Southern Technical University
Technical Institute / Qurna
Dep. of Computer Systems Techniques

Second class
Subject : Data Structures
Lecturer : Israa Mahmood Hayder
Lecture no.16,17

هياكل البيانات اللاخطية

(Graphs)

ـ الاسبوع السادس عشرـ السابع عشر ـ

السادس عشر ـ السابع عشر	* هياكل البيانات اللاخطية non-linear data structures. ـ المخططات graphs. ـ أنواع المخططات graphs types. ـ طرق تمثيل المخططات graphs representation.

B// Rationale (مبررات الوحدة) :-

This unit introduces important mathematical structures called graphs that have applications in subjects as diverse as sociology, chemistry, geography, and electrical engineering. We shall study methods to represent graphs with the data structures available to us and shall construct several important algorithms for processing graphs.

C// Central Ideas (الفكرة المركزية):-

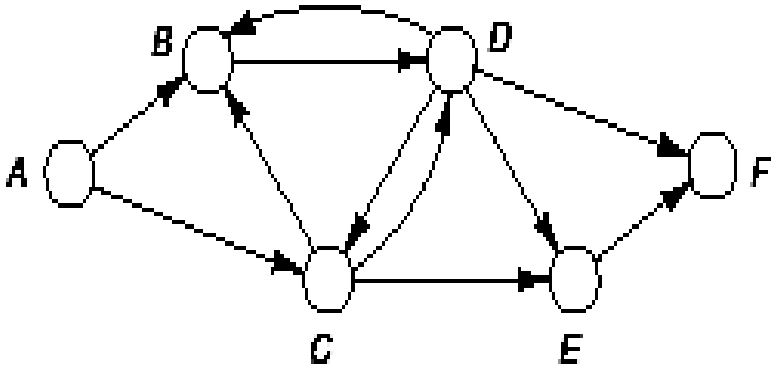
- Definitions and Examples
- Graph types (Directed and Undirected Graphs)
- Graph representation

D// Objectives (أهداف الوحدة):-

After studying this unit , the student will be able to:-

- Define the graph
- Recognize the graph types and representation

Circle the correct answer considering the shown graph:



1- Outdegree of the node d:

a) 2 b)4 c)6

2- The indegree of the node c :

a) 2 b)3 c)5

3- The graph is:

a) cyclic b)acyclic c)isolated

4- The longest path is:

a) B to E b)A to E c)A to F

5- which path is simple path:

a) B to D b) C to F c) A to D

6- Adjacent set for vertex C:

a) {A,D,E,F} b) {B,D,E,F} c) {C,D,E,F}

7- Adjacent set for vertex E:

a) {D,C} b) {B,D,E,F} c) {F}

8- The degree of the graph:

a) 2 b) 3 c) 4

9- The following is a cycle in the graph:

a) ((A,B),(B,D),(D,F)) b) ((A,C),(C,B),(B,A)) c) ((D,E),(E,F))

10- The ((B,D),(D,C),(C,B)) is :

a) a cycle b)simple path c) loop

1- Examples and Definitions

A graph can be thought of as collection of points joined together by ***lines*** or ***Arcs***.

- The points on a graph are called ***Vertices*** or ***Nodes***.

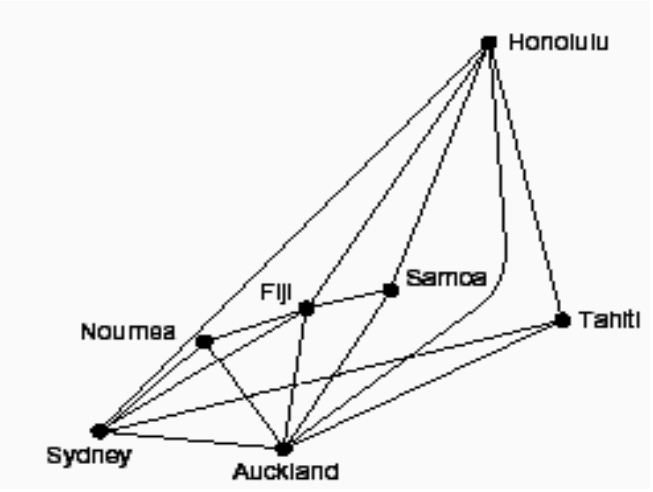


Figure (17-1a)

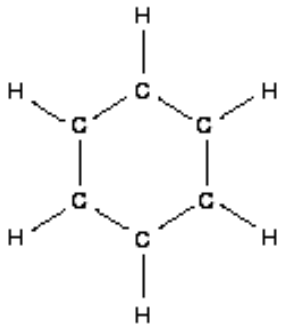


Figure (17-1b)

