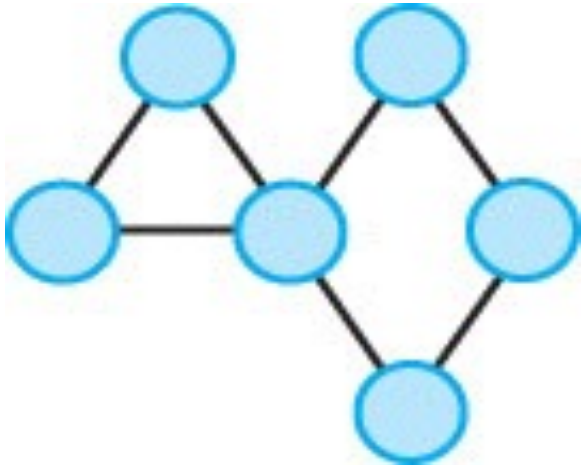
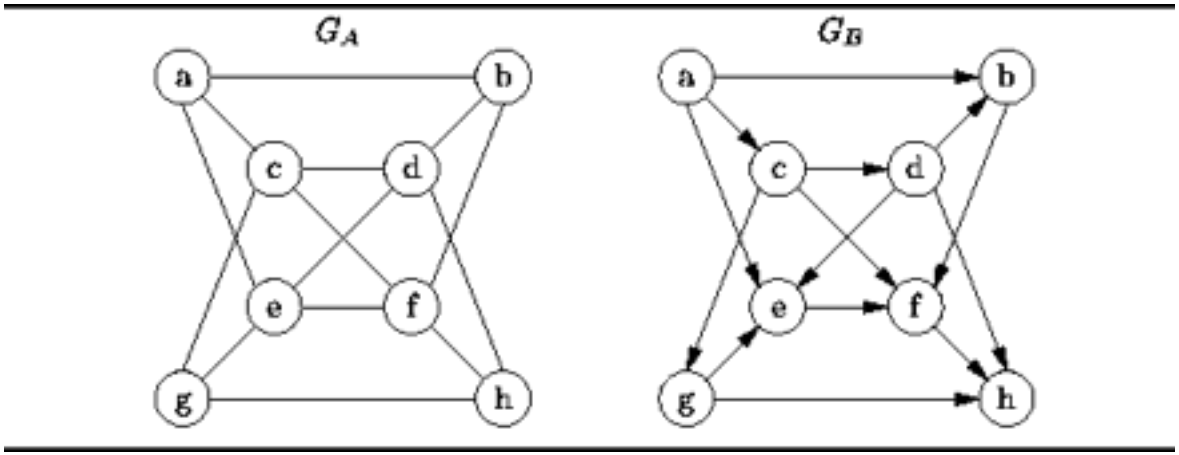


Programming With Data Structures

Introduction to Graphs



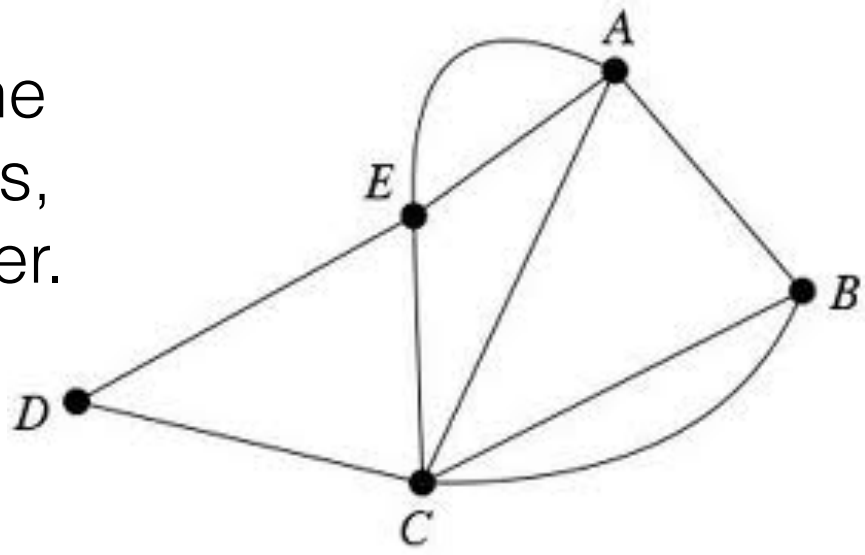
Reminders and Topics

- **Project 8 due this Friday 4pm**

- This and next lectures:
  - **Graphs**
  - **Representation of Graphs**
  - **Search in 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 the pair of vertices, such as (A, B), that it connects.
- Note that an edge reflects the relationship between vertices, and its length does not matter.
- How is a graph different from a tree?

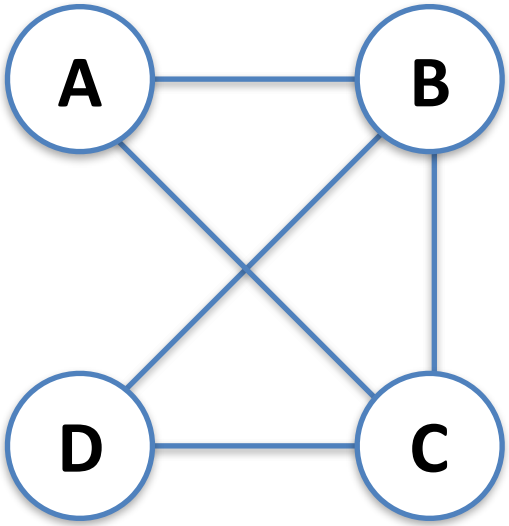


Formalism

- Mathematically, a graph G is defined as follows:

  G = (V, E)

