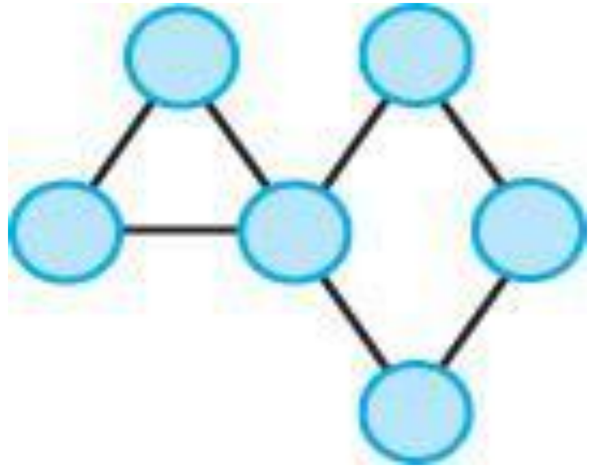
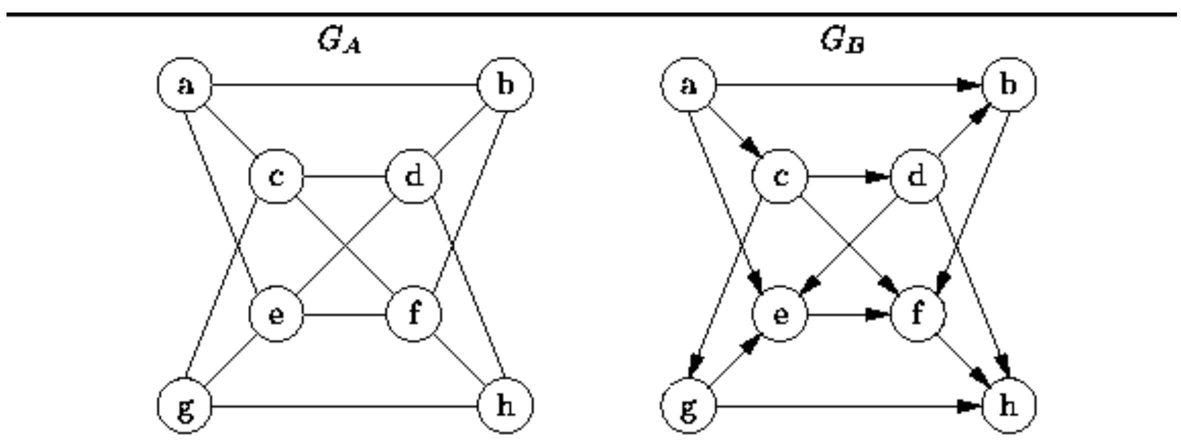


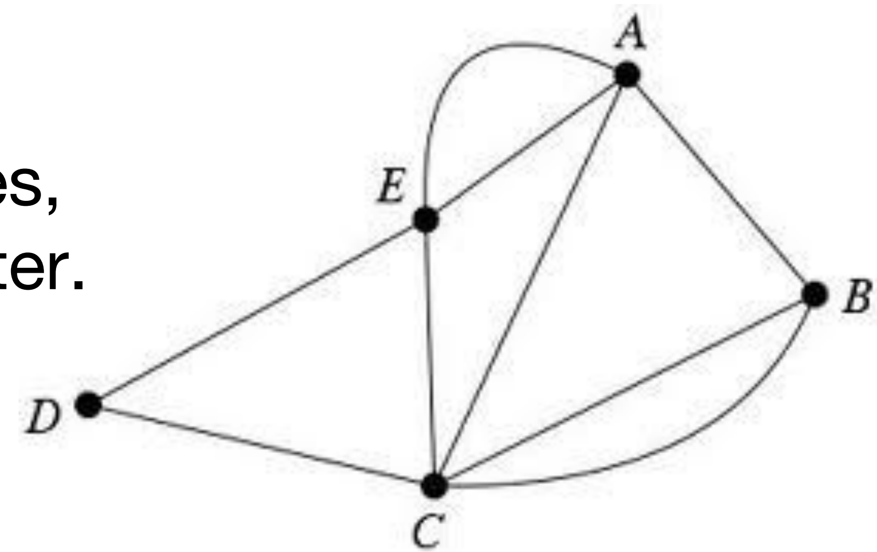
Programming With Data Structures

Introduction to Graphs



Graphs

- Similar to trees, graphs are made up of **vertices (nodes)** and **edges (links)** between those vertices.
- A **vertex** is referenced by a name / label, or index.
- An **edge** is referenced by a pair of vertices, such as (A, B), that it connects.
- Here an edge reflects the relationship between vertices, and its length does not matter.
- How is a graph different from a tree?

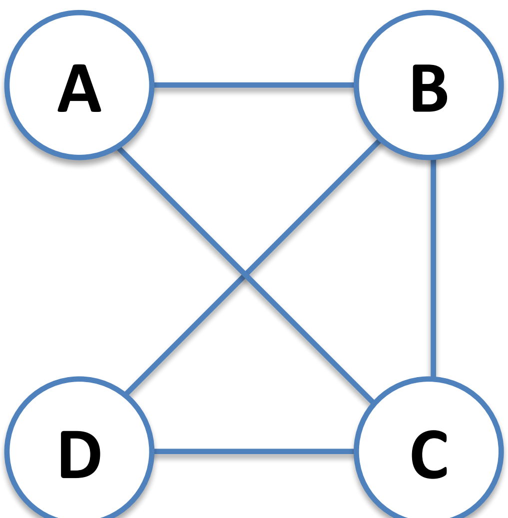


Formalism

- Mathematically, a graph G is defined as follows:

G = (V, E)

where **V** defines a set of vertices; **E** defines a set of edges (i.e. pairs of vertices)



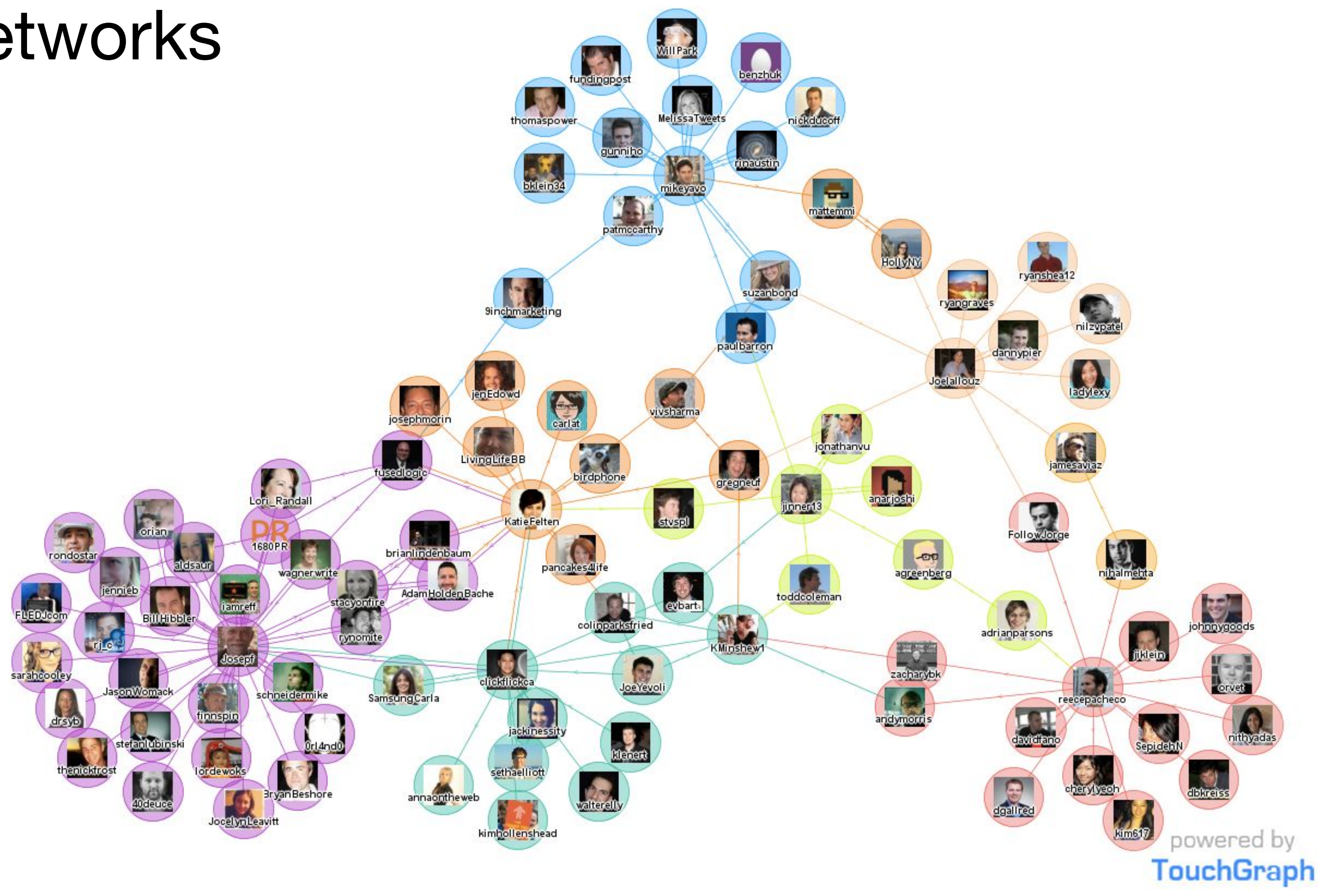

```
V = {A, B, C, D}

E = {(A,B),(A,C),(B,C),
     (B,D),(C,D)}
```

