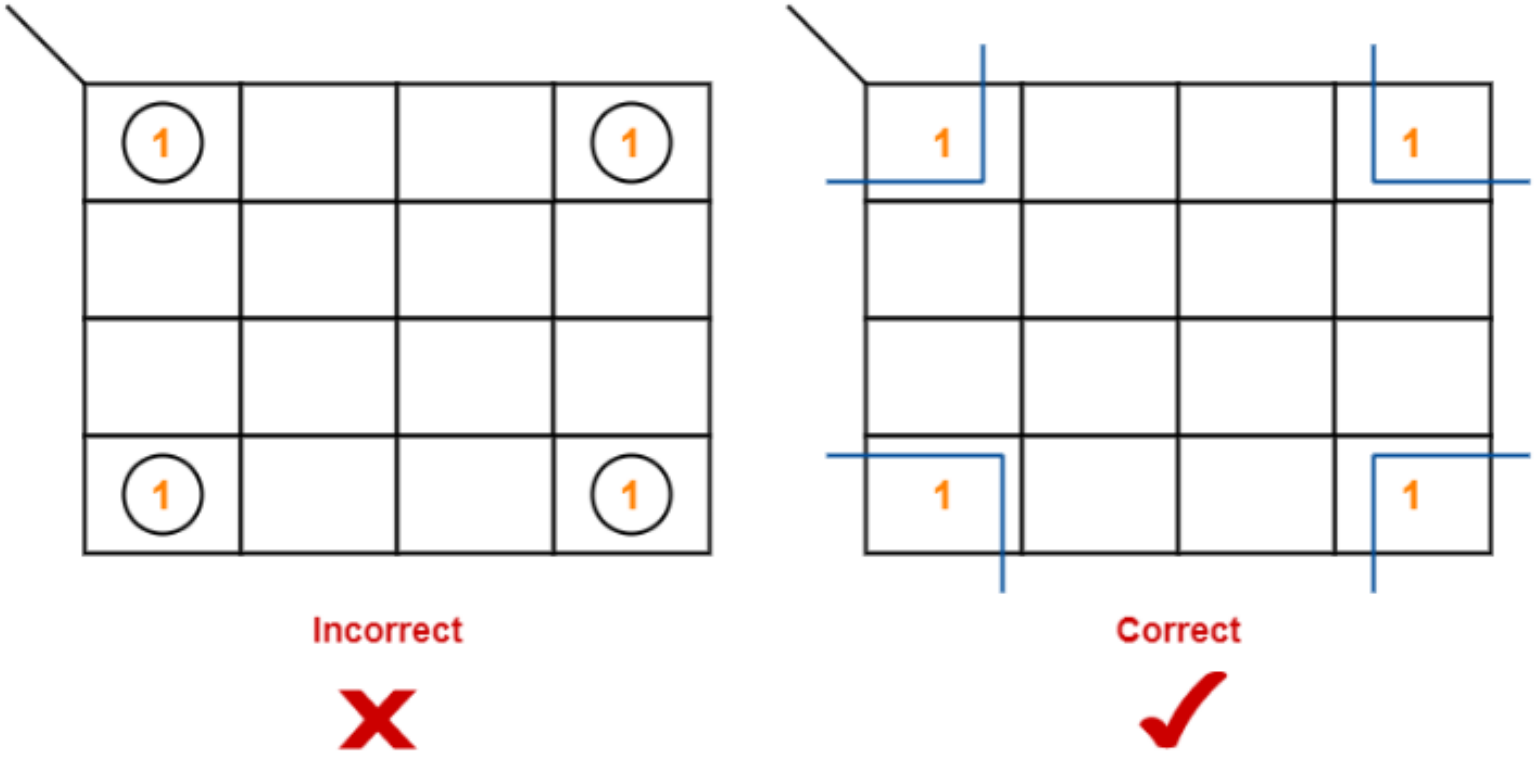


Correct

- Groups can be only either horizontal or vertical.
- We can not create groups of diagonal or any other shape.
- Each group should be as large as possible.



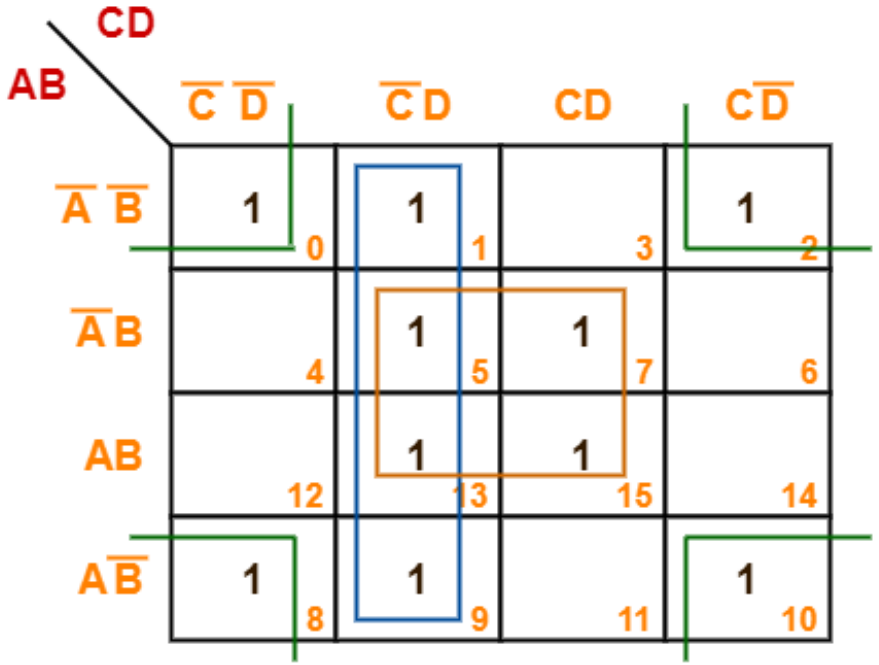
- Opposite grouping and corner grouping are allowed.
- There should be as few groups as possible.