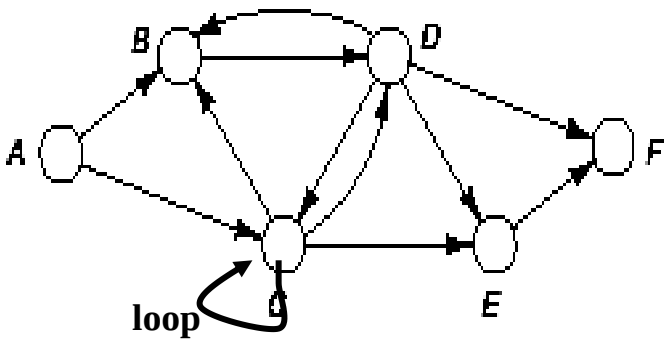
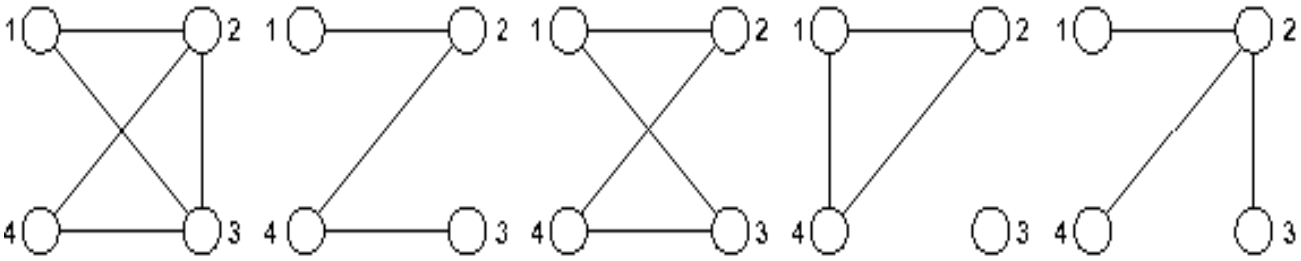


Figure (17-1c)

- A node which is not adjacent to any other node called is an ***Isolated node***.
- A graph is called ***connected*** if there is a path from any vertex to any other vertex.
- If graph is disconnected, we shall refer to a maximal subset of connected vertices as a ***component***.
- ***Path*** from one node to Another is a sequence of arcs.
- ***Loop*** is an edge of the graph which join node to itself.
- ***Cycle:*** is the path that begin and end with the same vertex and no arcs occurs more than once in the path.
- ***Cyclic Graph:*** is a graph that have a cycle.
- ***Acyclic Graph:*** is a graph that have no cycle in it.
- ***Simple path:*** path does not contain cycle.
- ***Adjacent nodes:*** Nodes that are connected by an edge in the graph.



(a) Connected (b) Path (c)Cycle (d)Disconnected (e) Tree

Figure (17-2): Various kinds of undirected graphs

- ***Graph*** is collection of vertices (V) connected with edges (G)

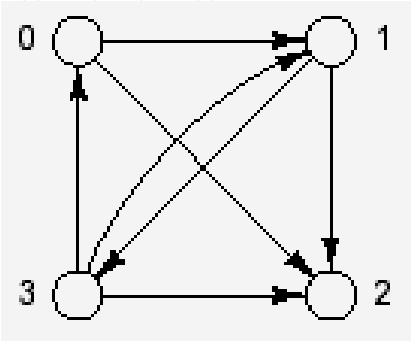
G=(E,V)
In figure (17-c):
Vertics V(G)= { A, B, C, D, E, F }
Edges E(G)= { (A,B), (B,D), (D,F), (D,B), (D,C),(C,B)(C,D),(A,C),(C,E), (C,D) }

- ***Length of the path:*** is the number of arcs appearing in the sequence of the path.

Ex1: In Figure(17-2a) the path between 1 and 3 is: (1,2),(2,3)
And path length=2

Ex2: In the following figure(17.3):