Graphs

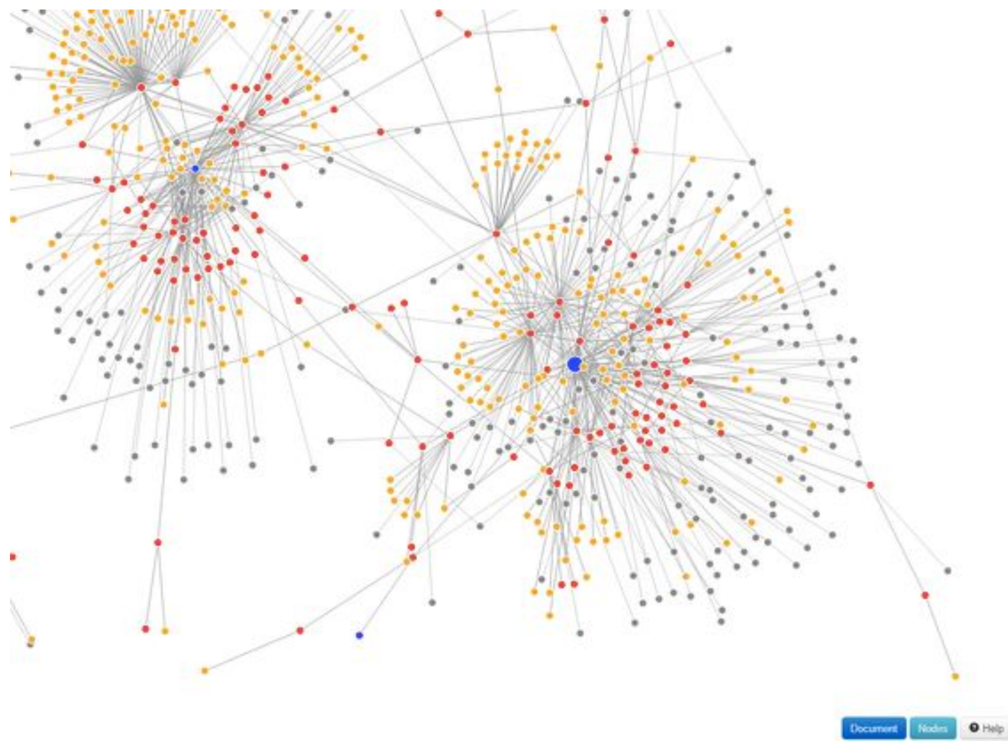
- Graphs are one of the most versatile data structures. It's useful for a lot of real-world applications.

- Social networks



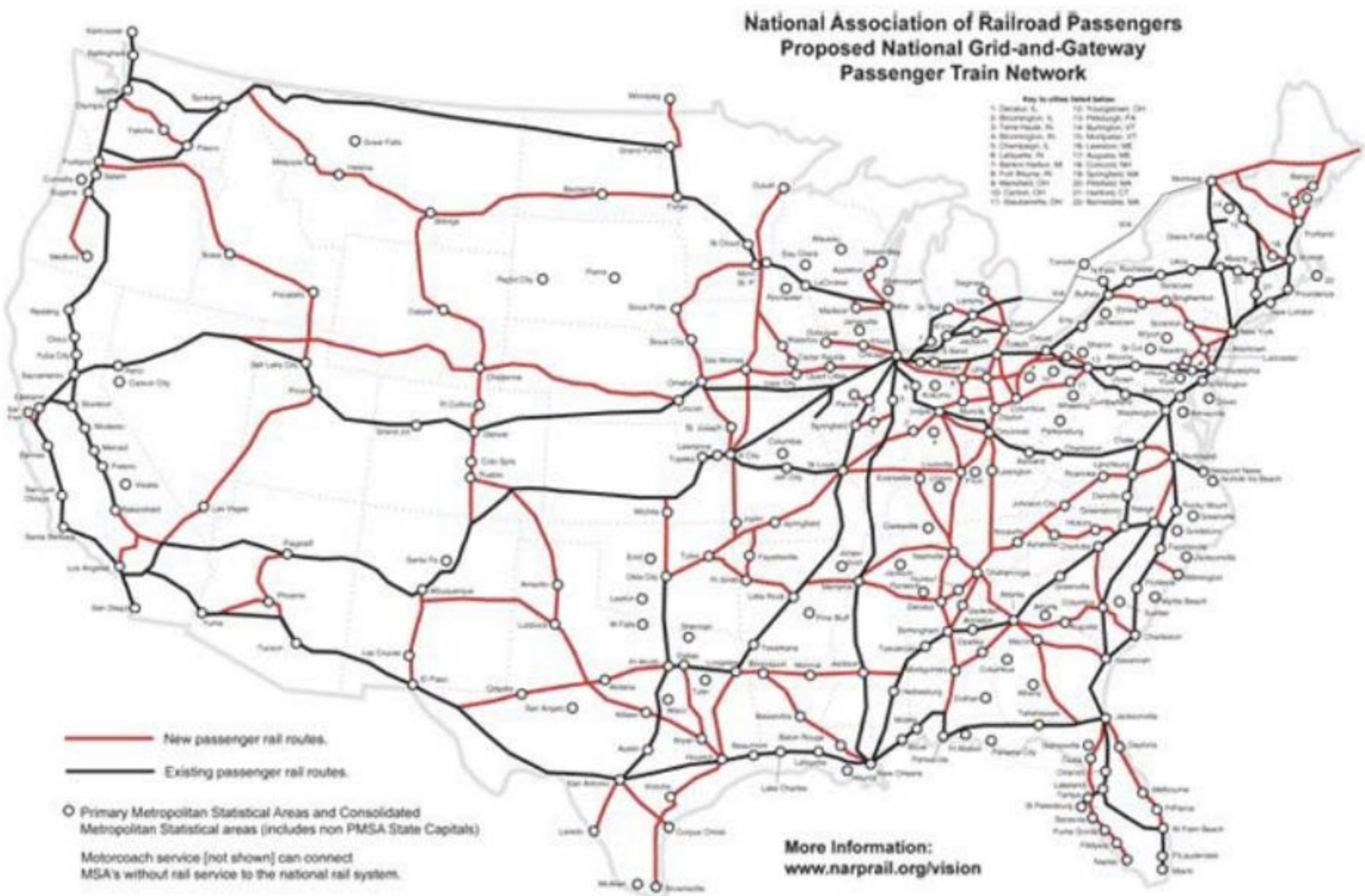
Graphs

- Links between Webpages / Websites
  - Ever thought about how Google page rank is calculated?