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




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

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

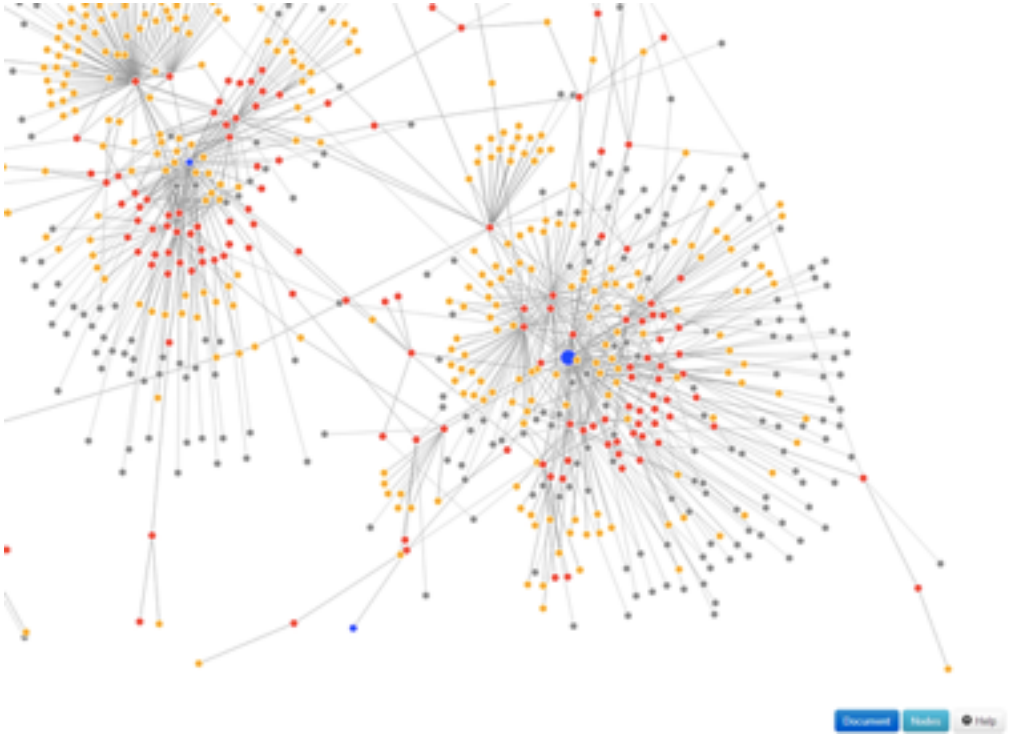
Graphs

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

- Social networks

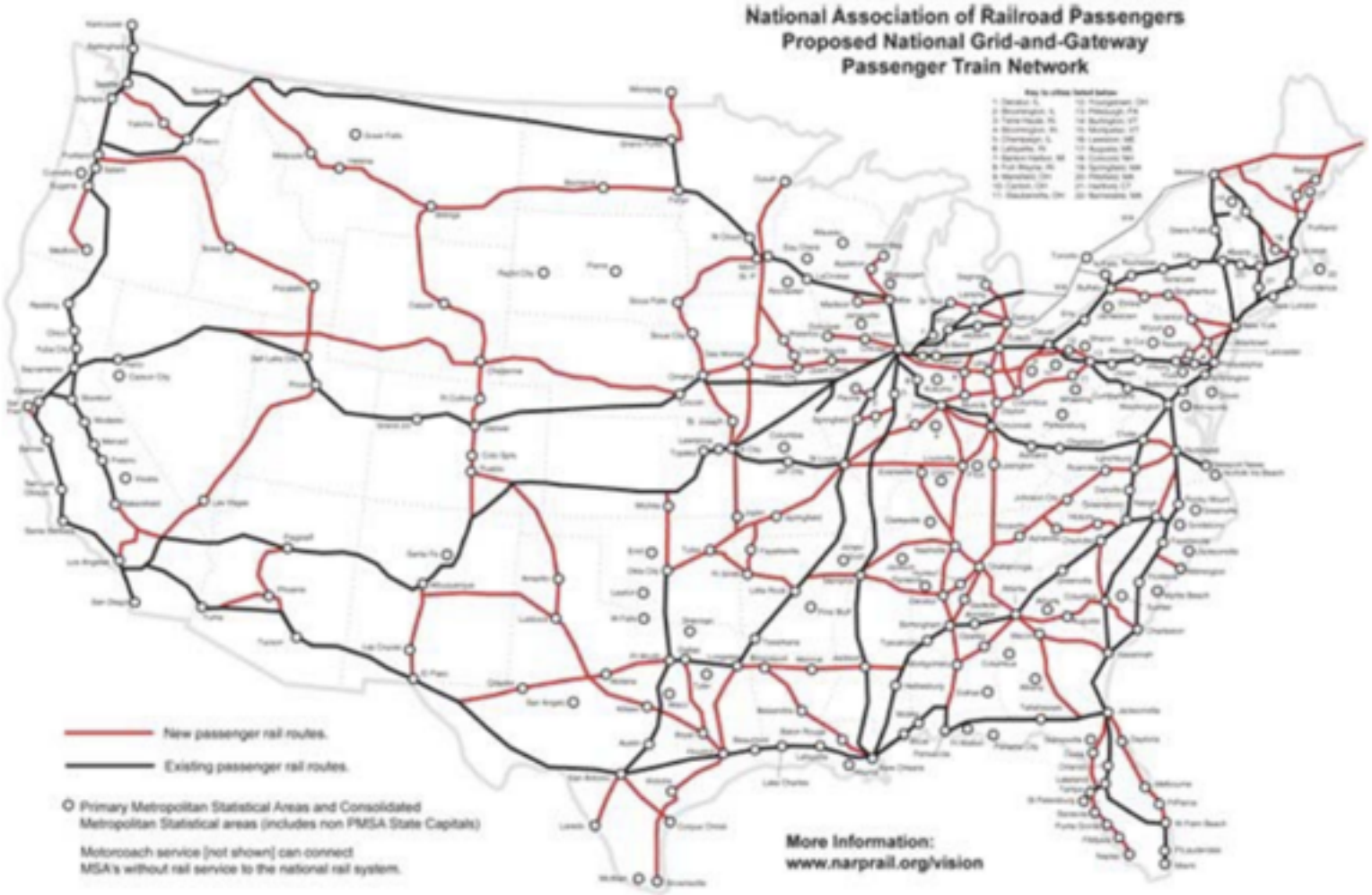
Graphs

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



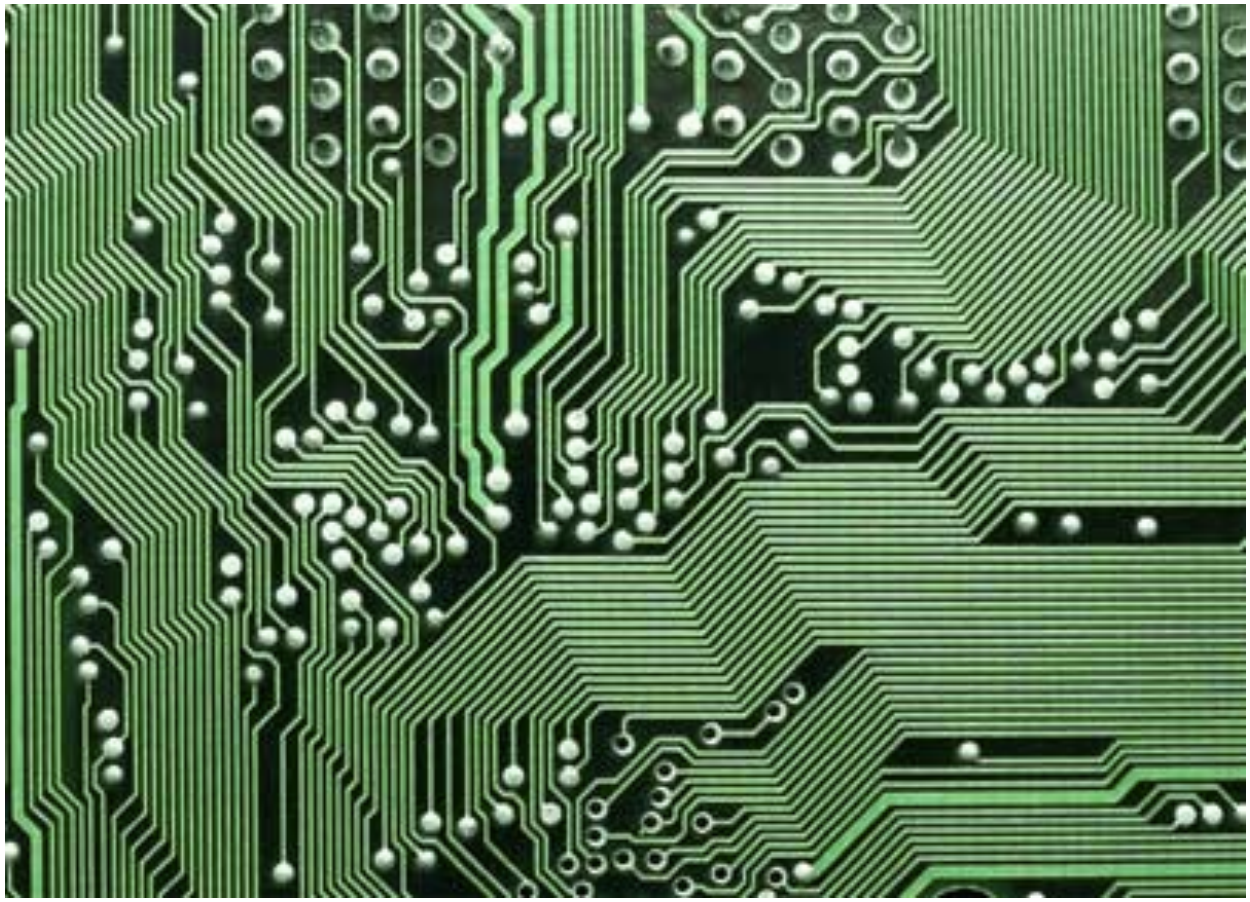
Graphs

- Roadmaps



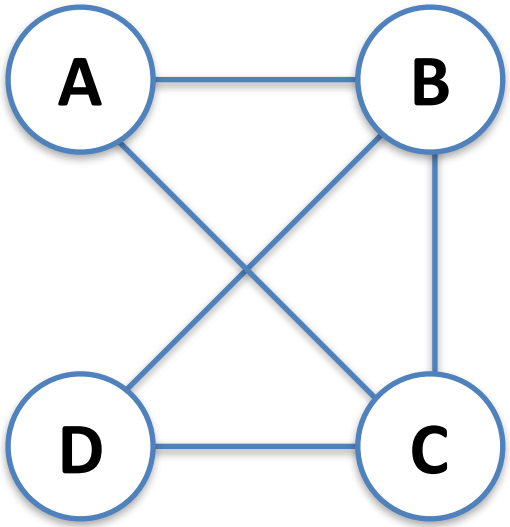
Graphs

- Circuit Board Design



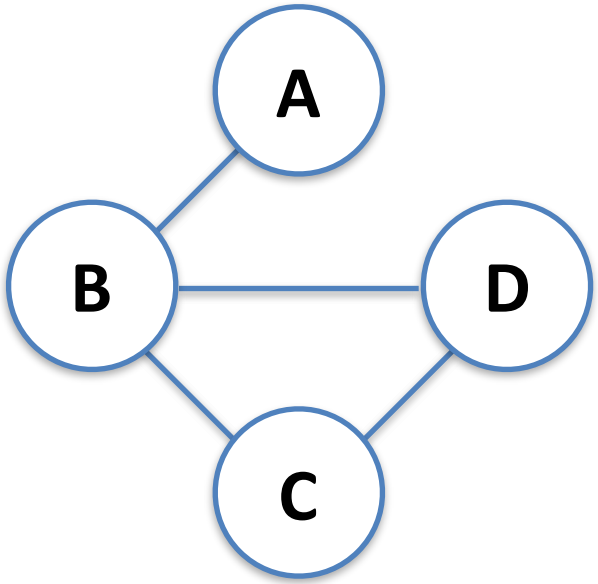
Graph Terminology

- **Adjacency**: two vertices are adjacent if they are connected by an edge.
- **Neighbors**: the set of vertices that are adjacent to a given vertex. The number of neighbors is called the **valence** 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

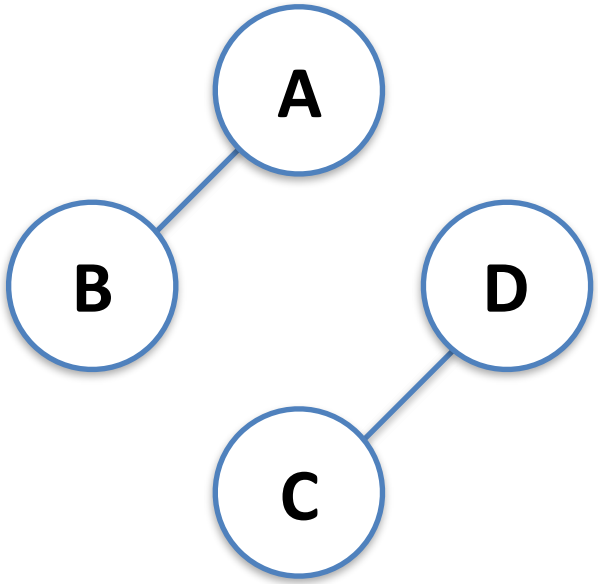


Graph Terminology

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



Connected



Non-connected

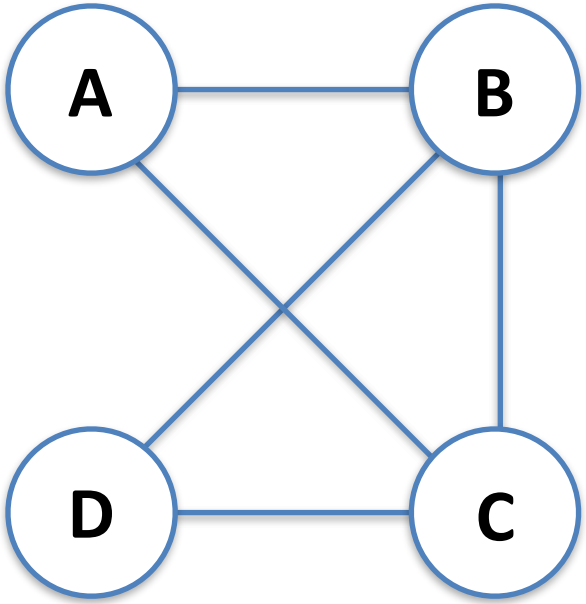
Undirected Graphs

The examples we covered so far assume undirected graphs.

Definition

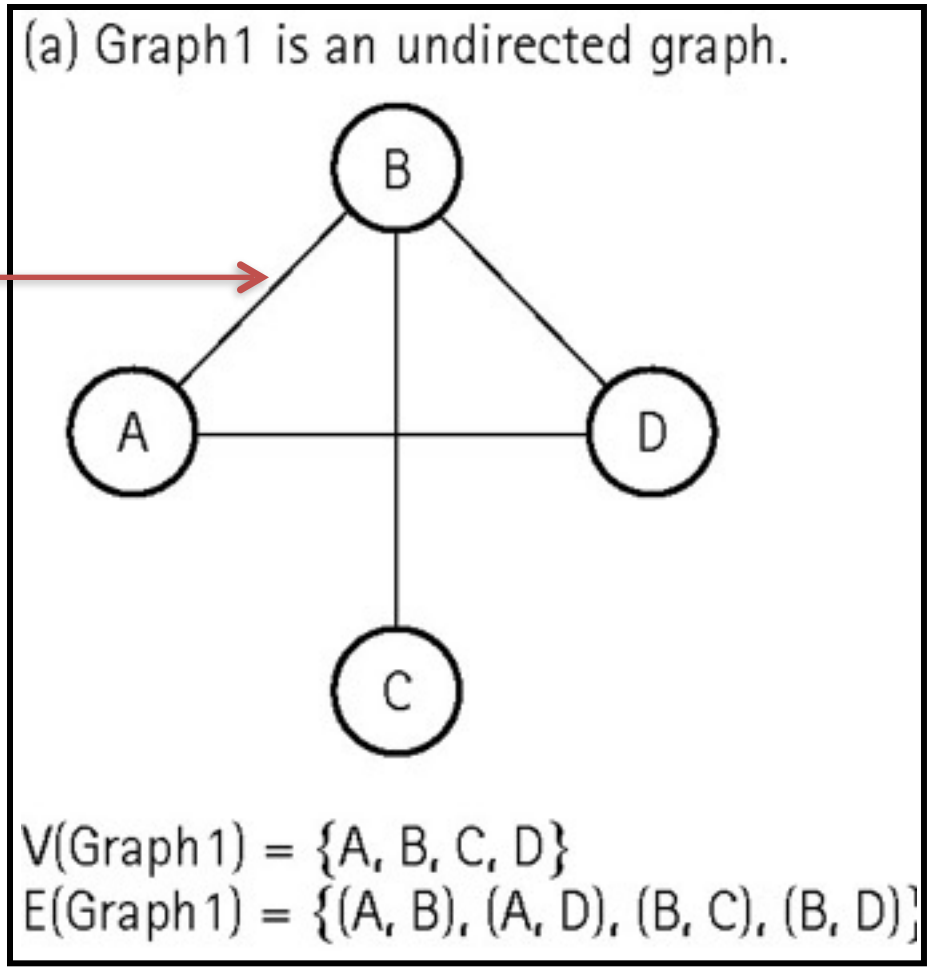
- An **undirected** graph is a graph where the pairings representing the edges are unordered
- That is, a graph in which the edges have no direction
- (A, B) and (B, A) refer to the same edge