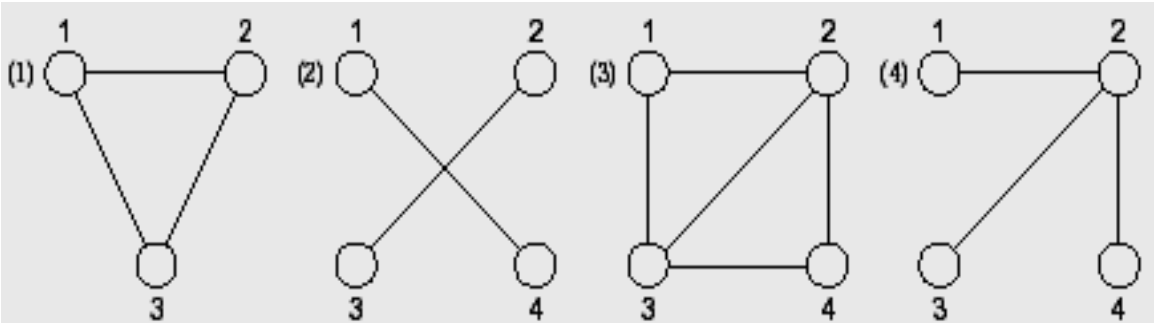
Path example: ((3,1),(0,1),(1,2))
((0,1), (1,3))
Cycle example: ((0,1), (1,3), (3,0))
((3,1), (1,3))



Figure(17-3)

Quiz1:

1- In following graphs :



(a) Find all the cycles in each of the following graphs.
(b) Which of these graphs are connected?
(c) Which of these graphs are tree?

2- Directed and Undirected Graphs:

If the pairs are unordered, then *G* is called an ***undirected graph***; if the pairs are ordered, then *G* is called a ***directed graph***. The term *directed graph* is often shortened to ***digraph***, and the unqualified term *graph* usually means *undirected graph*. The natural way to picture a graph is to represent vertices as points or circles and edges as line segments or arcs connecting the vertices. If the graph is directed, then the line segments or arcs have arrowheads indicating the direction. Figure 17.3 shows several examples of graphs.

Definitions:

Directed graph: Is a graph that the relation between its edges are ordered, the direction is important to define the relation.

Undirected graph: Is a graph that the relation between nodes is not ordered (undirected), i.e the relation (A,D) is as (D,A).

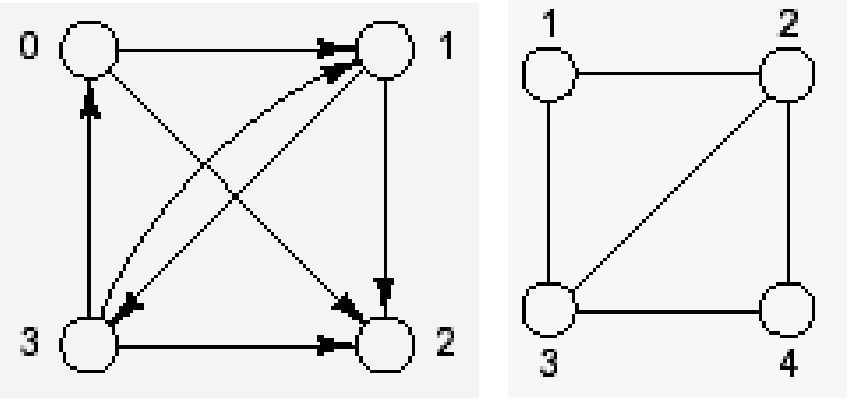


Figure 18.3: (a) directed graph (b) undirected graph

Graph Degree In Directed Graph:

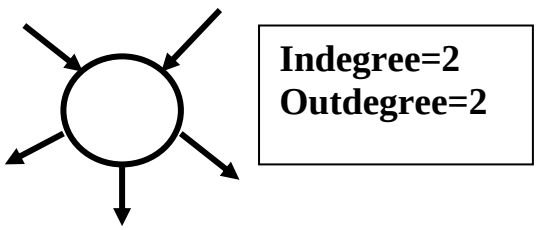
Indegree(Degree): Number of edges any node have as terminal node (input edges).

Outdegree: Number of edges any node have as initial node (output edges).

Totaldegree: Number of edges any node have as initial and terminal node (input and output edges).

Graph degree: the most degree node.

Ex: consider the following node:



Quiz2:

1. What is a *graph*? What are *edges* and *vertices*?

2. What is the difference between an *undirected* and a *directed* graph?

3. Define the terms *path*, *cycle*.

3- Graphs Representation:

There are wo methods for graph representation:

A-The Set Representation

Graphs are defined in terms of sets, and it is natural to look first to sets to determine their representation as data. First, we have a set of vertices, and, second, we have the edges as a set of pairs of vertices. Rather than attempting to represent this set of pairs directly, we divide it into pieces by considering the set of edges attached to each vertex separately.