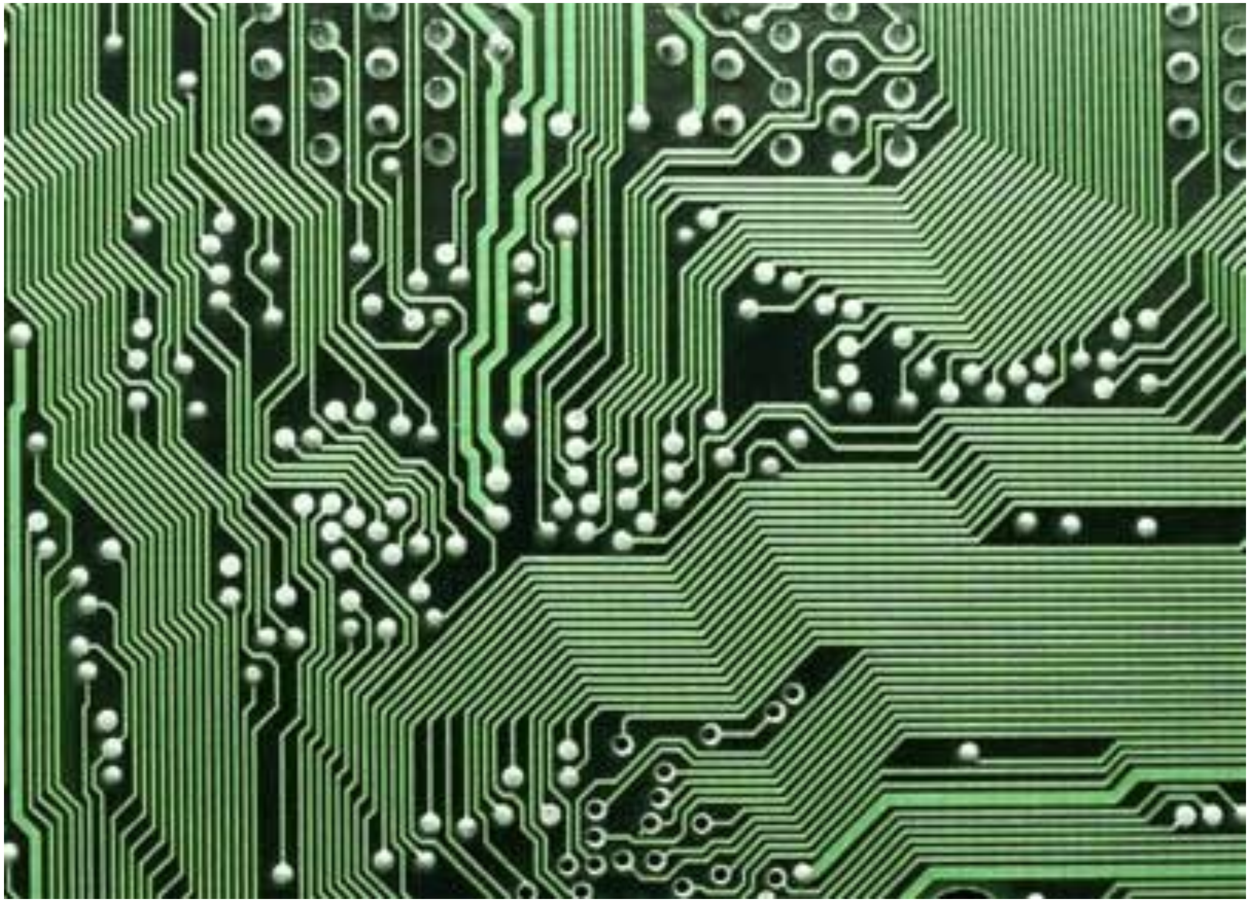
Graphs

- Roadmaps



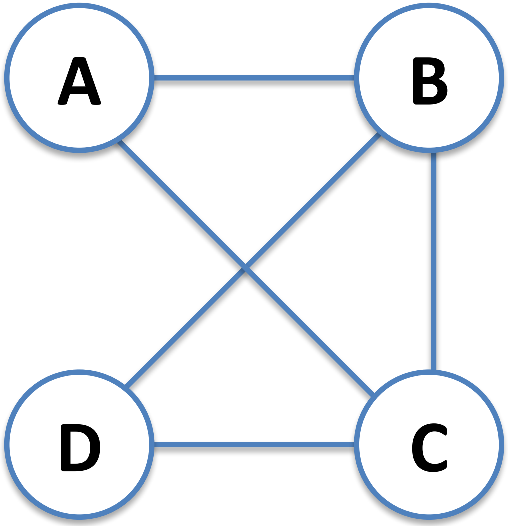
Graphs

- Circuit Board Design



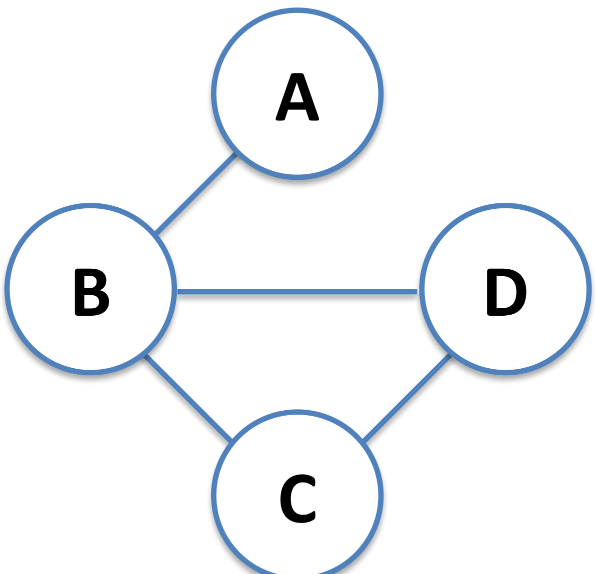
Graph Terminology

- **Adjacency**: two vertices are adjacent if they are directly connected by an edge.
- **Neighbors**: the set of vertices that are adjacent to a given vertex. The number of neighbors is called the **<u>valence</u>** (or degree) of the vertex.
- **Paths**: a sequence of edges that connect two vertices. Note that there may be multiple paths that connect two vertices!
    - Examples: A->C, A->B->C A->B->D->C

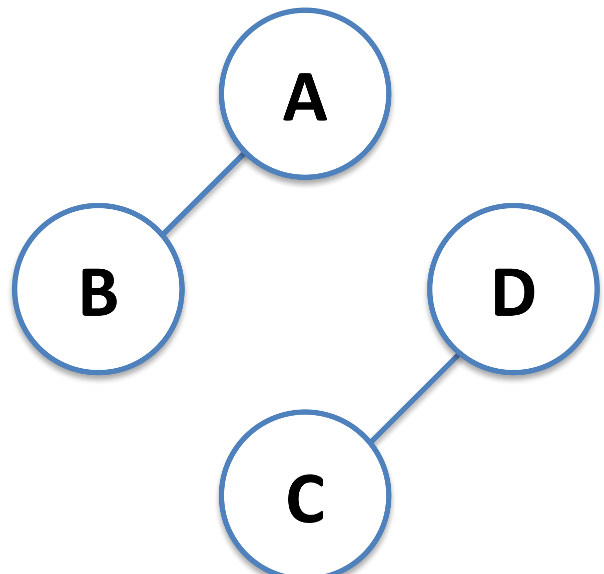


Graph Terminology

- **Connected Graph**: a graph is said to be connected if there exists **at least one path between <u>every pair</u> of vertices**.



Connected



Non-connected

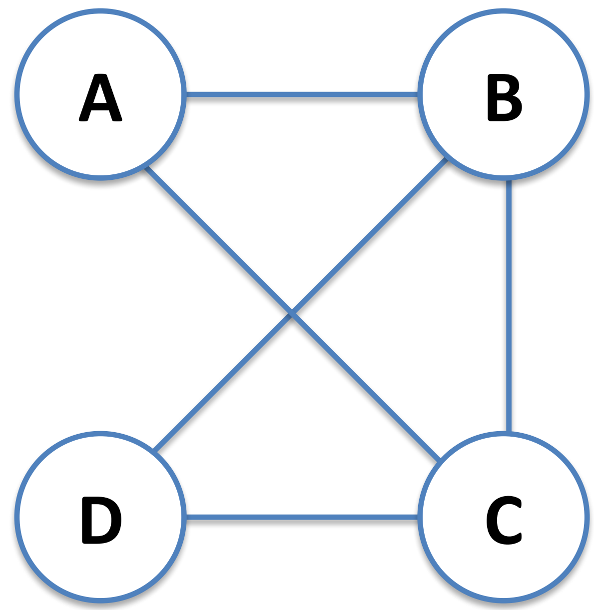
Undirected Graphs

The examples we covered so far assume undirected graphs.

Definition

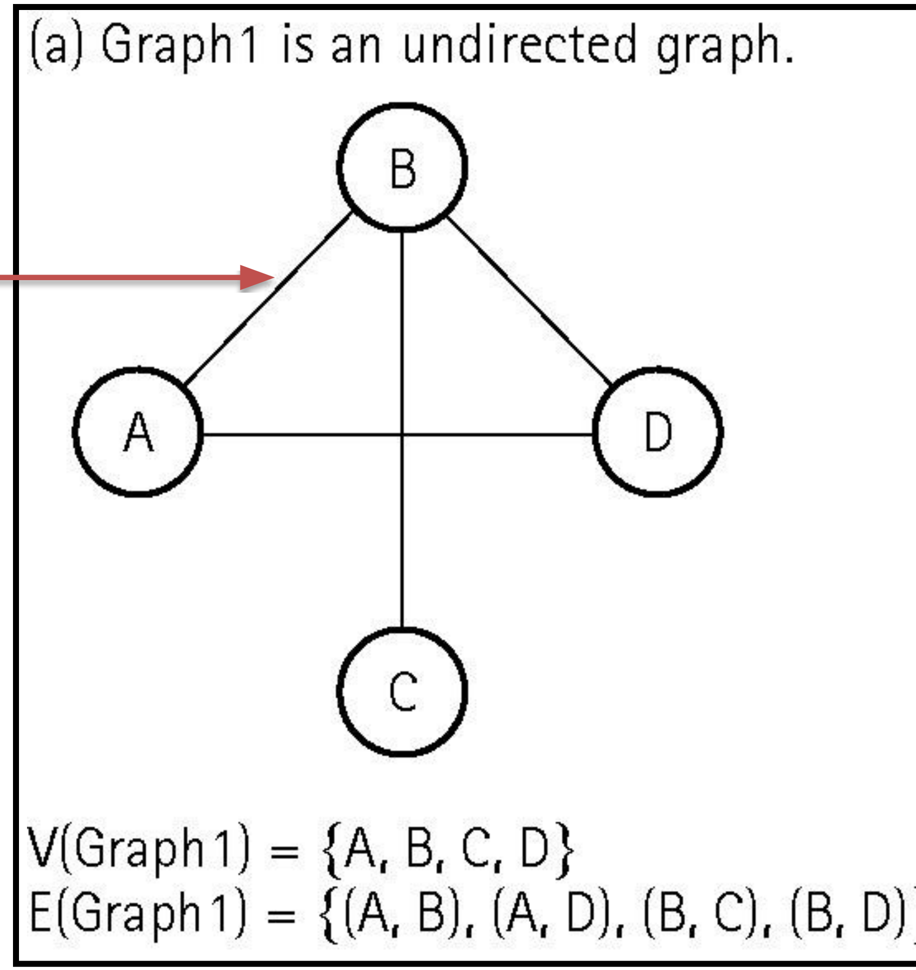
- An **undirected** graph is a graph where the pairings representing each edge are unordered
- That is, a graph in which the edges have no direction
- (A, B) and (B, A) refer to exactly the same edge

Example:



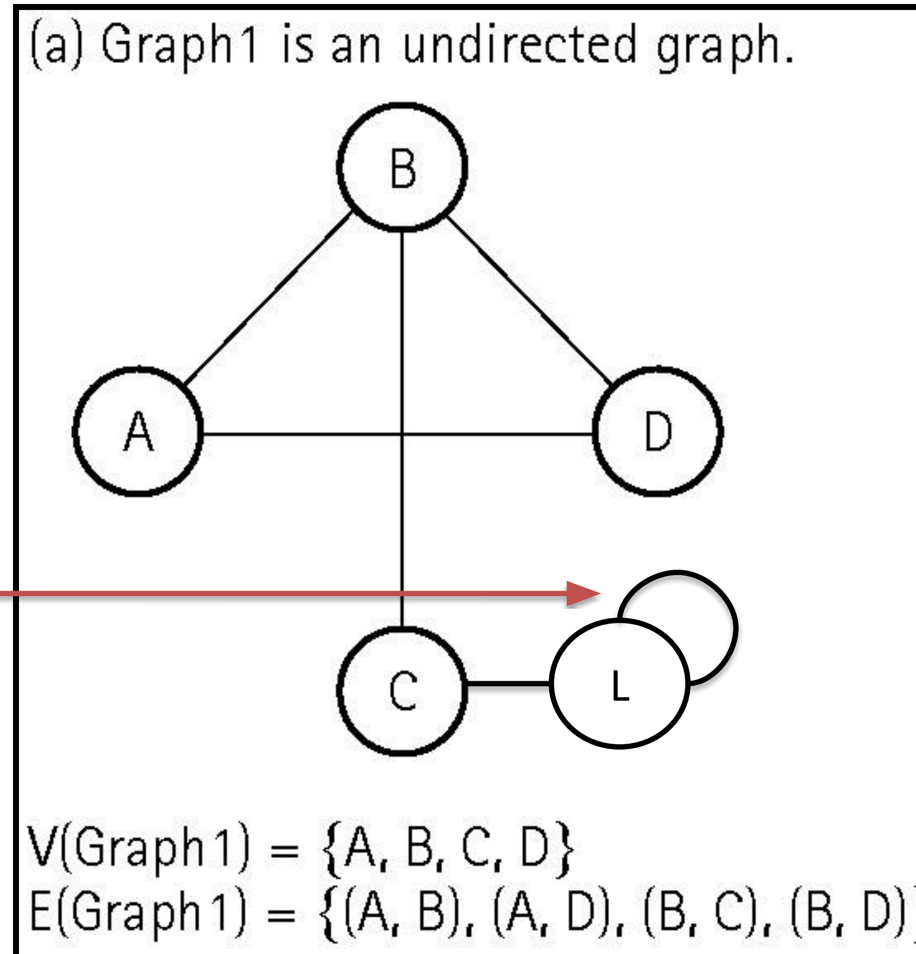
Undirected Graphs Details

- An edge in an undirected graph can be traversed in either direction
- You can go from A to B and vice versa