Example:



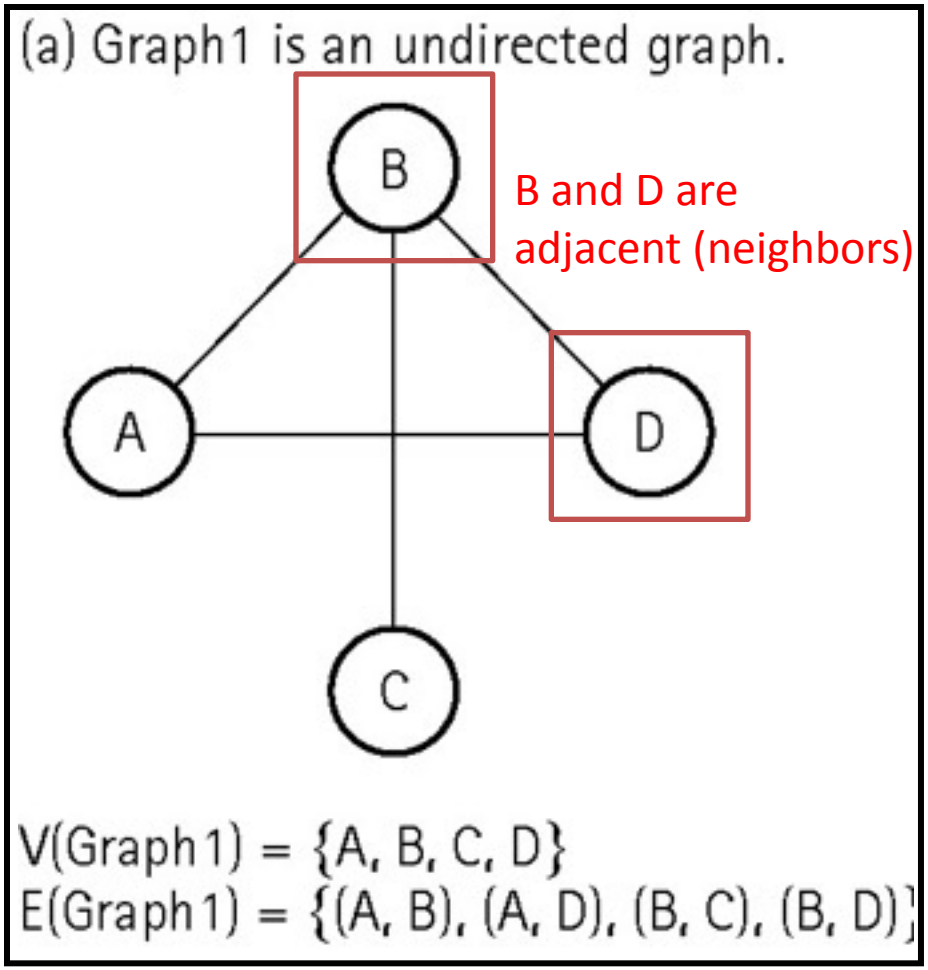
Undirected Graphs Details

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



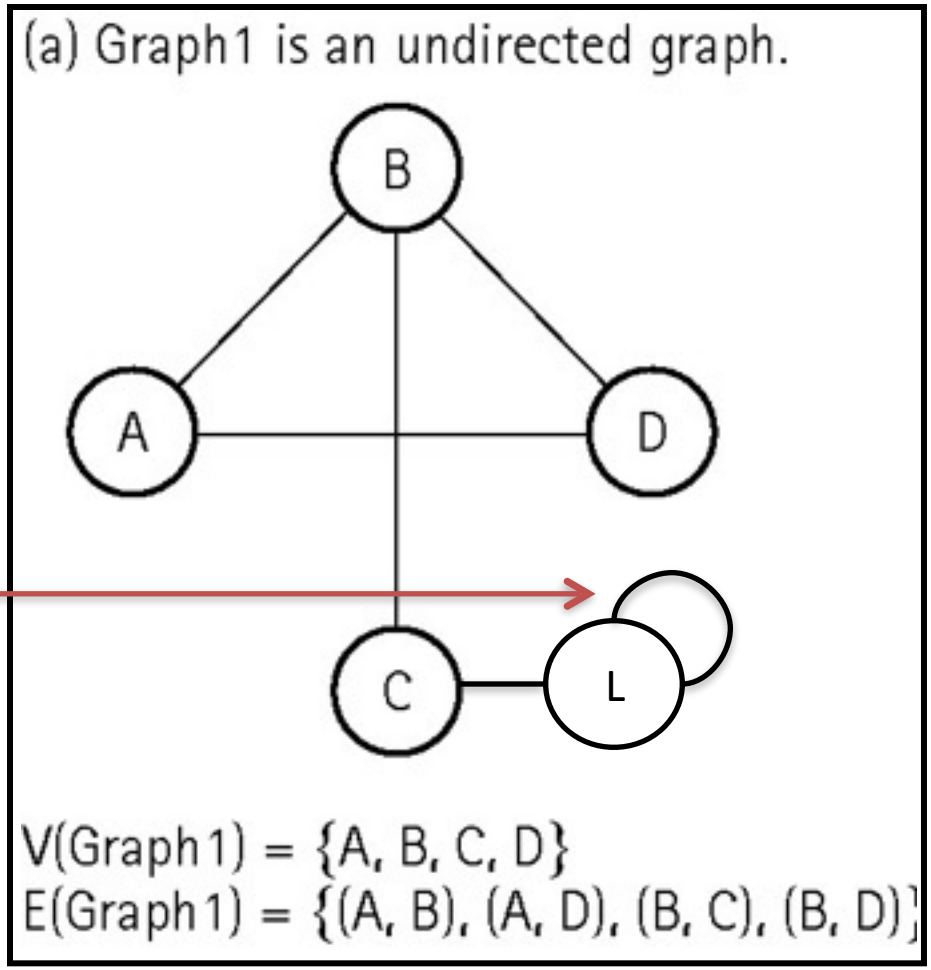
Undirected Graphs Details

- Two vertices are said to be adjacent if there is an edge connecting them.
- They are called neighbors to each other.



Undirected Graphs Details

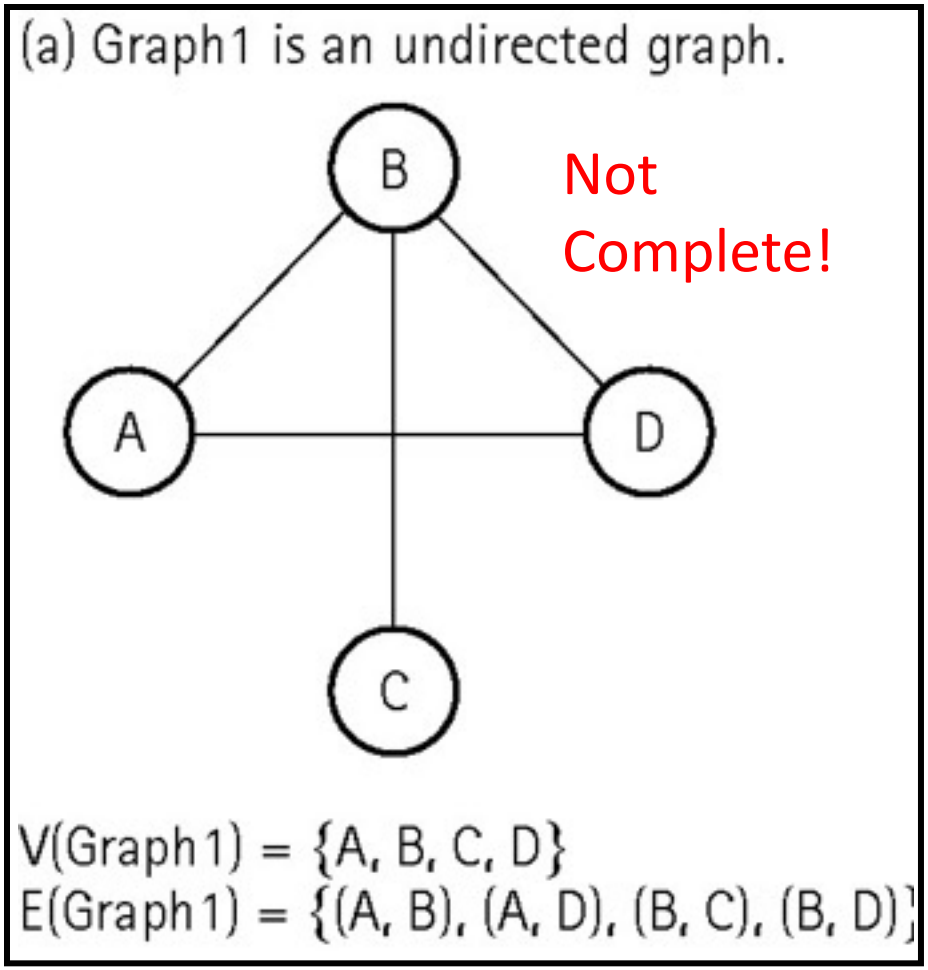
- An edge that connects a vertex to itself is called a <u>self-loop</u> or <u>sling</u>



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

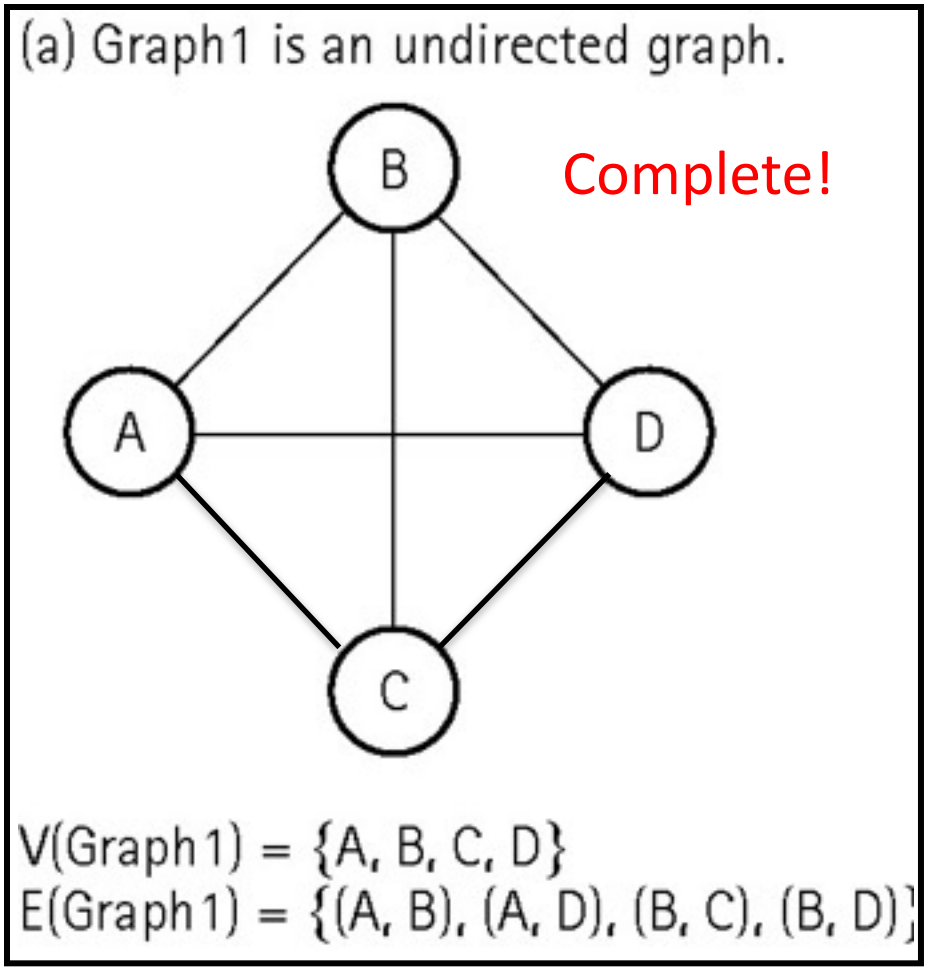
Undirected Graphs Details

- An undirected graph is considered <u>complete</u> if it has the maximum number of edges connecting vertices



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 **6** vertices?