Example: Minimize the following boolean function-

$$F(A, B, C, D) = \Sigma m(0, 1, 2, 5, 7, 8, 9, 10, 13, 15)$$

- **Sol:** Draw a 4 x 4 K Map. Fill the cells of K Map in accordance with the given boolean function and now form the groups as per rules:
- F(A, B, C, D) = (A'B + AB)(C'D + CD) + (A'B' + A'B + AB + AB')C'D + (A'B' + AB')(C'D' + CD')
- So minimized boolean expression is-
- = BD + C'D + B'D



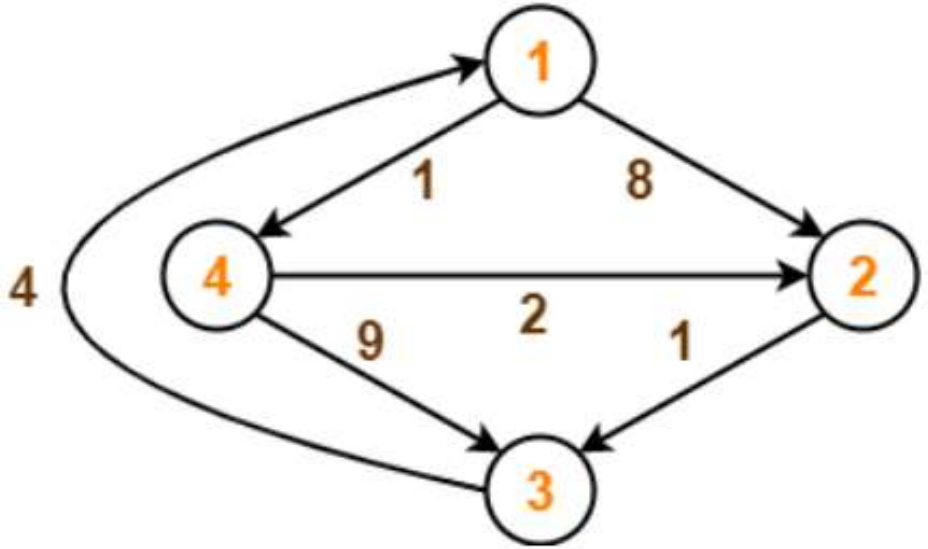
Floyd Warshall Algorithm:

- It is used to solve All Pairs Shortest Path Problem.
- It computes the shortest path between every pair of vertices of the given graph.
- Floyd Warshall Algorithm is an example of dynamic programming approach.
- It is extremely simple and easy to implement.
- It is best suited for dense graphs because its complexity depends only on the number of vertices in the given graph.

Algorithm:

1. **Create a |V| x |V| matrix // It represents the distance between every pair of vertices as given**
2. **For each cell (i,j) in M do-**
3. **if i = = j**
4. **M[i][j] = 0 // For all diagonal elements, value = 0**
5. **if (i , j) is an edge in E**
6. **M[i][j] = weight(i,j) // If there exists a direct edge between the vertices, value = weight of edge**
7. **else**
8. **M[i][j] = infinity // If there is no direct edge between the vertices, value = ∞**
9. **for k from 1 to |V|**
10. **for i from 1 to |V|**
11. **for j from 1 to |V|**
12. **if M[i][j] > M[i][k] + M[k][j]**
13. **M[i][j] = M[i][k] + M[k][j] // Equal to the shortest value**

Q: Using Warshall Algorithm, find the shortest path distance between every pair of vertices.



Solution:

➢Step 1:

- Remove all the self loops and parallel edges (keeping the lowest weight edge) from the graph.
- In the given graph, there are neither self edges nor parallel edges.

➢Step 2:

- Write the initial distance matrix which represents the distance between every pair of vertices in the form of given weights.
- For diagonal elements (representing self-loops), distance value = 0.
- For vertices having a direct edge between them, distance value = weight of that edge.
- For vertices having no direct edge between them, distance value = ∞.

Initial distance matrix for the given graph is-

$$D_0 = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \end{array} \begin{array}{c} \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{bmatrix} 0 & 8 & \infty & 1 \\ \infty & 0 & 1 & \infty \\ 4 & \infty & 0 & \infty \\ \infty & 2 & 9 & 0 \end{bmatrix} \end{array}$$

Step 3: Using Floyd Warshall Algorithm, write the following 4 matrices-

- $D_1 =$

	1	2	3	4
1	0	8	∞	1
2	∞	0	1	∞
3	4	12	0	5
4	∞	2	9	0

$D_2 =$

	1	2	3	4
1	0	8	9	1
2	∞	0	1	∞
3	4	12	0	5
4	∞	2	3	0

$D_3 =$

	1	2	3	4
1	0	8	9	1
2	5	0	1	6
3	4	12	0	5
4	7	2	3	0

$D_4 =$

	1	2	3	4
1	0	3	4	1
2	5	0	1	6
3	4	7	0	5
4	7	2	3	0

The last matrix D_4 represents the shortest path distance between every pair of vertices.

Remember:

In the question, there are 4 vertices in the given graph.
So, there will be total 4 matrices of order 4 x 4 in the solution excluding the initial distance matrix.
Diagonal elements of each matrix will always be 0.

$$D_4 = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \end{array} \begin{array}{c} \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{bmatrix} 0 & 3 & 4 & 1 \\ 5 & 0 & 1 & 6 \\ 4 & 7 & 0 & 5 \\ 7 & 2 & 3 & 0 \end{bmatrix} \end{array}$$