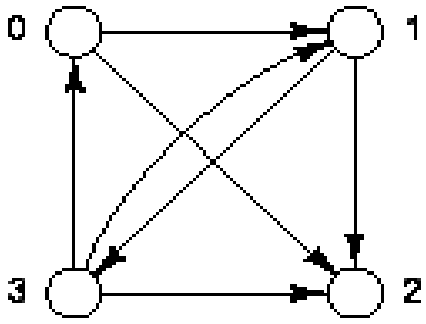
1. Implementation of Sets

There are two general ways for us to implement sets of vertices in data structures and algorithms. One way is to represent the set as a *list* of its elements, the other implementation, often called a ***bit string***, keeps a Boolean value for each potential element of the set to indicate whether or not it is in the set.

2. Adjacency Tables (Matrix)

In the foregoing implementation, the structure Set is essentially implemented as an array of **bool** entries. Each entry indicates whether or not the corresponding vertex is a member of the set. If we substitute this array for a set of neighbors, we find that the array neighbors in the definition of **class** Graph can be changed to an array of arrays, that is, to a two-dimensional array, as follows:

The adjacency table has a natural interpretation: adjacency[v][w] is true if and only if vertex v is adjacent to vertex w. If the graph is directed, we interpret adjacency [v][w] as indicating whether or not the edge from v to w is in the graph. If the graph is undirected, then the adjacency table must be symmetric; that is, adjacency [v][w] = adjacency[w][v] for all v and w. The representation of a graph by adjacency sets and by an adjacency table is illustrated in figure below.



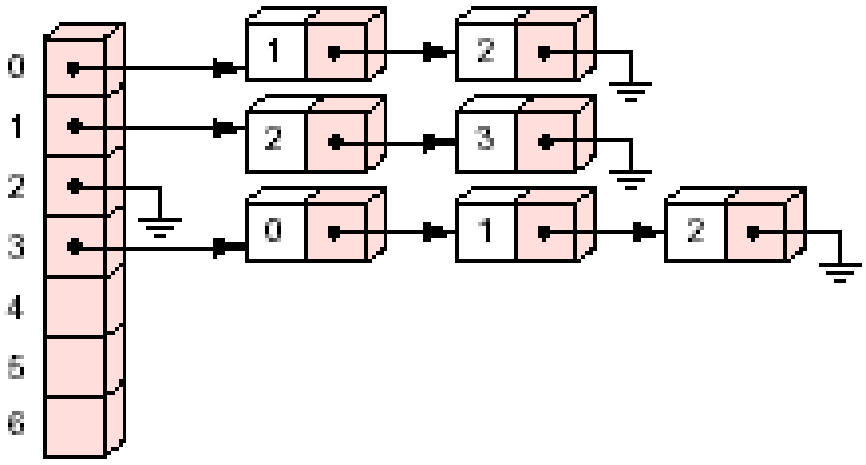
vertex	Set
0	{ 1, 2 }
1	{ 2, 3 }
2	∅
3	{ 0, 1, 2 }