(a) 5

(b) 10

(c) 15

(d) 20

(e) 25

Answer on next slide

Clicker Question #1

How many edges are there in a complete graph with **6** vertices?

(a) 5

(b) 10

(c) 15

(d) 20

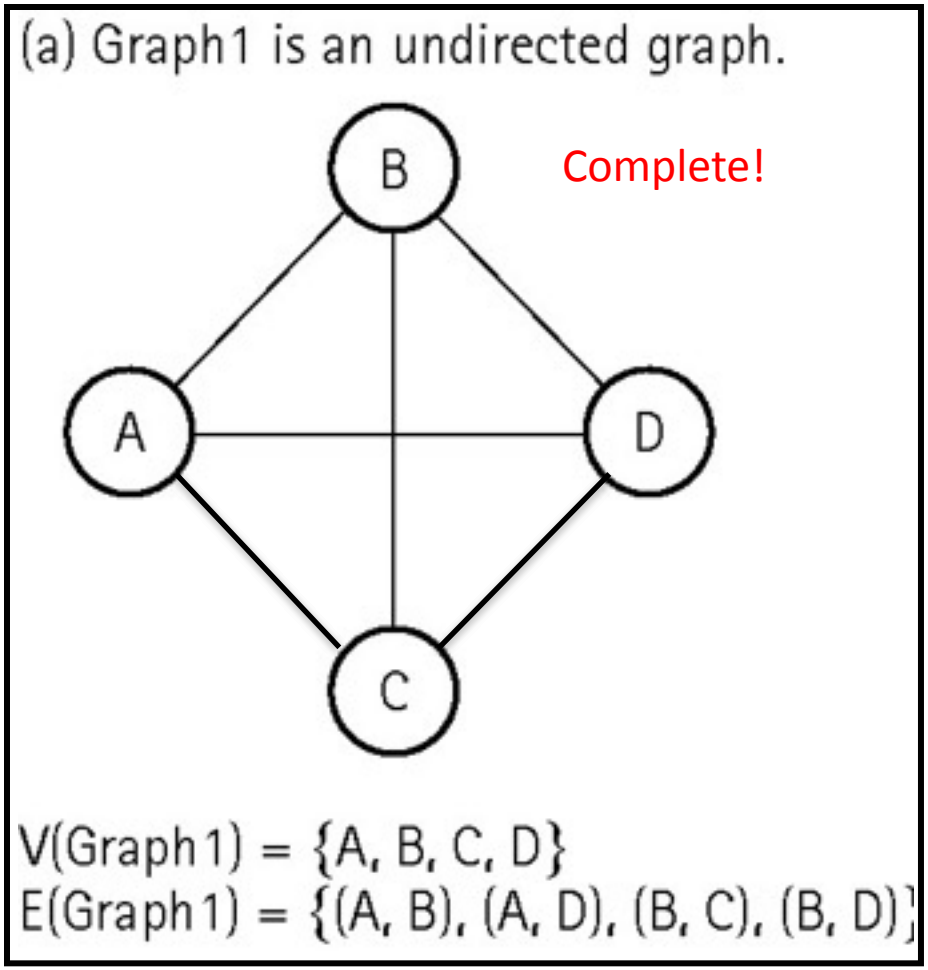
(e) 25

Complete Graphs

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

(N-1)+(N-2)+(N-3)+…+1 =

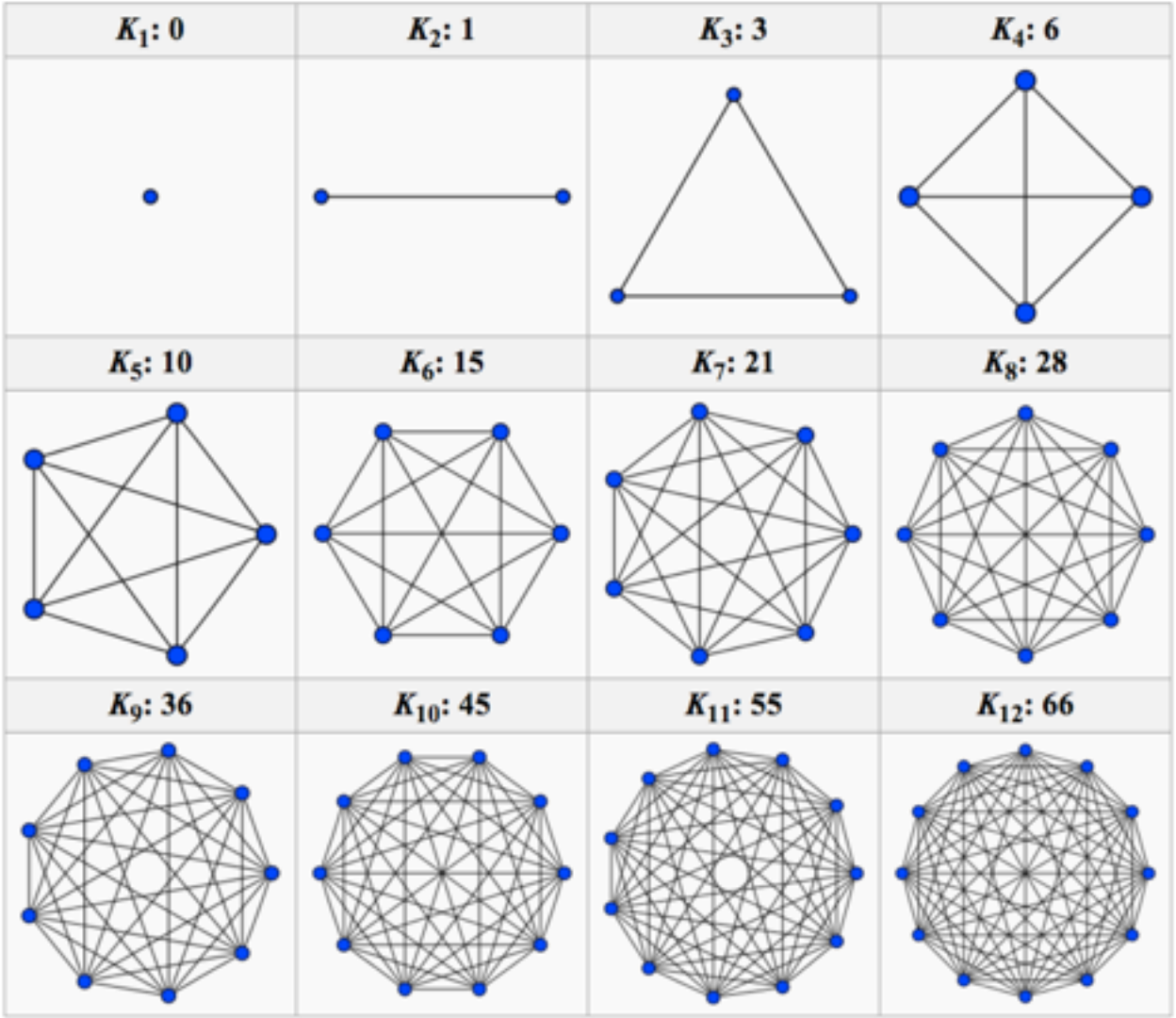
N(N-1)/2



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

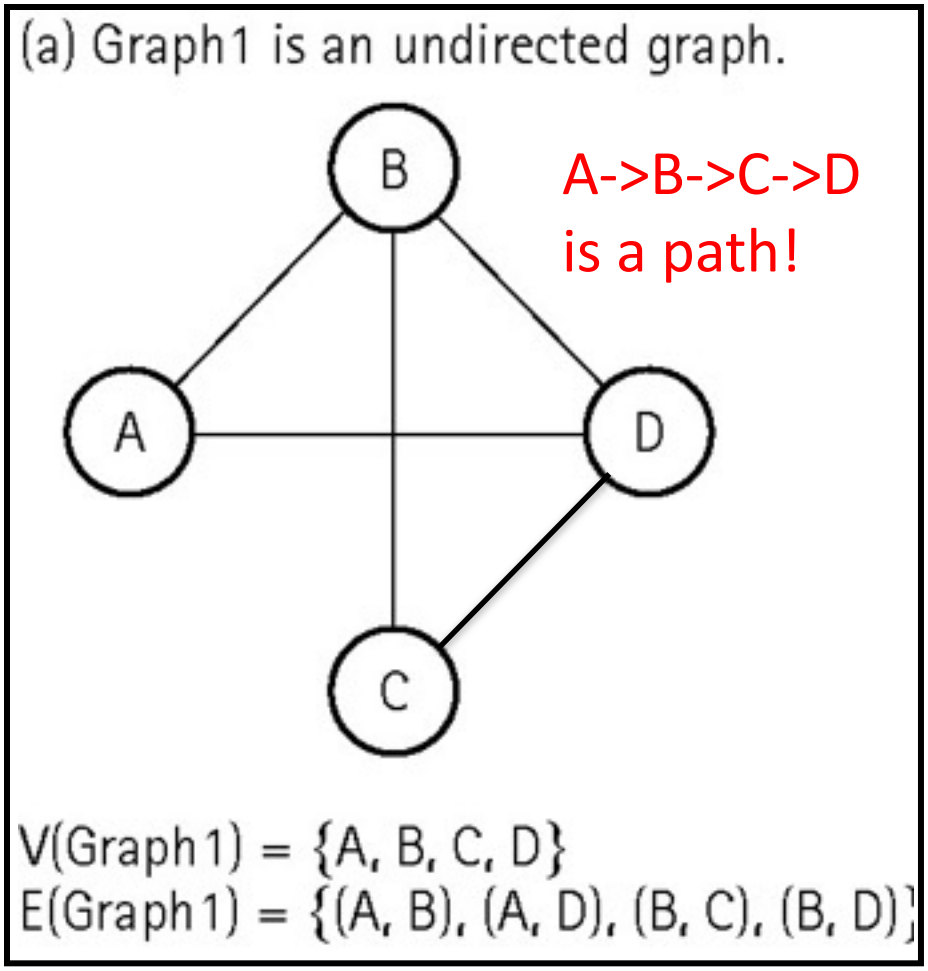
Complete Graphs

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



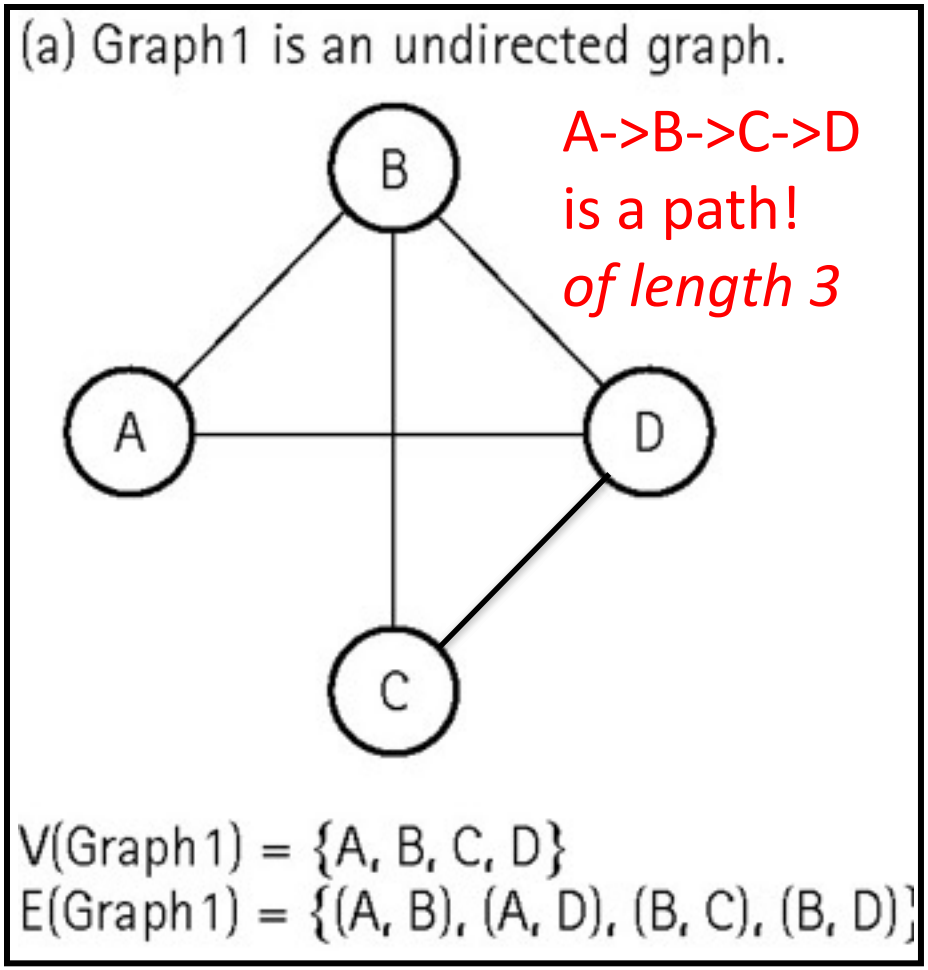
Undirected Graphs Details

- 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 generally only care about simple paths.



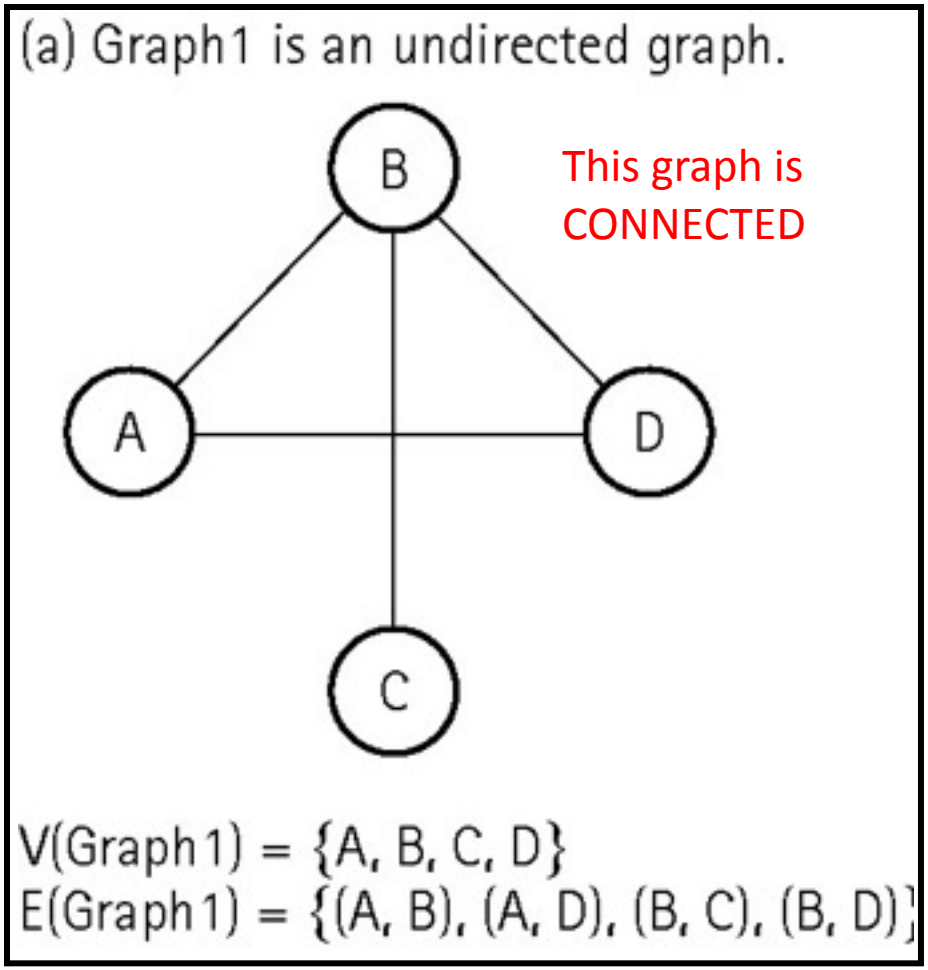
Undirected Graphs Details

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



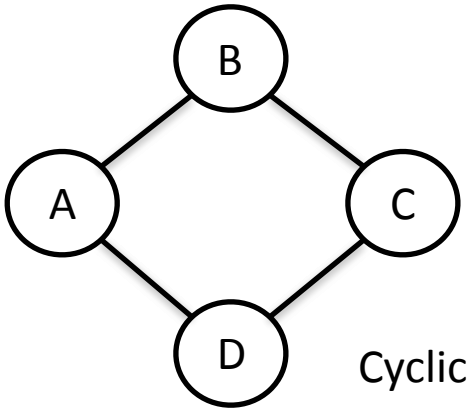
Undirected Graphs Details