Undirected Graphs Details

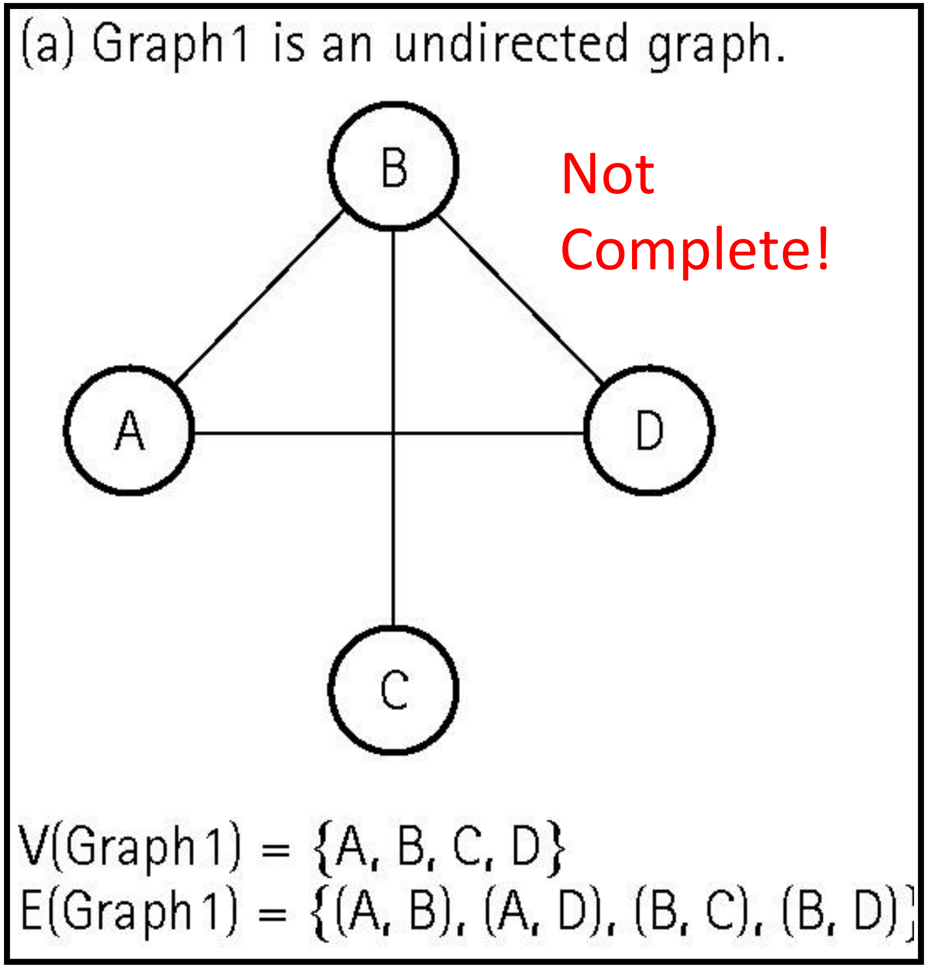
- An edge that connects a vertex to itself is called a <u>self-loop</u> or <u>buckle</u>
- In this course we stick to "<u>simple graphs</u>": no loops or multiple edges between nodes



+ {(C, L), (L, L)}

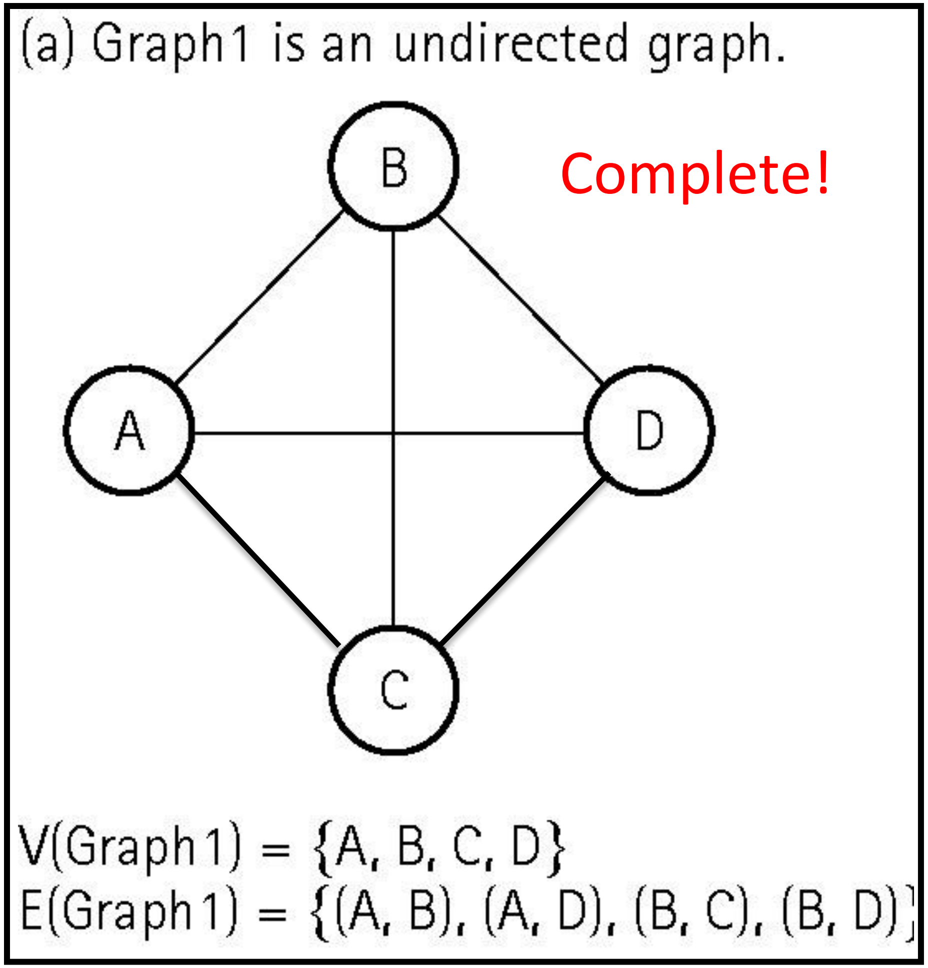
Undirected Graphs Details

- An undirected graph is considered <u>complete</u> if it has the maximum number of edges connecting vertices.
- In other words, every pair of vertices is connected by an edge.



Undirected Graphs Details

- An undirected graph is considered <u>complete</u> if it has the maximum number of edges connecting vertices
- Example: a complete graph with 4 vertices has a total of 6 edges



+ {(A, C), (C, D)}

Clicker Question #1

How many edges are there in a complete graph with **8** vertices?

(a) 6

(b) 15

(c) 20

(d) 21

(e) 28

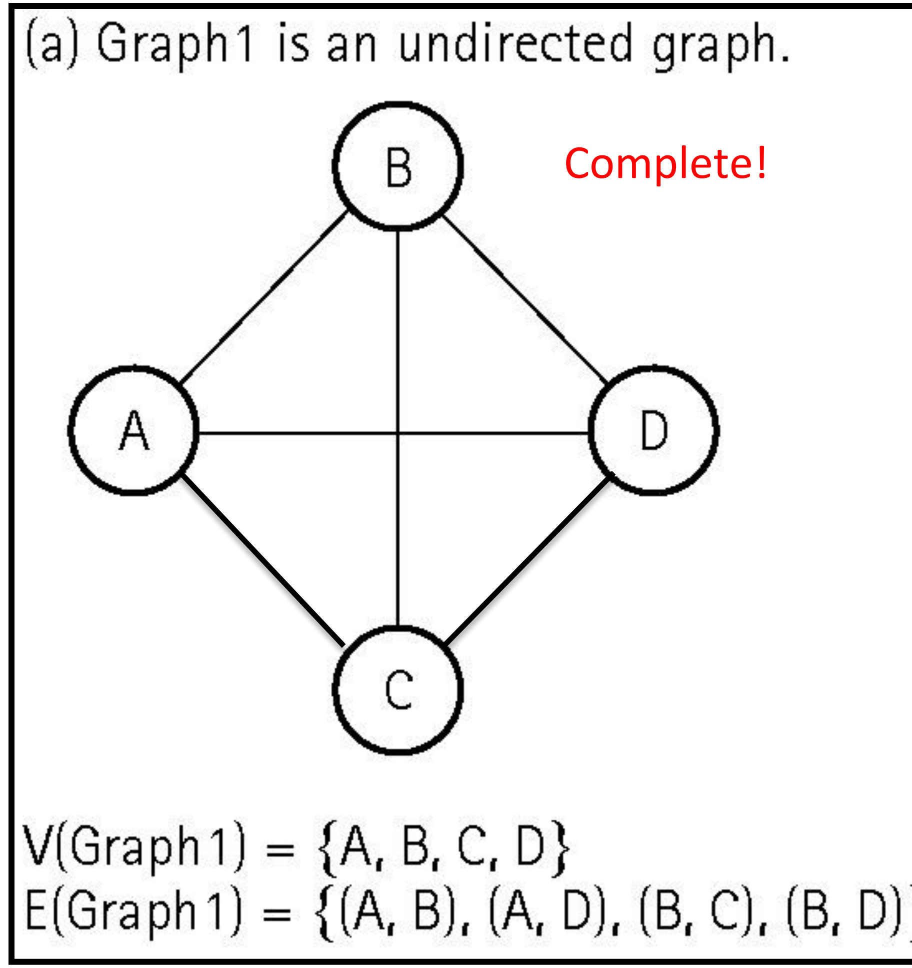
Answer on next slide

Complete Graphs

- Pop Quiz: how many edges are there in a complete graph containing N vertices?