	0	1	2	3
0	F	T	T	F
1	F	F	T	T
2	F	F	F	F
3	T	T	T	F

Directed graph **Adjacency sets** **Adjacency table**

3- Adjacency Lists

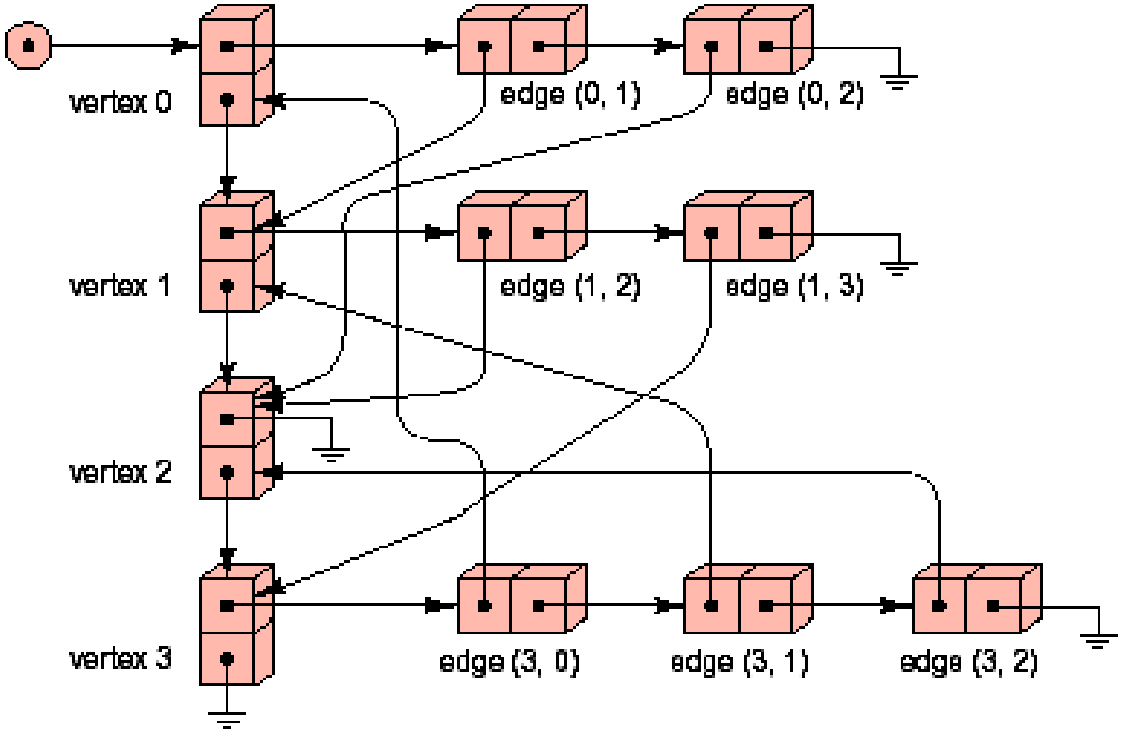
Another way to represent a set is as a *list* of its elements. For representing a graph, we shall then have both a list of vertices and, for each vertex, a list of adjacent vertices. We can consider implementations of graphs that use either contiguous lists or simply linked lists. For more advanced applications, however, it is often useful to employ more sophisticated implementations of lists as binary or multiway search trees or as heaps. Note that, by identifying vertices with their indices in the previous representations, we have *ipso facto* implemented the vertex set as a contiguous list, but now we should make a deliberate choice concerning the use of contiguous or linked lists.



Adjacency Lists

B- Linked Implementation

Greatest flexibility is obtained by using linked objects for both the vertices and the adjacency lists. This implementation is illustrated in the following figure and results in a definition such as the following:



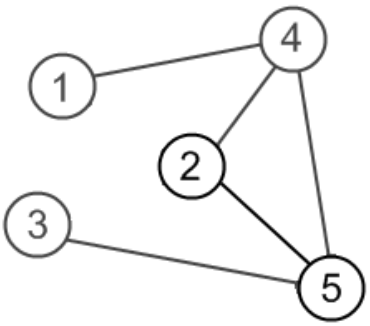
Linked list

Quiz3:

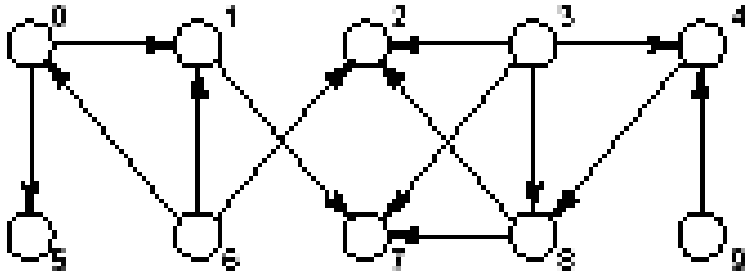
1- Write adjacent table for graph 1.

2- Represent the following graph using adjacency sets and adjacency table

3- Represent the following graphs using adjacency sets and adjacency table



Circle the correct answer considering the shown graph:



1- The degree of the graph:
a) 2 b) 3 c)4

2- The graph is:
a- cyclic b)acyclic c)isolated

3- The path (3,8) is :
a) a cycle b)simple path c) loop

4- The longest path is:
a) 0 to 7 b)1 to 7 c)3 to 7

5- which path is simple path:
a) 3 to 8 b) 3 to 7 c) 9 to 8

6-Outdegree of the node 8:
a) 4 b)3 c)2

7- Adjacent set for vertex 8:
a) {2,3,4,7} b) {2,7} c) {3,7}

8- The indegree of the node 6 :
a) 2 b)3 c)0

9- Adjacent set for vertex 2:
a) {3,7,6} b) (3} c) {3,2}

10- Which of the following is a path in the graph:
a) ((1,2),(2,3)) b) ((1,7),(7,8)) c) ((4,8),(8,3))

References:

1- Data Structures Demystified, by Jim Keogh and Ken Davidson, ISBN:0072253592, McGraw-Hill/Osborne © 2004

2- هياكل البيانات / الطبعة الثانية، تاليف د.عصام الصفار،اصدارات السفير للنشر/ بغداد، ٢٠٠١

3- الحقيبة التعليمية لمادة هياكل البيانات ،اعداد الاستاذ نفارت يوسف الياس ،المعهد التقني كركوك