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

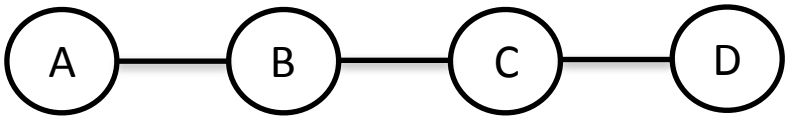


Cycles

- 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



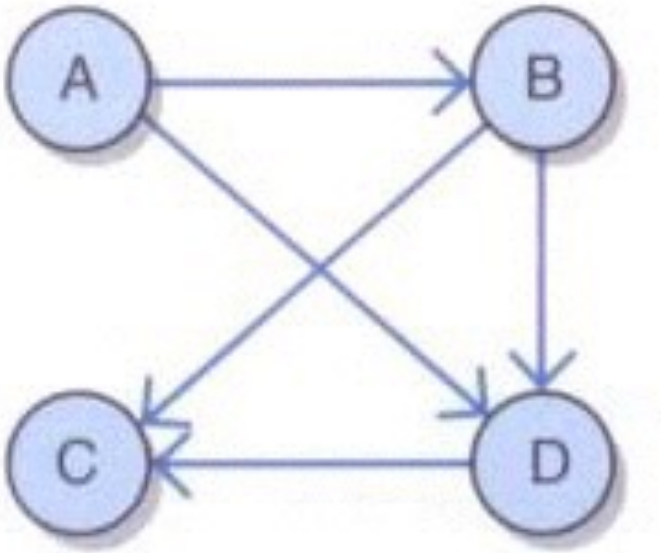
Acyclic

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!

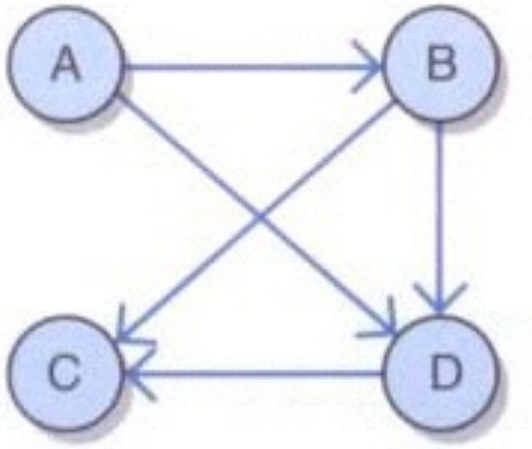
Directed Graph Example

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



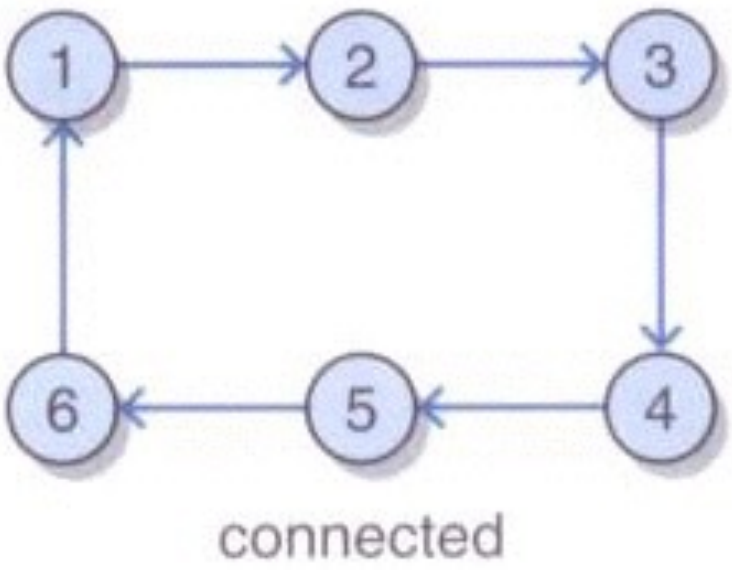
Directed Graphs Definitions

- 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 any two vertices in the graph 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?

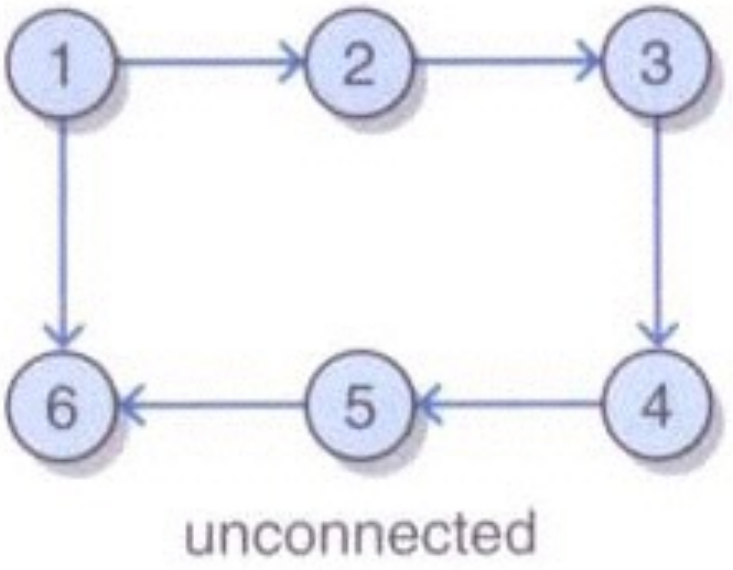


Connected Directed Graphs

- 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 path between every pair of vertices. Otherwise, it’s non-connected graph.
- The combinations: connected directed graph, connected undirected graph, non-connected directed graph etc.

