# Graphs المخططات

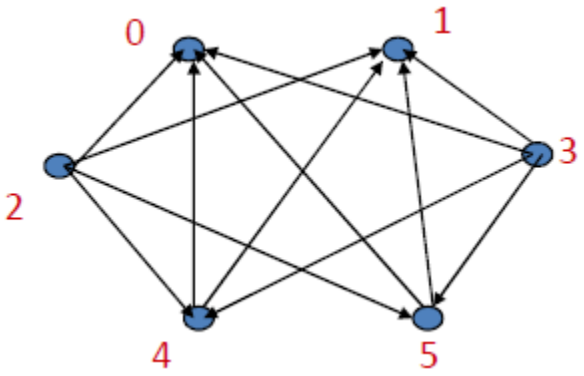


**Graph** is a data structure that consists of the following two components:
**1.** A finite set of vertices also called as nodes.
**2.** A finite set of ordered pair of the form (u, v) called as edge.
The pair is ordered because (u, v) is not same as (v, u) in case of directed graph(di-graph). The pair of form (u, v) indicates that there is an edge from vertex u to vertex v.

* The edges may contain weight/value/cost.

Graphs are used to represent many real life applications:
1- Graphs are used to represent networks. The networks may include paths in a city or telephone network or circuit network.
2- Graphs are also used in social networks like facebook . For example, in facebook, each person is represented with a vertex(or node). Each node is a structure and contains information like person id, name, gender and locale.

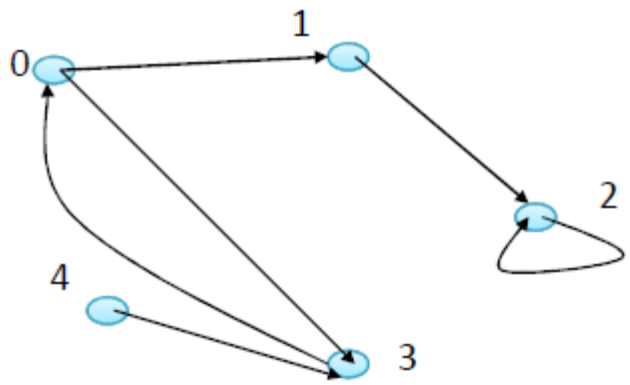
Following is an example directed graph with 6 vertices.




Following two are the most commonly used representations of graph.
**1.** Adjacency Matrix
**2.** Adjacency List
The choice of the graph representation is situation specific. It totally depends on the type of operations to be performed and ease of use.

## Adjacency Matrix:

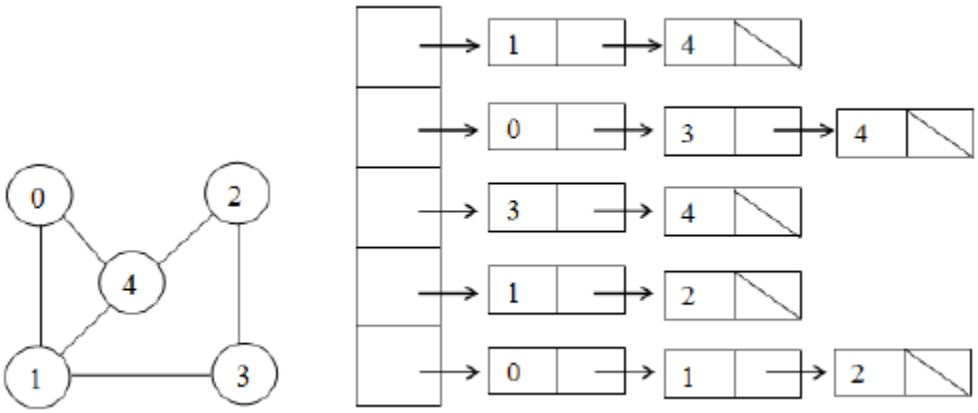
• In this representation, each graph of **n** nodes is represented by an **n x n** matrix A, that is, a two-dimensional array A
• The nodes are (re)-labeled 1,2,…,n
1. A [i] [j] = 1 if (i,j) is an edge
2. A [i] [j] = 0 if (i,j) is not an edge