(N-1)+(N-2)+(N-3)+...+1 =

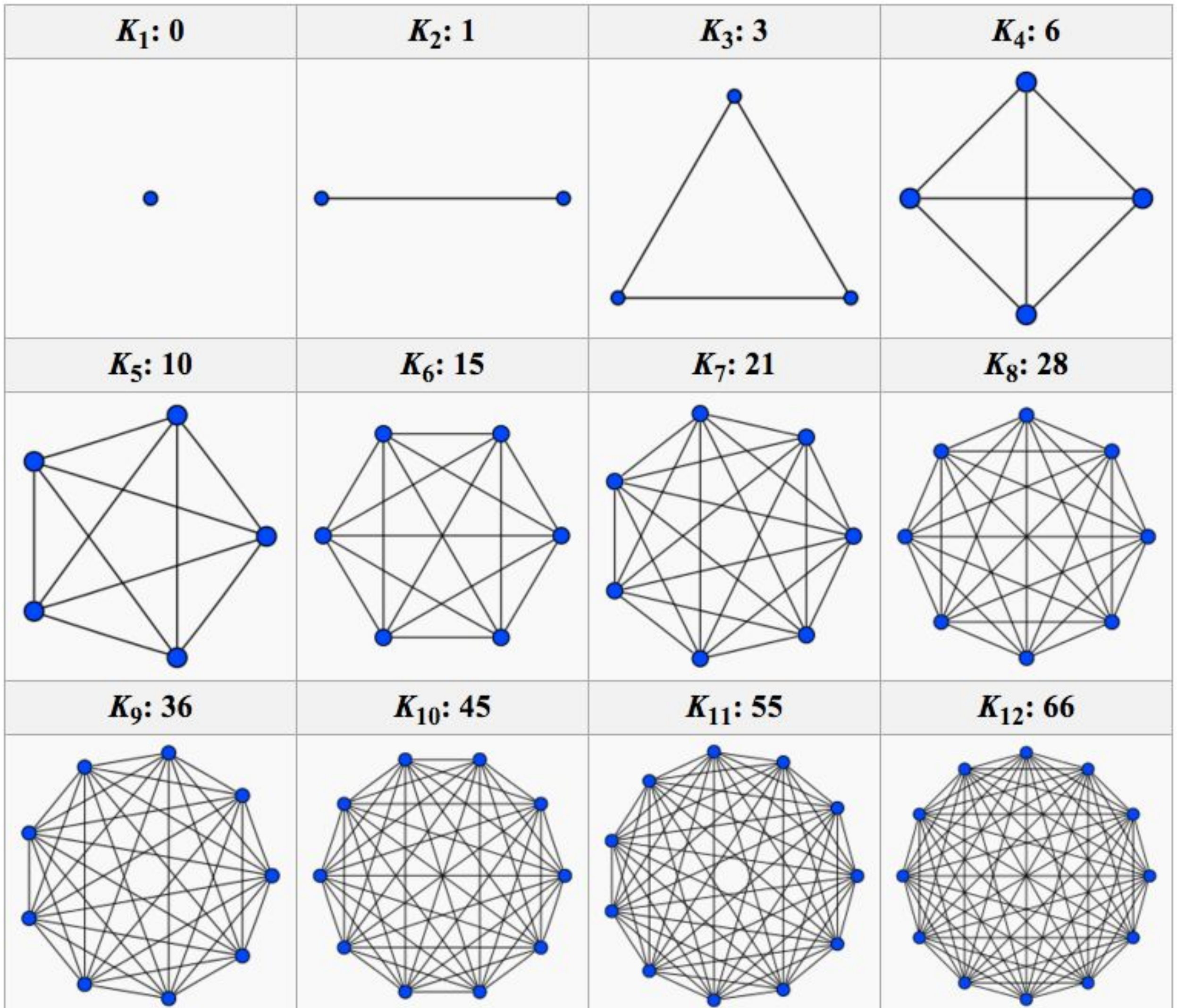
N(N-1)/2



+ {(A, C), (C, D)}

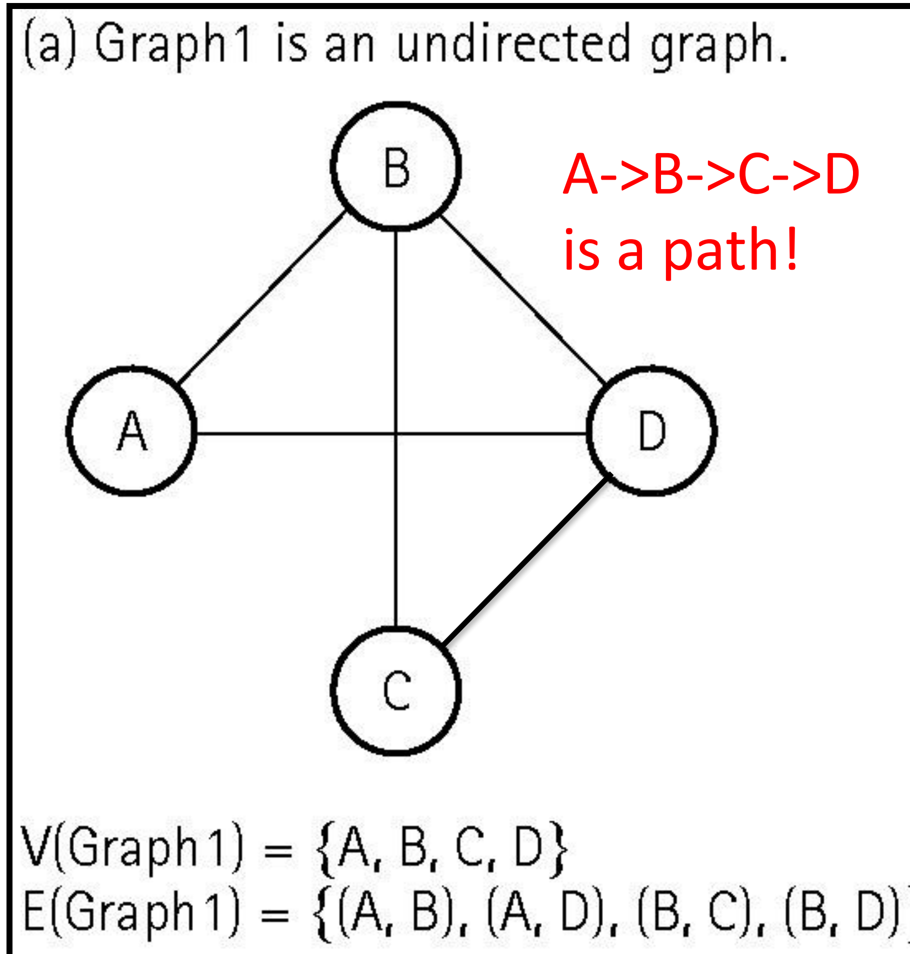
Complete Graphs

A complete graph with N nodes are often referred to as K_N



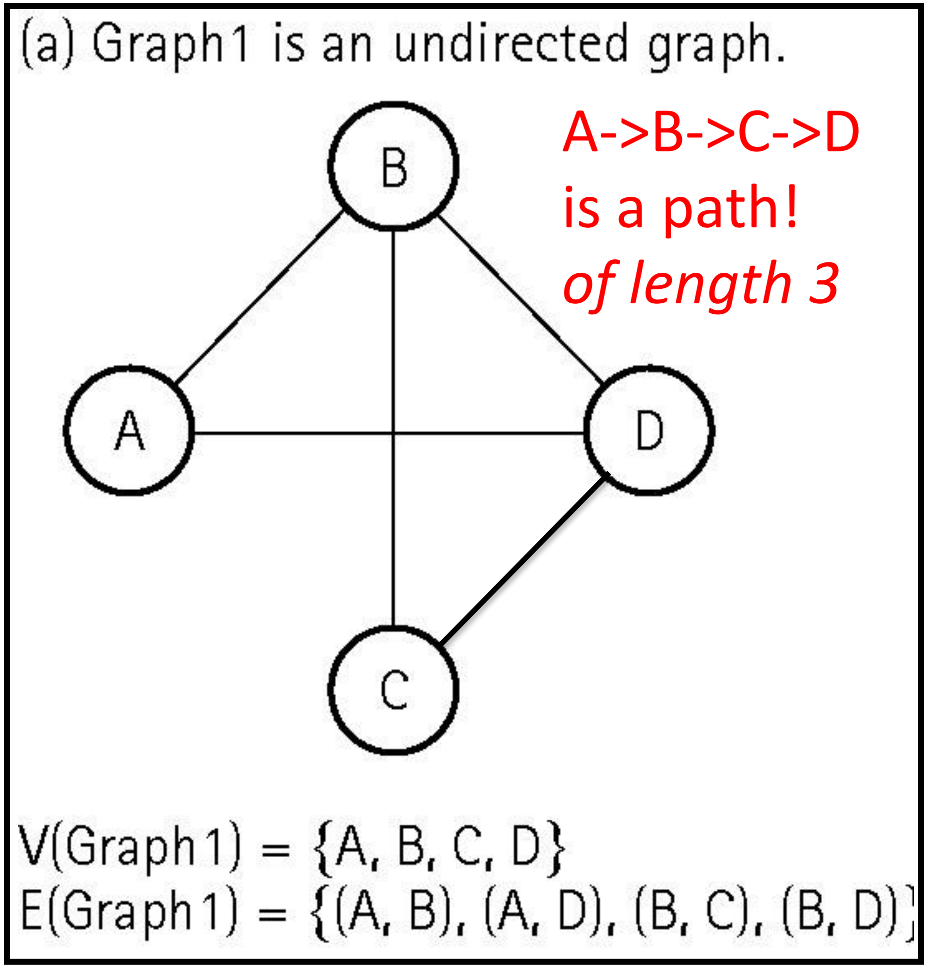
Undirected Graphs Details

- A **path** is a sequence of edges that connects two vertices in a graph
- A **simple path** contains no repeated vertices (i.e. does not cross over itself). For example, A->D->B->D->C is a non-simple path!
- We assume 'paths' refer to simple paths.