Clicker Question #2

This graph is:

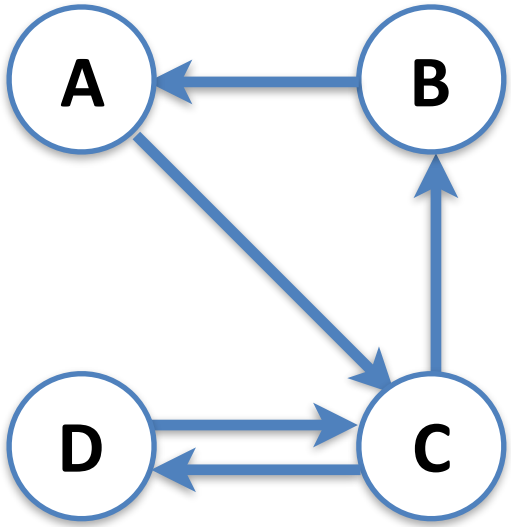
(a) connected directed

(b) connected undirected

(c) non-connected directed

(d) non-connected undirected

(e) none of the above



Answer on next slide

Clicker Question #2

This graph is:

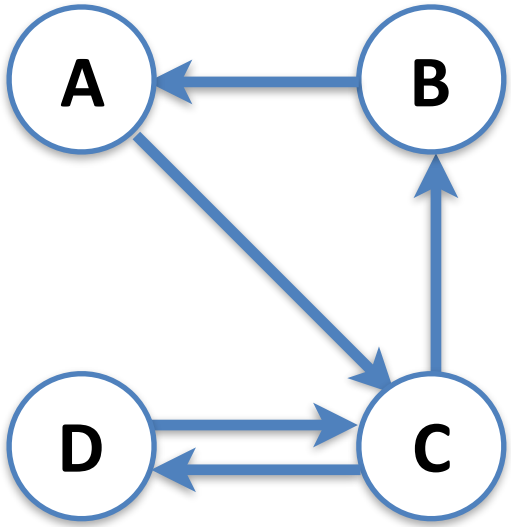
(a) connected directed

(b) connected undirected

(c) non-connected directed

(d) non-connected undirected

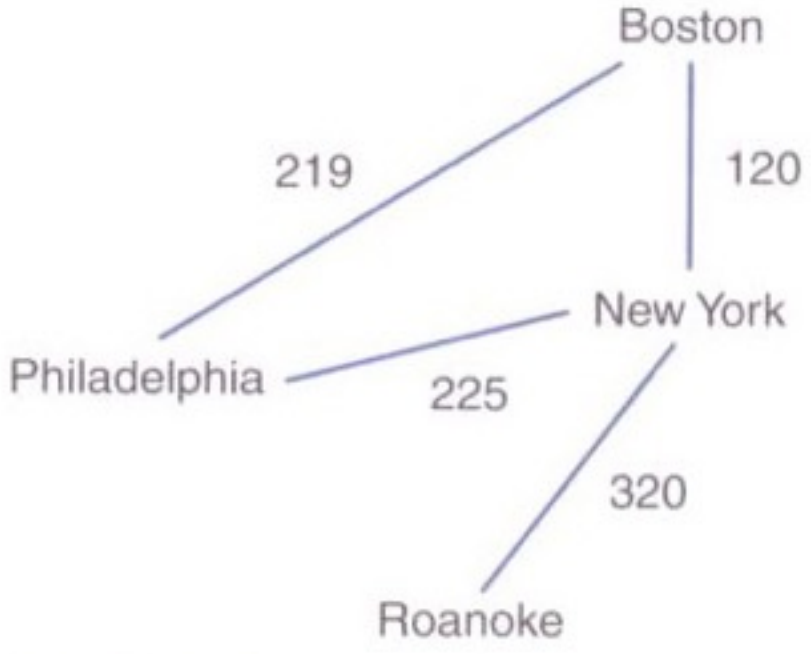
(e) none of the above



Weighted Graphs

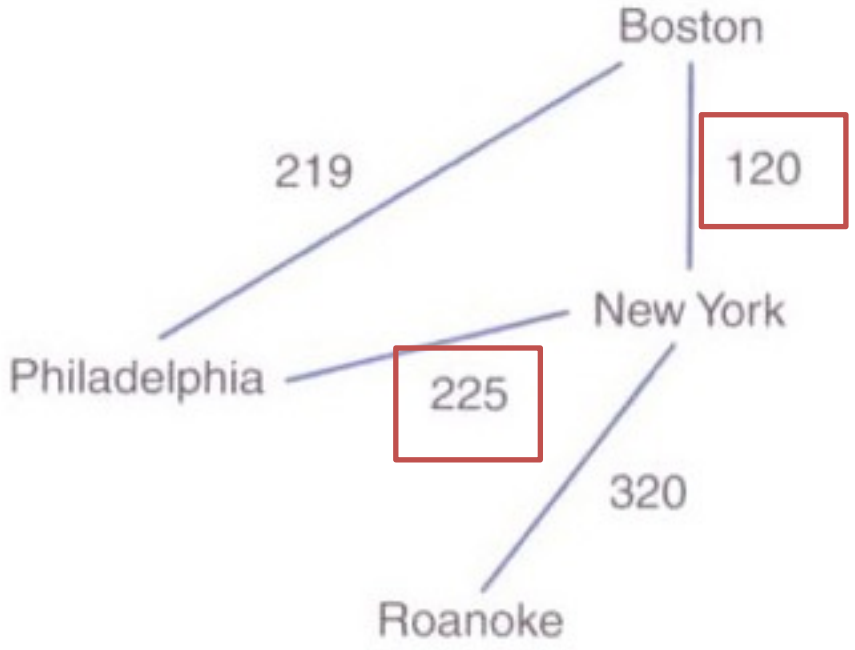
- A weighted graph is a graph with weights (or costs) associated with each edge.

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



Weighted Graphs

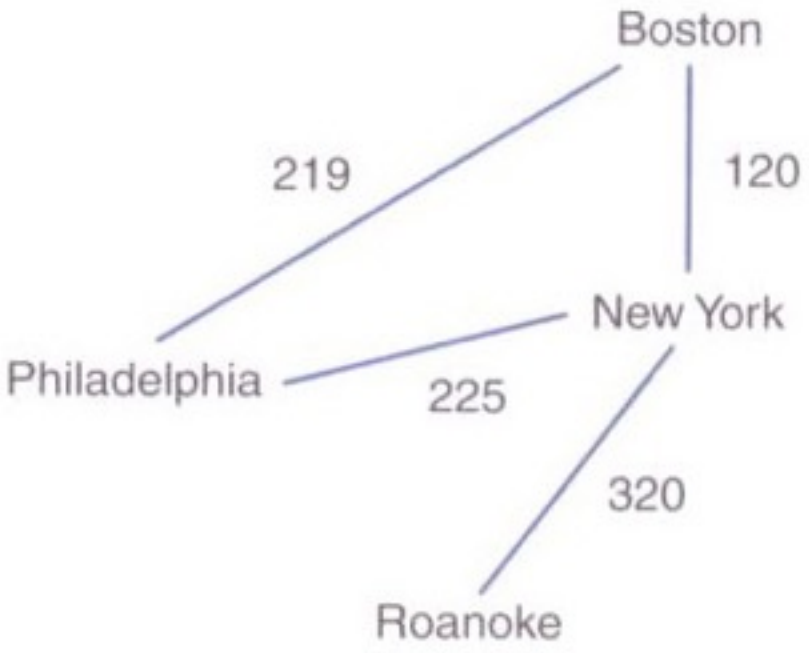
- If there is a path between two vertices, the path weight is the sum of the weights of the edges in the path.
- As you may have multiple paths, each path may have a different weight.
- In many cases, you may want the path with the minimum / maximum weight.



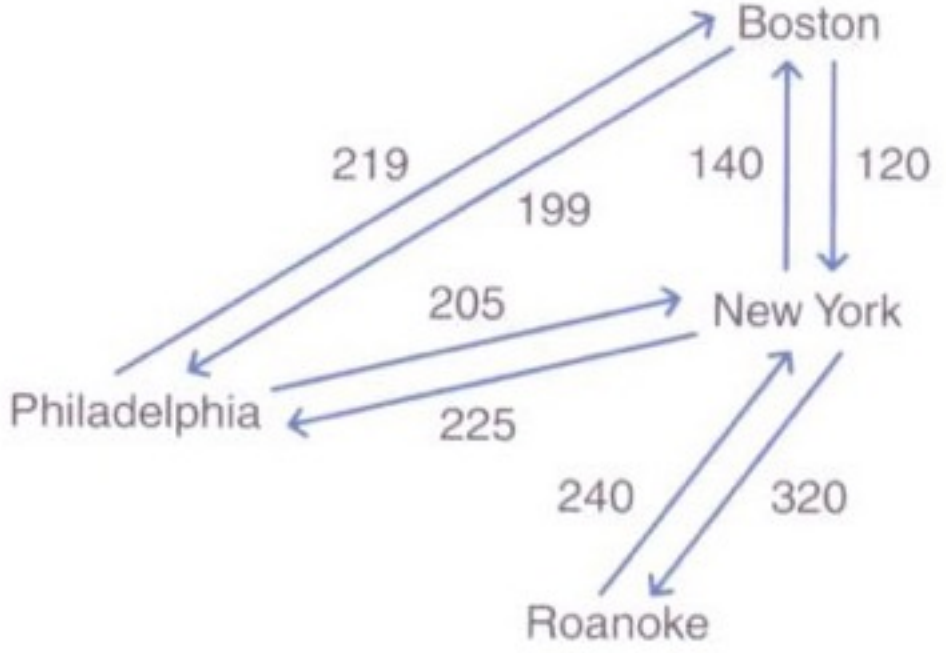
The path from Boston to Philly through NYC:
120 + 225 = 345

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).