| A | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| 0 | 0 | 1 | 0 | 1 | 0 |
| 1 | 0 | 0 | 1 | 0 | 0 |
| 2 | 0 | 0 | 1 | 0 | 0 |
| 3 | 1 | 0 | 0 | 0 | 0 |
| 4 | 0 | 0 | 0 | 1 | 0 |

## Adjacency List:

• A graph of **n** nodes is represented by a one-dimensional array **L** of linked lists, where
– **L [i]** is the linked list containing all the nodes adjacent from node **i**.
– The nodes in the list **L[i]** are in no particular order.



**Graphs** can be **undirected** or **directed:**
**Undirected graph:** When the edges in a graph have no direction, the graph is called undirected.
**Directed graph:** When the edges in a graph have a direction, the graph is called directed.

## Basic Operations on graphs:

### Additions

- addNode**:** adds vertices to your graph
- addEdge**:** creates edges between two given vertices in your graph

### Removals

- removeNode**:** removes vertices from your graph
- removeEdge**:** removes edges between two given vertices in your graph

### Search

- contains**:** checks if your graph contains a given value
- hasEdge**:** checks if a connection exists between two given nodes in your graph

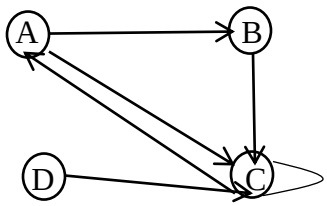
### Display Vertex

Displays a vertex of the graph.

**Properties of graphs**:

**1. Adjacent nodes**: two nodes are adjacent if they are connected by an edge
• **Ex: 1** is adjacent to **3.**

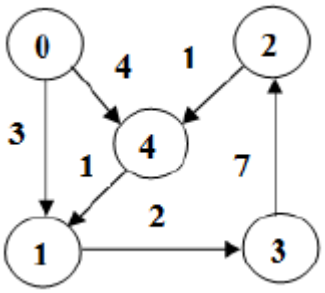

**Examples of Graphs:**
• V = { A, B, C, D}
• E = { (A,B), (B,C), (A,C), (C,A), (C,C), (D,C) }
• When ( x, y) is an edge,
• we say that x is adjacent to y, and y is adjacent from x.
• A is adjacent to B.
• B is not adjacent to A.



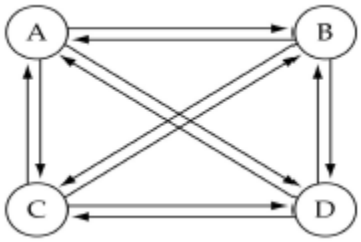
• C is adjacent from B.

**2. Path:** a sequence of vertices that connect two nodes in a graph.

**3. Weighted graph:** a graph in which each edge carries a value




**4. Complete graph:** a graph in which every vertex is directly connected to every other vertex.




**5-self loop:**

**6-multi_edge:**

**7-**The length of the path is N-1.

**8-**simple path الطريق البسيط : no repeated vertices

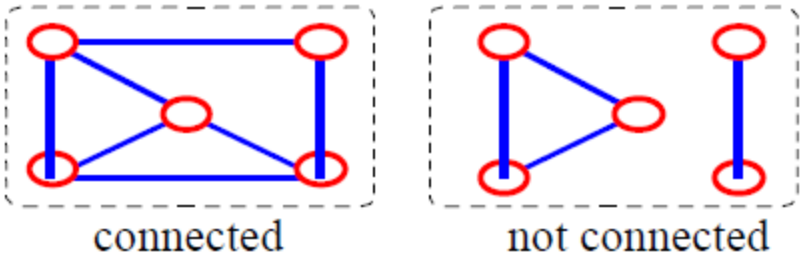
**9-**Cycle الحلقة : simple path, except that the last vertex is the same as the first vertex