Undirected Graphs Details

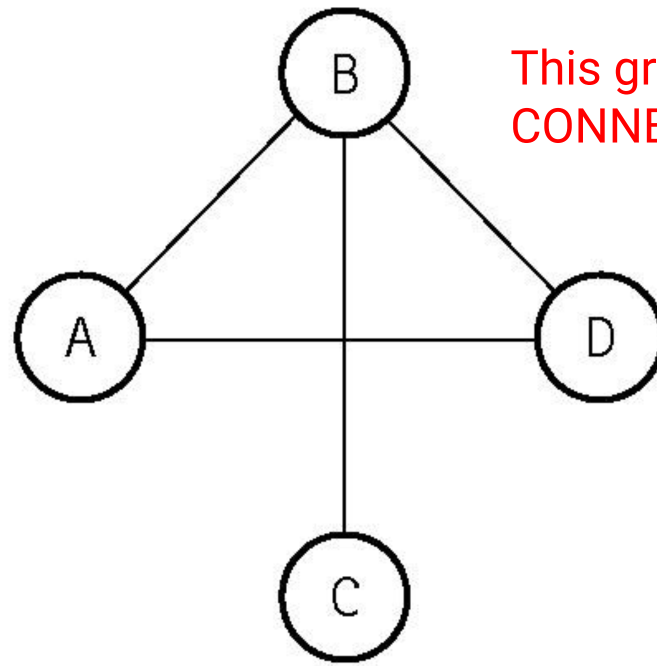
- The length of a path is the number of edges in the path (or the number of vertices minus 1)



Undirected Graphs Details

- An undirected graph is considered <u>connected</u> if for any two vertices in the graph there is a path between them

(a) Graph1 is an undirected graph.



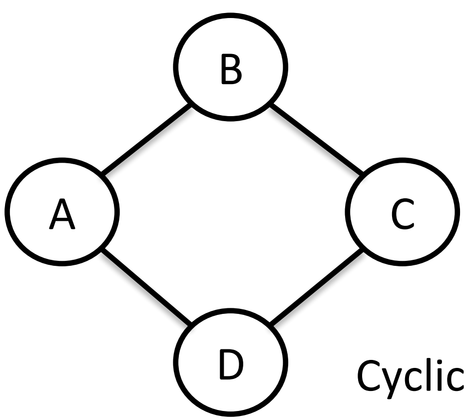
This graph is CONNECTED

V(Graph1) = {A, B, C, D}

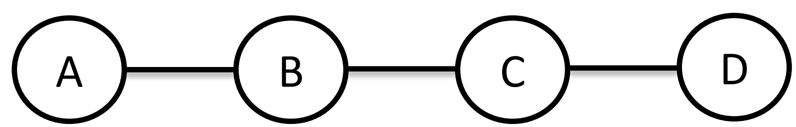
E(Graph1) = {(A, B), (A, D), (B, C), (B, D)}

Cycles

- A *cycle* is a path in which the first and last vertices are the same and none of the edges are repeated
- A graph that has no cycles is called acyclic



Cyclic



Acyclic

Directed Graphs

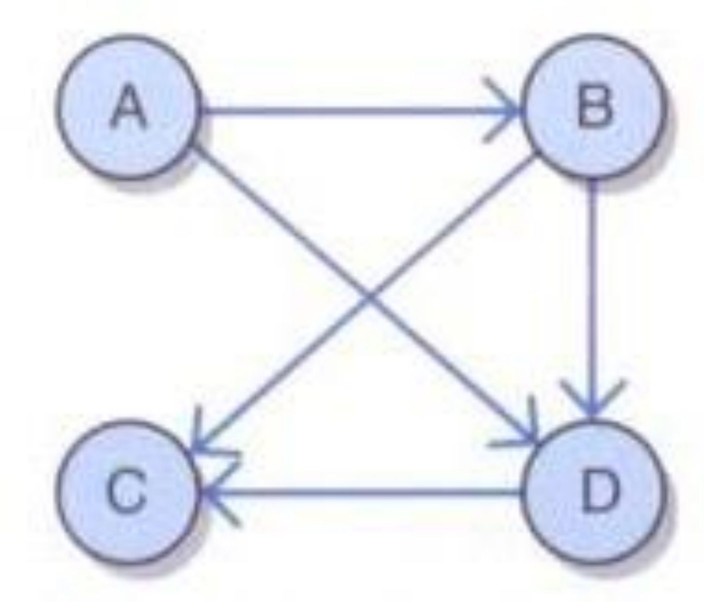
- A **directed graph**, sometimes referred to as a *digraph*, is a graph where the edges are **ordered** pairs of vertices

- In other words, a graph in which each edge is *directed* (visually, it has an arrow).

- This means that edges (A, B) and (B, A) are **different** edges!

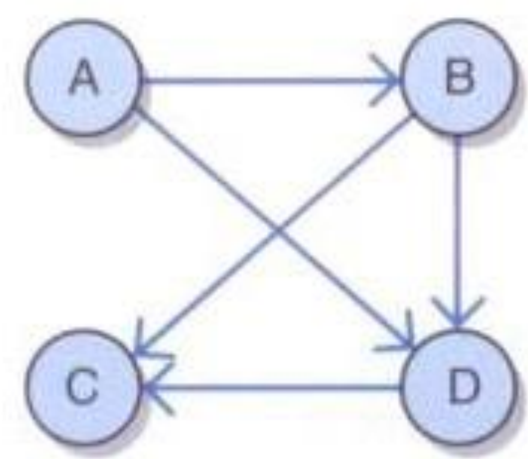
Directed Graph Example

- A directed graph with
  - Vertices A, B, C, D
  - Edges (A, B), (A, D), (B, C), (B, D), and (D, C)



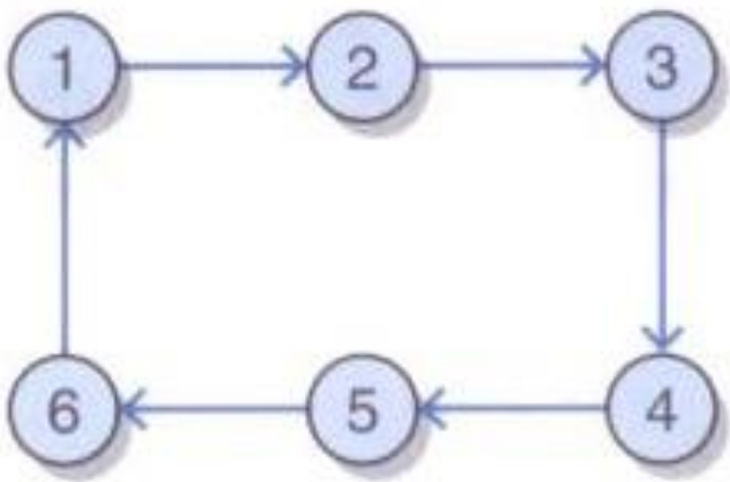
Directed Graphs Definitions

- Previous definitions change slightly for directed graphs
  - A **path** from vertex 1 to vertex 2 in a directed graph is a sequence of directed edges that connect vertex 1 to vertex 2.
  - A directed graph is **connected** if for every pair of ordered vertices there is a path between them.
- Is there a path from A to C?
- Is there a path from C to A?
- Is this directed graph a connected graph?

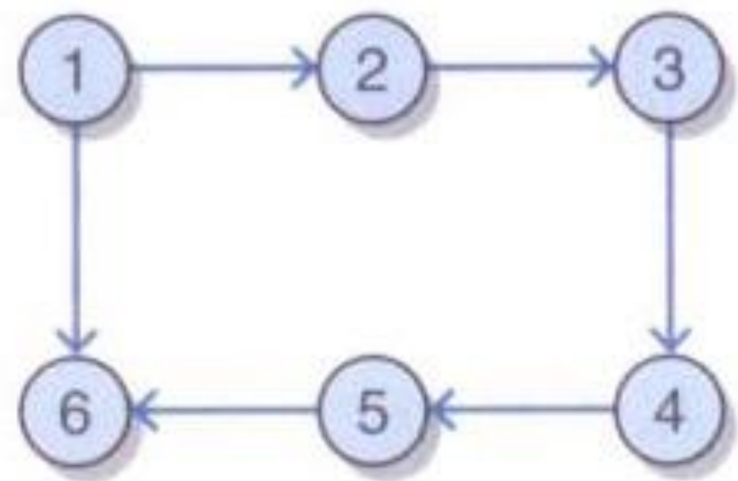


Connected Directed Graphs

- A connected directed graph and a non-connected directed graph:



connected



unconnected

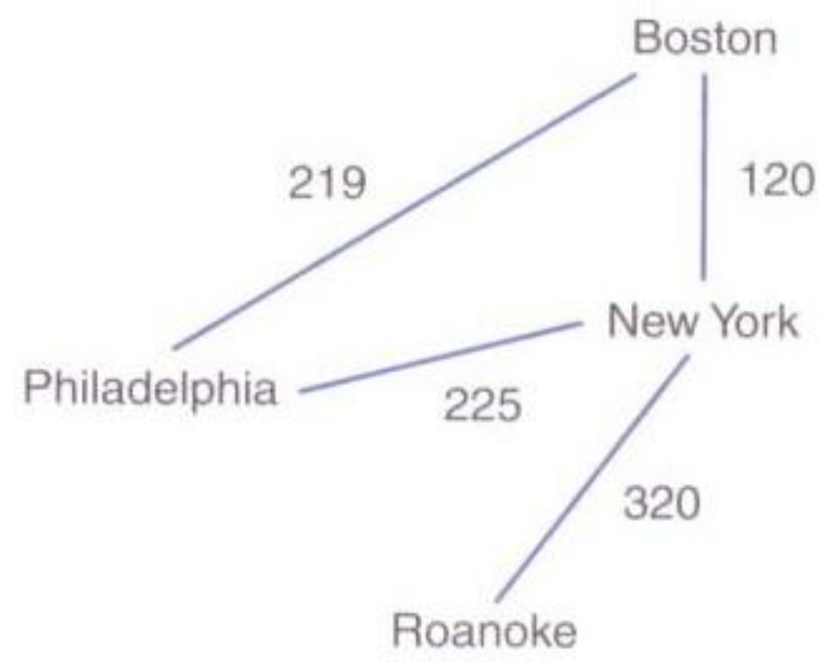
Don’t Confuse the Terms

- **Directed Graph**: edges are directional (visually arrows). (A, B) is different from (B, A).
- **Undirected Graph**: edges are bi-directional (visually no arrows). (A, B) is the same as (B, A). Think of it as a special case of directed graph where each edge is by default bi-directional.
- **Connected Graph**: there exists a <u>path</u> between every pair of vertices. Otherwise, it’s non-connected graph.
- The combinations: connected directed graph, connected undirected graph, non-connected directed graph etc.