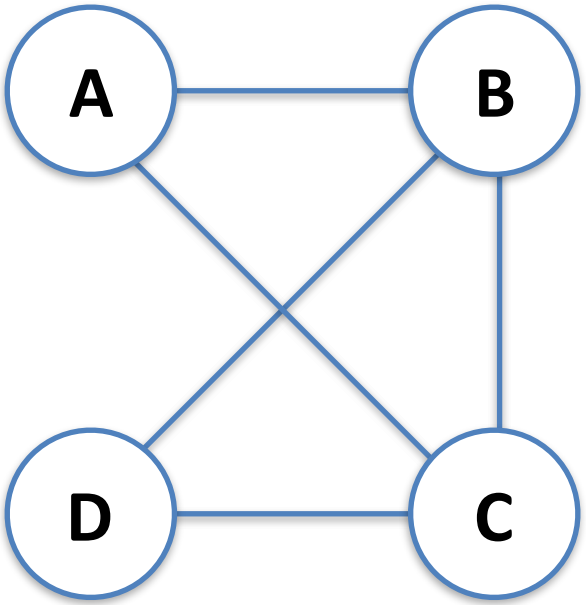
Undirected Weighted Graph



Directed Weighted Graph

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 unweighted, undirected graph. (The matrix is binary and symmetric)



to

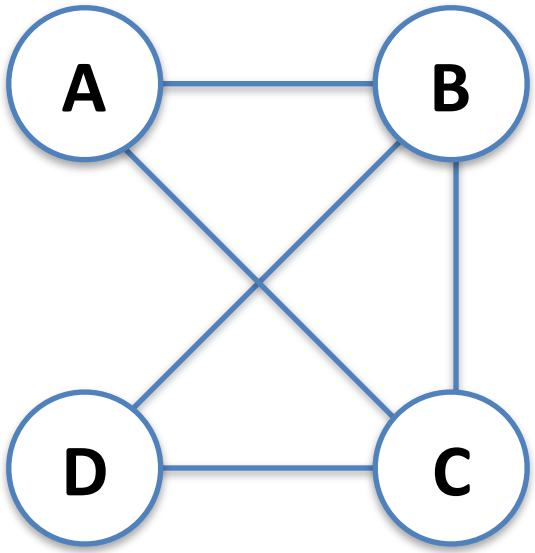
from

	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:

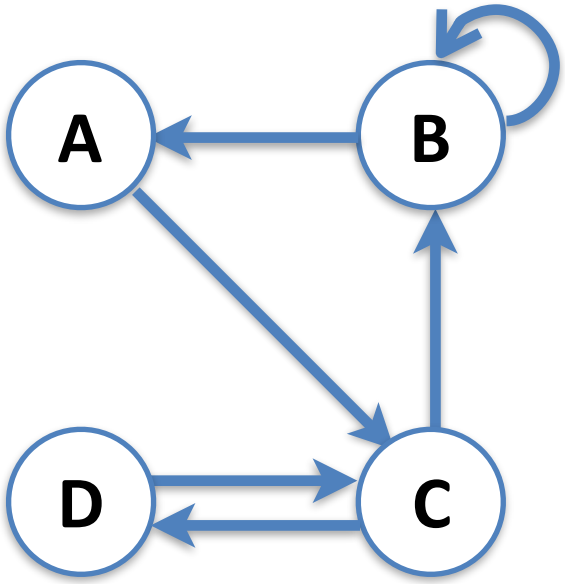
- The number of neighbors each vertex has. How?
- Is there a path between two vertices? If so, what's the path length?
- Is this a connected graph?



from \ to	*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 non-symmetric)



from \ to	*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 from A to C? If so, what's the path length? (If there are multiple, pick the shortest)

(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

from \ to	*A*	*B*	*C*	*D*
A	0	1	0	1
B	0	0	0	1
C	1	1	1	1
D	0	0	0	0

Answer on next slide

Clicker Question #3

Given the adjacency matrix of a directed graph, is there a path from A to C? If so, what's the path length? (If there are multiple, pick th