A cycle is a path such that $v_1=v_N$

**10-**connected graph: There is a path between each two vertices

**نستطيع ان نصل الى اي عقدة بواسطة اي عقدة اخرى بطريقة مباشرة او غير مباشرة**

**11**-Disconnected graph : There are at least two vertices not connected by a path.




To Search for a certain node or traverse all nodes in the graph:

**-Depth First Search البحث بالعمق**

Once a possible path is found, continue the search until the end of the path

**-Breadth First Search البحث بالعرض**

Start several paths at a time, and advance in each one step at a time

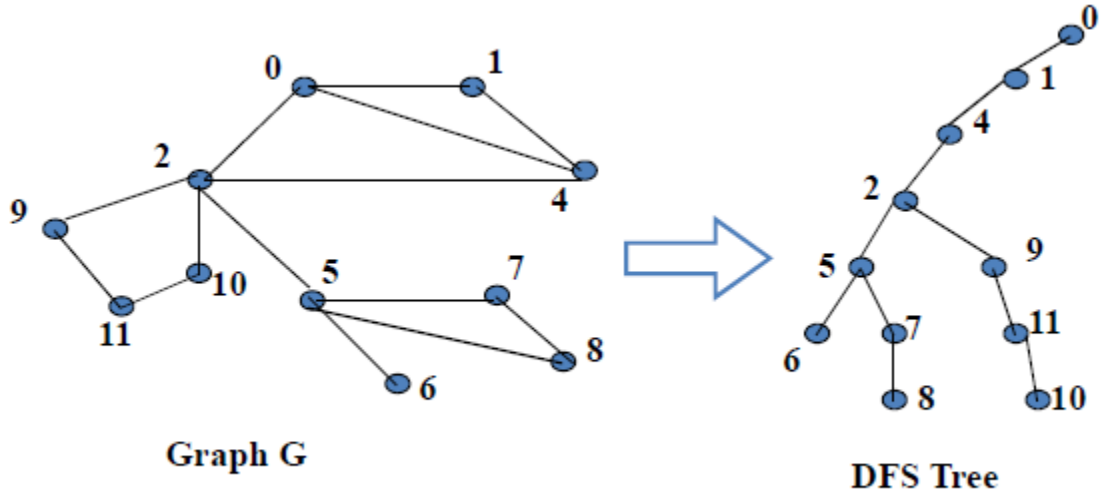
## Depth-First Search(DFS):

**DFS follows the following rules:**

1. Select an unvisited node **x**, visit it, and treat as the current node .
2. Find an unvisited neighbor of the current node, visit it, and make it the new current node;
3. If the current node has no unvisited neighbors, backtrack to the its parent, and make that parent the new current node;
4. Repeat steps 3 and 4 until no more nodes can be visited.

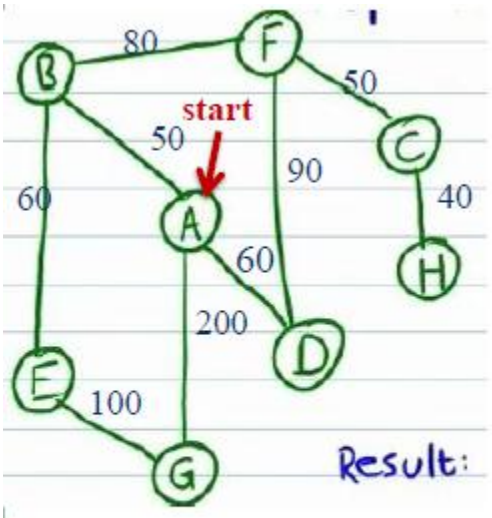
5. If there are still unvisited nodes, repeat from step 1.

**Examples**:




Graph G

DFS Tree




Result: A B E G F C H D