Weighted Graphs

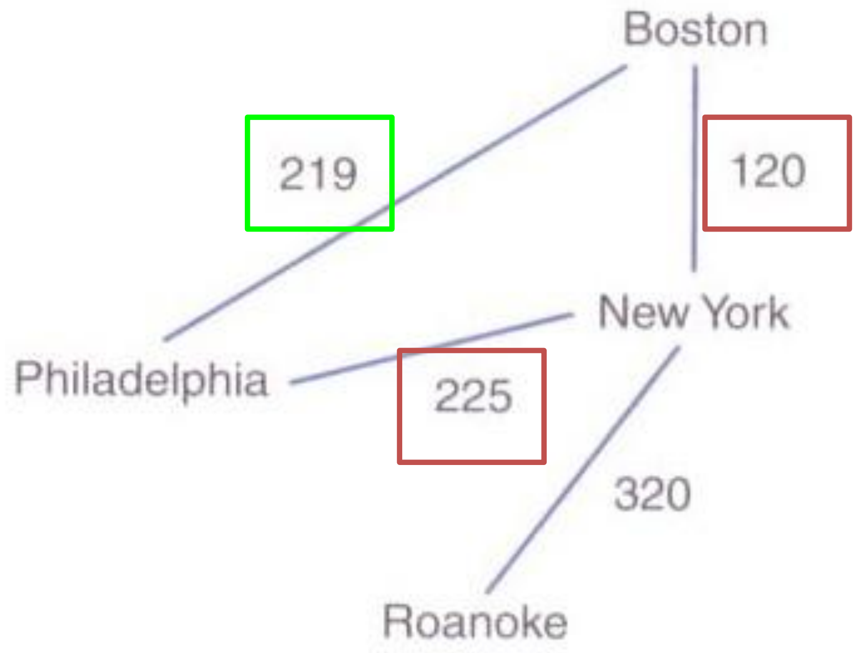
- A <u>weighted graph</u> is a graph with weights (or costs) associated with each edge.

- Imagine the graph on the right shows the airline price between every two connected cities.



Weighted Graphs

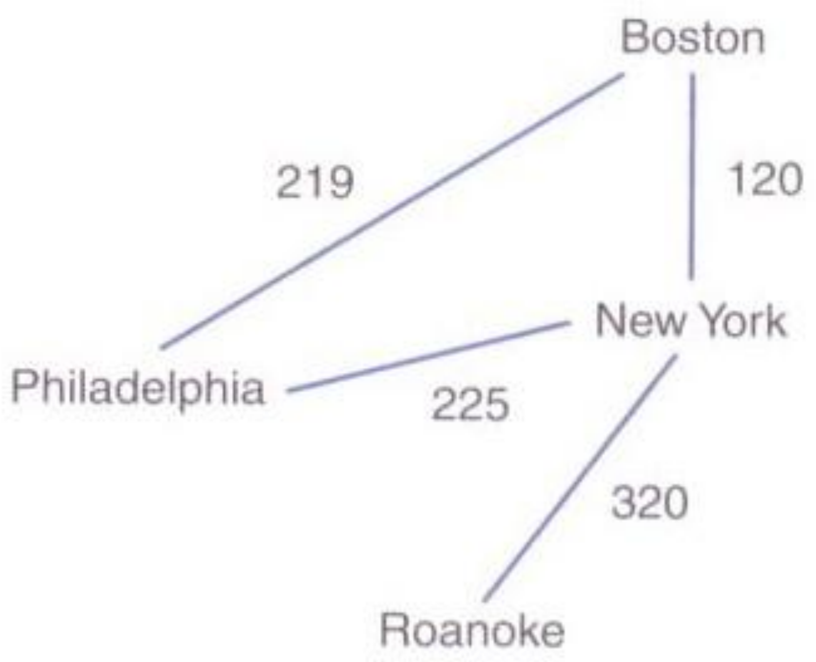
- If there is a path between two vertices, the path weight is the sum of the weights of the edges in the path.
- As you may have multiple paths, each path may have a different weight.
- In many cases, you may want the path with the minimum / maximum weight.



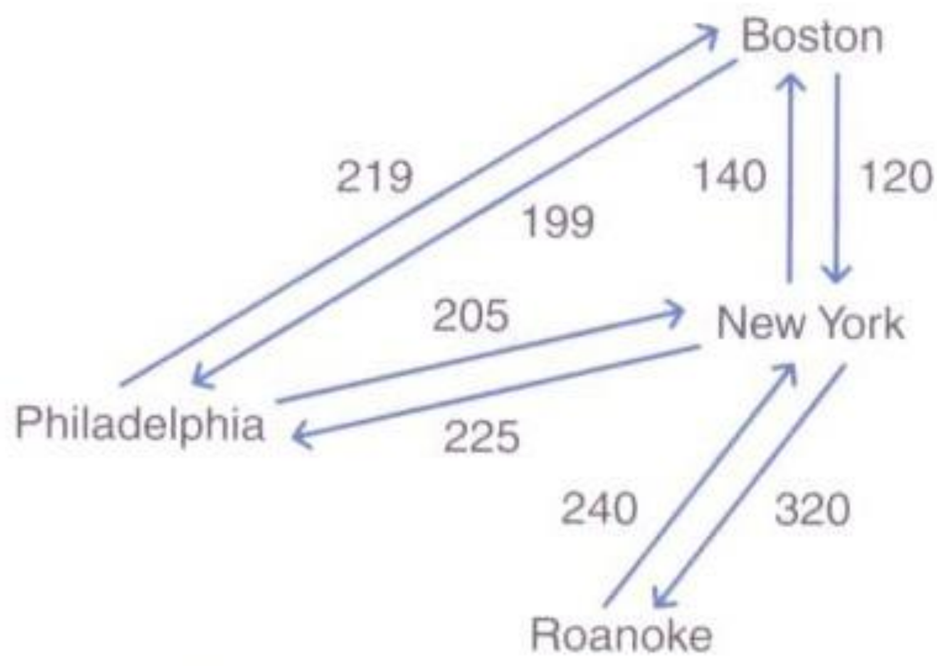
The path from Boston to Philly through NYC:
120 + 225 = 345

Weighted Graphs

- Weighted graphs may be either undirected or directed
- On a directed graph, the weight may be different depending on the direction (e.g. airline price may not be symmetric).



Undirected Weighted Graph



Directed Weighted Graph

Clicker Question #2

Say you have found the shortest path in a directed graph between two nodes, s and t. If you increase the **weight/cost of every edge** in the graph by 1, does that path always remain the shortest path between s and t?

(a) Yes

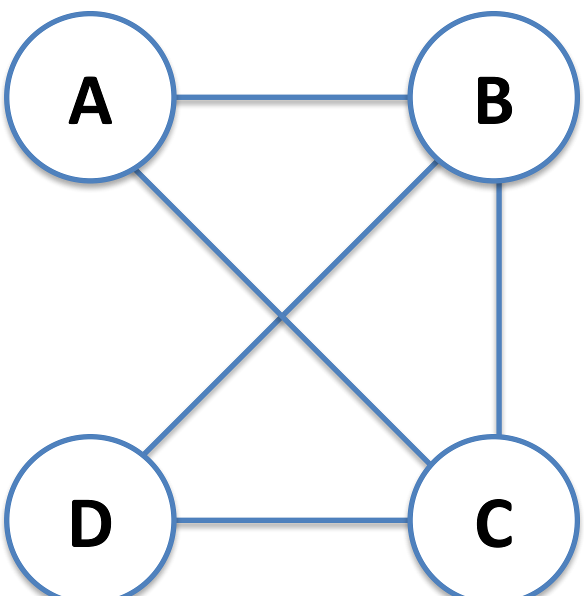
(b) No

Answer on next slide

Representing Graphs

- **Adjacency Matrix**
  For a graph with N nodes, its adjacency matrix is an `N x N` table that shows the existence (and weights if weighted) of all edges in the graph.

- Example of <u>unweighted, undirected</u> graph. (The matrix is **binary and symmetric**)



to

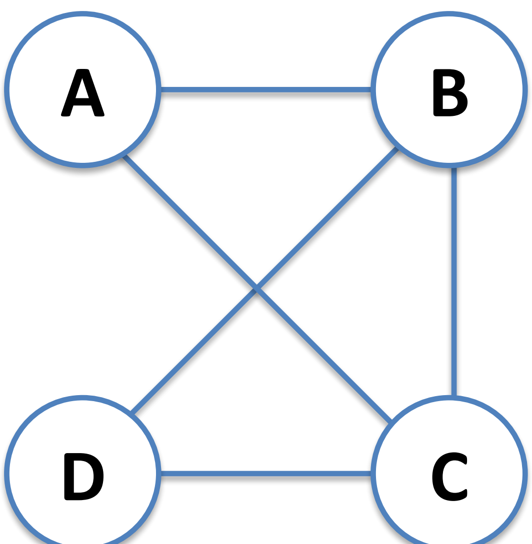
from

	A	*B*	*C*	*D*
A	0	1	1	0
B	1	0	1	1
C	1	1	0	1
D	0	1	1	0

Representing Graphs

The adjacency matrix captures all the edge information. From it, you can calculate, for example:

- The number of neighbors each vertex has. How?
- Is there a path between two vertices? If so, what's the path length?
- Is this a connected graph?