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

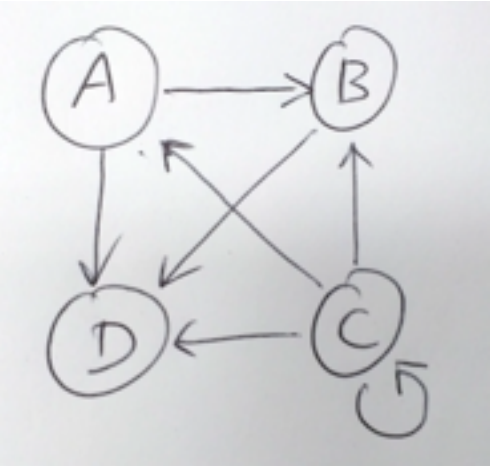
(a) Yes, path length is 1

(b) Yes, path length is 2

(c) Yes, path length is 3

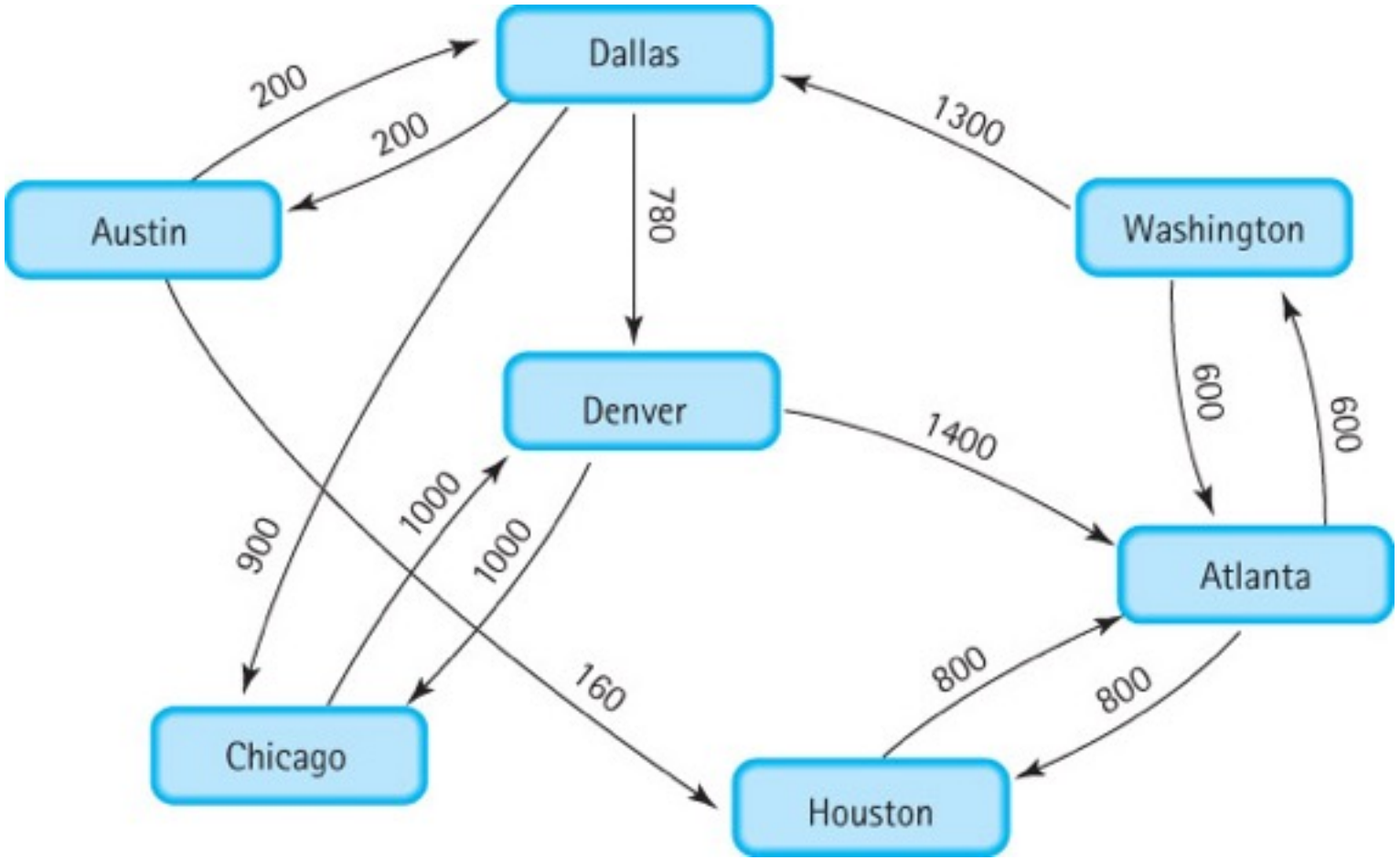
(d) Yes, path length is 4

(e) No

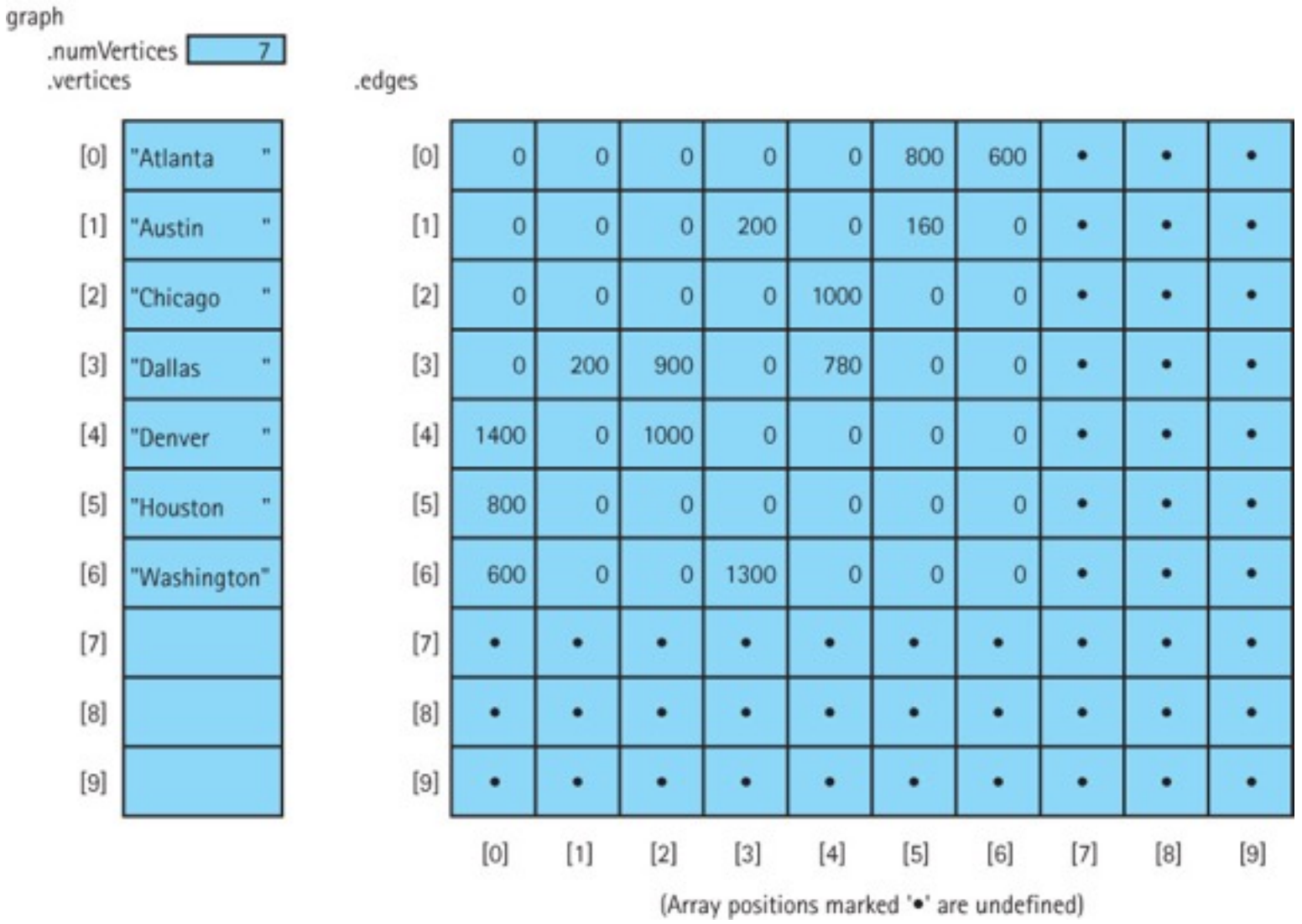


Representing Graphs

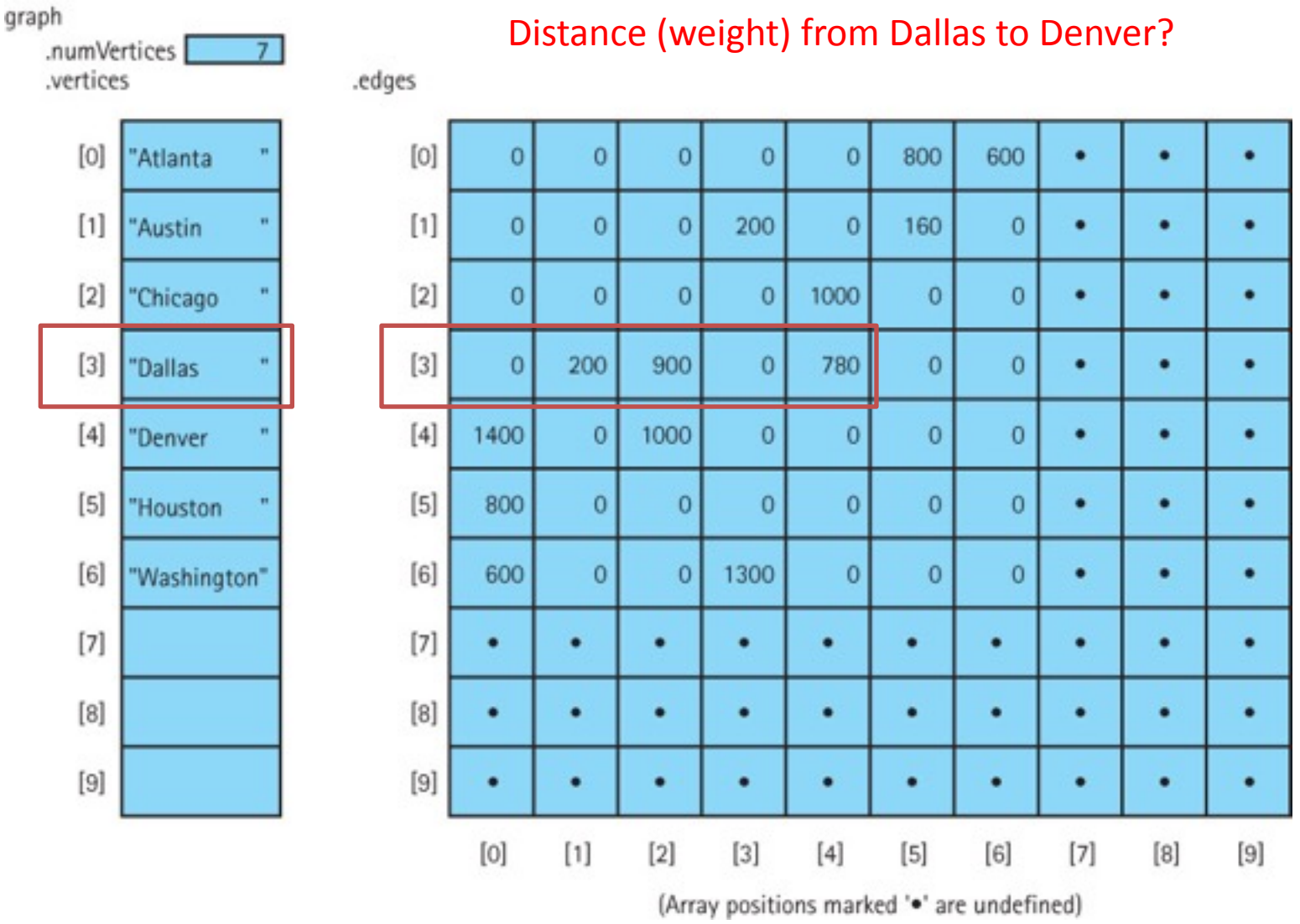
- Example of weighted, directed graph.



Adjacency Matrix



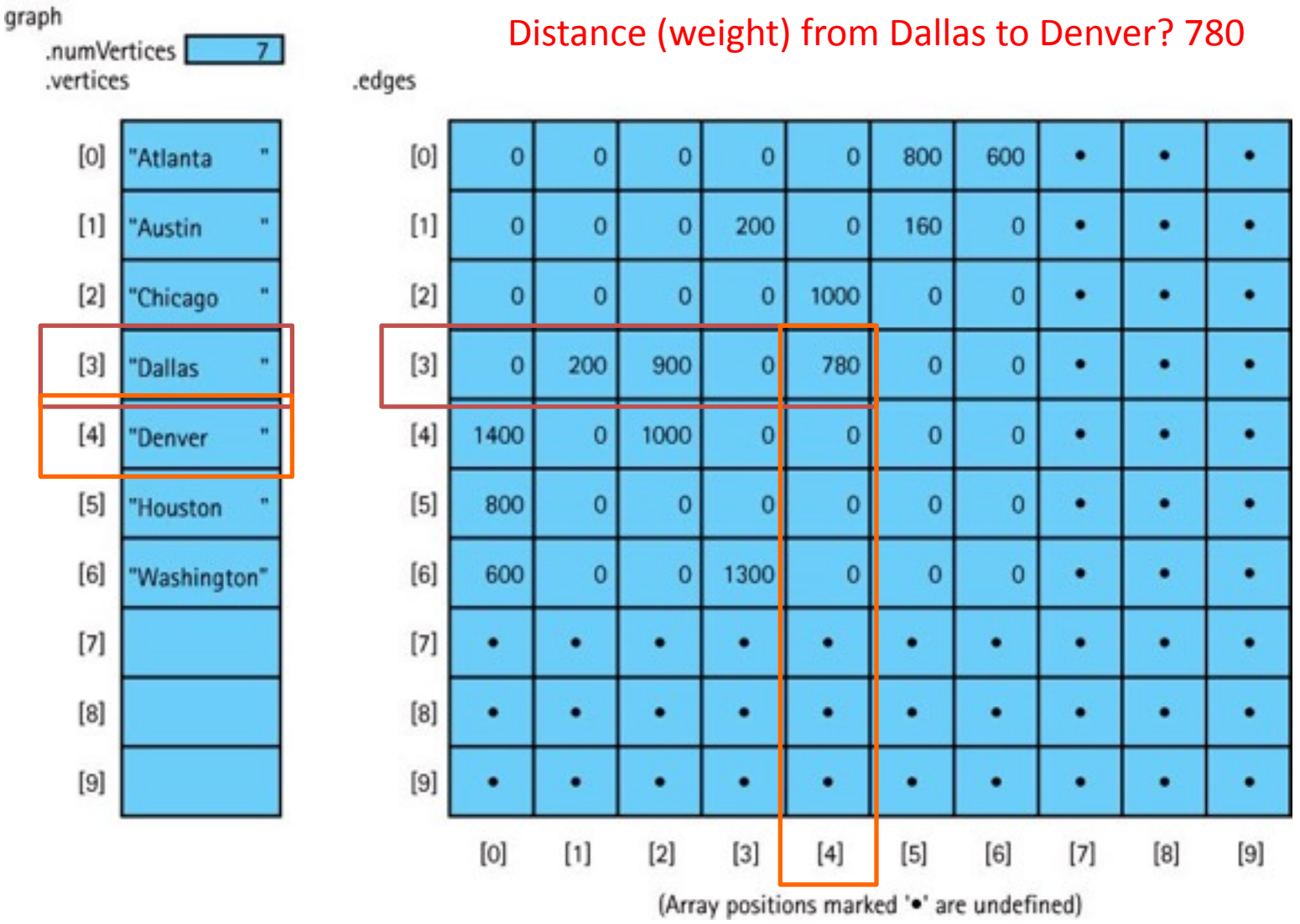
Adjacency Matrix



Distance (weight) from Dallas to Denver?

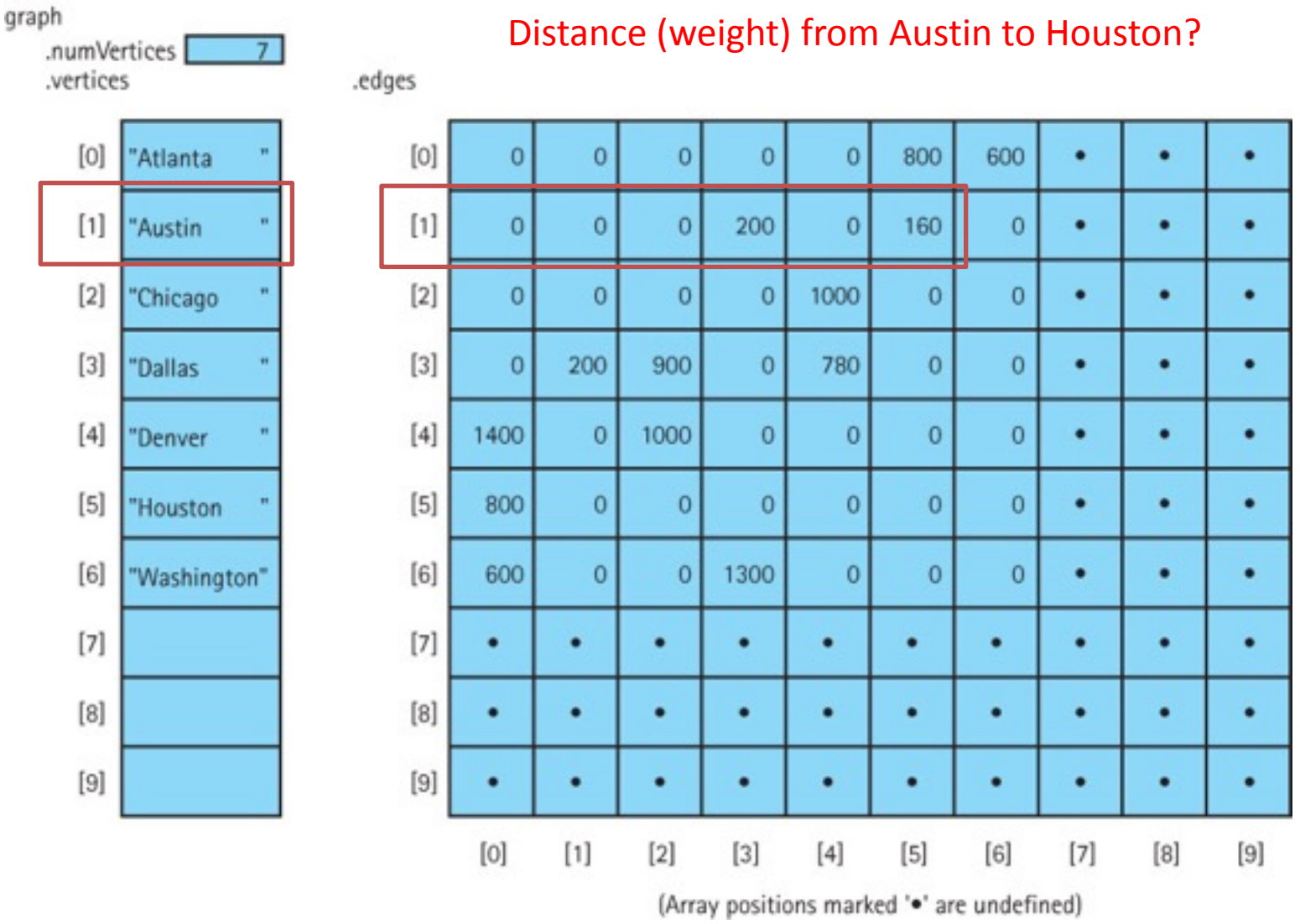
Adjacency Matrix

Distance (weight) from Dallas to Denver? 780



Adjacency Matrix

Distance (weight) from Austin to Houston?



Adjacency Matrix

Distance (weight) from Austin to Houston? 160