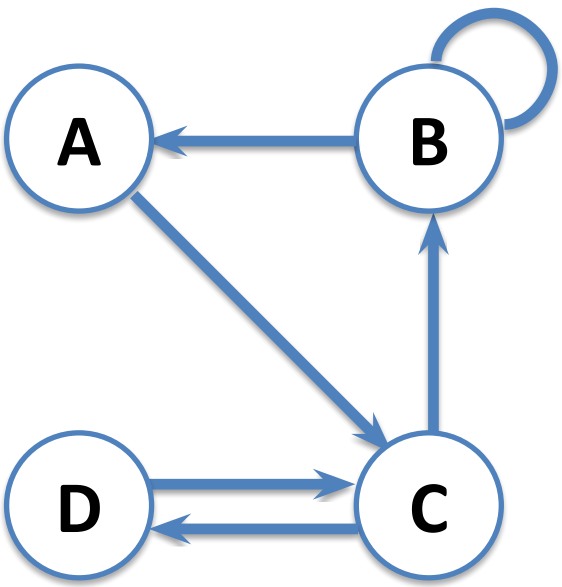
to

from

	A	B	C	D
A	0	1	1	0
B	1	0	1	1
C	1	1	0	1
D	0	1	1	0

Representing Graphs

- **Adjacency Matrix**
- Example of unweighted, directed graph. (The matrix is binary and generally non-symmetric)



to

from

	A	*B*	*C*	*D*
A	0	0	1	0
B	1	1	0	0
C	0	1	0	1
D	0	0	1	0

Clicker Question #3

Given the adjacency matrix of a directed graph, is there a path <u>from A to B</u>? If so, what's the shortest path length?

(a) Yes, path length is 1

(b) Yes, path length is 2

(c) Yes, path length is 3

(d) Yes, path length is 4

(e) No

to

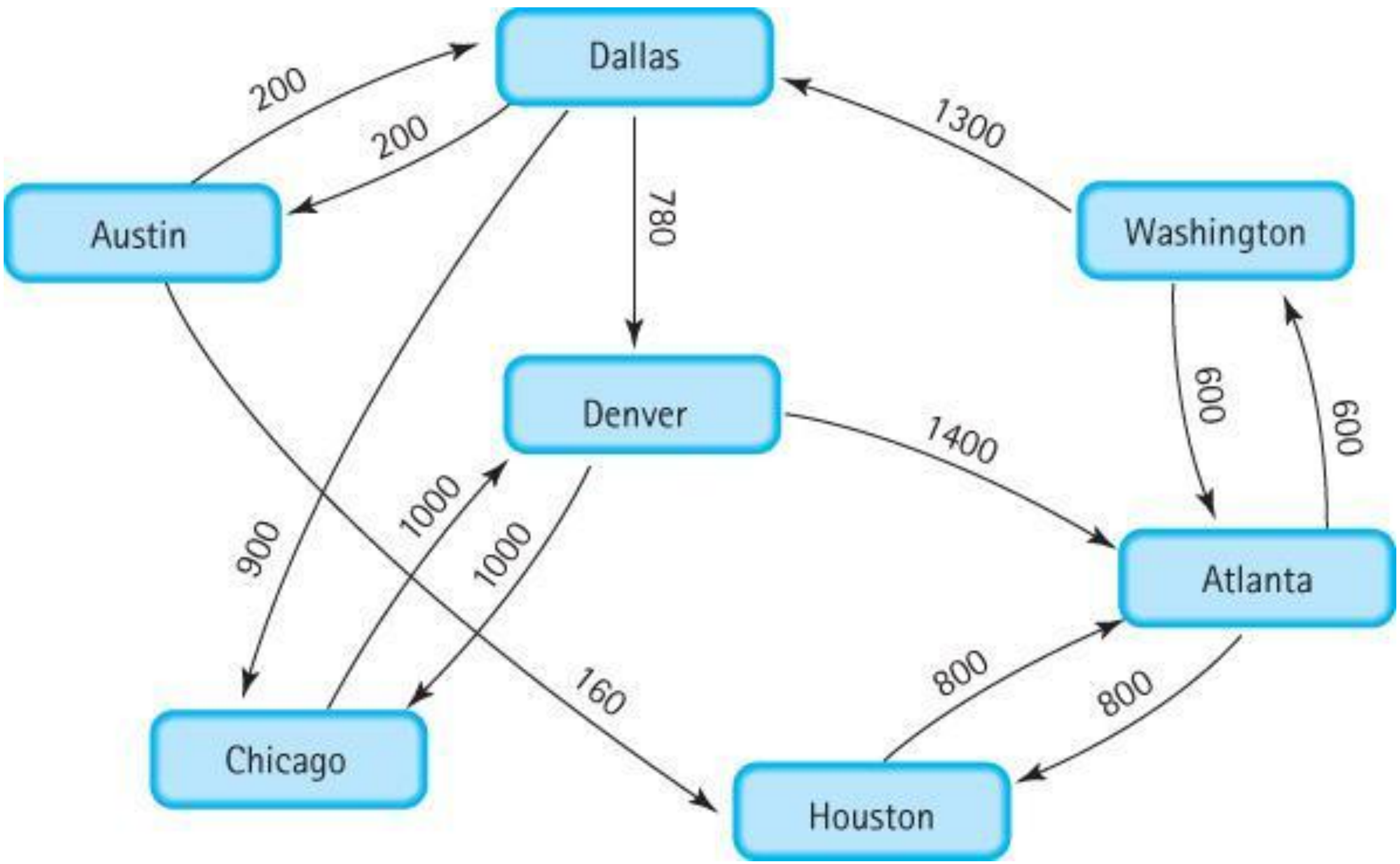
from

	A	***B***	***C***	***D***
A	0	0	1	1
B	0	0	0	1
C	1	1	1	1
D	0	0	0	0

Answer on next slide

Representing Graphs

- Example of <u>weighted, directed</u> graph.



Adjacency Matrix

graph
.numVertices 7
.vertices

[0]	"Atlanta "
[1]	"Austin "
[2]	"Chicago "
[3]	"Dallas "
[4]	"Denver "
[5]	"Houston "
[6]	"Washington"
[7]	
[8]	
[9]	

.edges

	[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]
[0]	0	0	0	0	0	800	600	•	•	•
[1]	0	0	0	200	0	160	0	•	•	•
[2]	0	0	0	0	1000	0	0	•	•	•
[3]	0	200	900	0	780	0	0	•	•	•
[4]	1400	0	1000	0	0	0	0	•	•	•
[5]	800	0	0	0	0	0	0	•	•	•
[6]	600	0	0	1300	0	0	0	•	•	•
[7]	•	•	•	•	•	•	•	•	•	•
[8]	•	•	•	•	•	•	•	•	•	•
[9]	•	•	•	•	•	•	•	•	•	•

(Array positions marked '•' are undefined)

Adjacency Matrix

Distance (weight) from Dallas to Denver?

graph

.numVertices 7

.vertices

[0]	"Atlanta "
[1]	"Austin "
[2]	"Chicago "
[3]	"Dallas "
[4]	"Denver "
[5]	"Houston "
[6]	"Washington"
[7]	
[8]	
[9]	

.edges

	[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]
[0]	0	0	0	0	0	800	600	•	•	•
[1]	0	0	0	200	0	160	0	•	•	•
[2]	0	0	0	0	1000	0	0	•	•	•
[3]	0	200	900	0	780	0	0	•	•	•
[4]	1400	0	1000	0	0	0	0	•	•	•
[5]	800	0	0	0	0	0	0	•	•	•
[6]	600	0	0	1300	0	0	0	•	•	•
[7]	•	•	•	•	•	•	•	•	•	•
[8]	•	•	•	•	•	•	•	•	•	•
[9]	•	•	•	•	•	•	•	•	•	•

(Array positions marked '•' are undefined)

Adjacency Matrix

Distance (weight) from Dallas to Denver?
780

graph
.numVertices 7
.vertices

[0]	"Atlanta "
[1]	"Austin "
[2]	"Chicago "
[3]	"Dallas "
[4]	"Denver "
[5]	"Houston "
[6]	"Washington"
[7]	
[8]	
[9]	

.edges

	[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]
[0]	0	0	0	0	0	800	600	•	•	•
[1]	0	0	0	200	0	160	0	•	•	•
[2]	0	0	0	0	1000	0	0	•	•	•
[3]	0	200	900	0	780	0	0	•	•	•
[4]	1400	0	1000	0	0	0	0	•	•	•
[5]	800	0	0	0	0	0	0	•	•	•
[6]	600	0	0	1300	0	0	0	•	•	•
[7]	•	•	•	•	•	•	•	•	•	•
[8]	•	•	•	•	•	•	•	•	•	•
[9]	•	•	•	•	•	•	•	•	•	•

(Array positions marked '•' are undefined)

Adjacency Matrix

Distance (weight) from Austin to Houston?

graph
.numVertices 7
.vertices

[0]	"Atlanta "
[1]	"Austin "
[2]	"Chicago "
[3]	"Dallas "
[4]	"Denver "
[5]	"Houston "
[6]	"Washington"
[7]	
[8]	
[9]	

.edges

	[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]
[0]	0	0	0	0	0	800	600	•	•	•
[1]	0	0	0	200	0	160	0	•	•	•
[2]	0	0	0	0	1000	0	0	•	•	•
[3]	0	200	900	0	780	0	0	•	•	•
[4]	1400	0	1000	0	0	0	0	•	•	•
[5]	800	0	0	0	0	0	0	•	•	•
[6]	600	0	0	1300	0	0	0	•	•	•
[7]	•	•	•	•	•	•	•	•	•	•
[8]	•	•	•	•	•	•	•	•	•	•
[9]	•	•	•	•	•	•	•	•	•	•

(Array positions marked '•' are undefined)

Adjacency Matrix

Distance (weight) from Austin to Houston?
160

graph
.numVertices 7
.vertices

[0]	"Atlanta "
[1]	"Austin "
[2]	"Chicago "
[3]	"Dallas "
[4]	"Denver "
[5]	"Houston "
[6]	"Washington"
[7]	
[8]	
[9]	

.edges

	[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]
[0]	0	0	0	0	0	800	600	•	•	•
[1]	0	0	0	200	0	160	0	•	•	•
[2]	0	0	0	0	1000	0	0	•	•	•
[3]	0	200	900	0	780	0	0	•	•	•
[4]	1400	0	1000	0	0	0	0	•	•	•
[5]	800	0	0	0	0	0	0	•	•	•
[6]	600	0	0	1300	0	0	0	•	•	•
[7]	•	•	•	•	•	•	•	•	•	•
[8]	•	•	•	•	•	•	•	•	•	•
[9]	•	•	•	•	•	•	•	•	•	•

(Array positions marked '•' are undefined)