

CSE 3318

Week of 07/03/2023

Instructor : Donna French

Graphs

So you might wonder...

What do graphs have to do with algorithms?

A lot of problems can be converted into graph problems.

If we have algorithms for solving graph problems, then we can also solve the problems that we can convert into graph problems.

Graphs

For example, have you seen one of these?

It has many names

15-puzzle

Gem Puzzle

Boss Puzzle

Game of Fifteen

Mystic Square



This game can be converted into a graph problem and be solved with a graph traversal algorithm.

Graphs

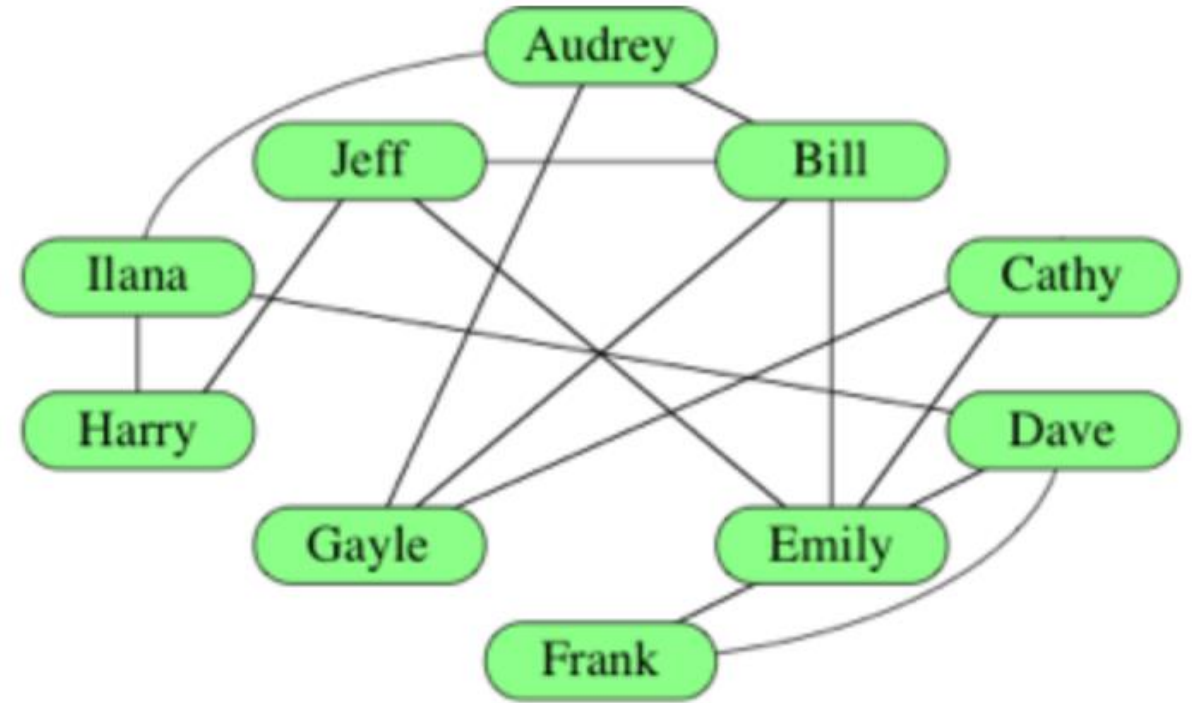
This represents a graph of a social network.

A line between 2 names indicates that people know each other.

Cathy knows Gayle and Gayle knows Bill but Cathy does not know Bill.

Each name is a **vertex** of the graph.

Each line is an **edge** – an edge connects 2 vertices (plural of vertex).



Graphs

How many vertices does this graph have?

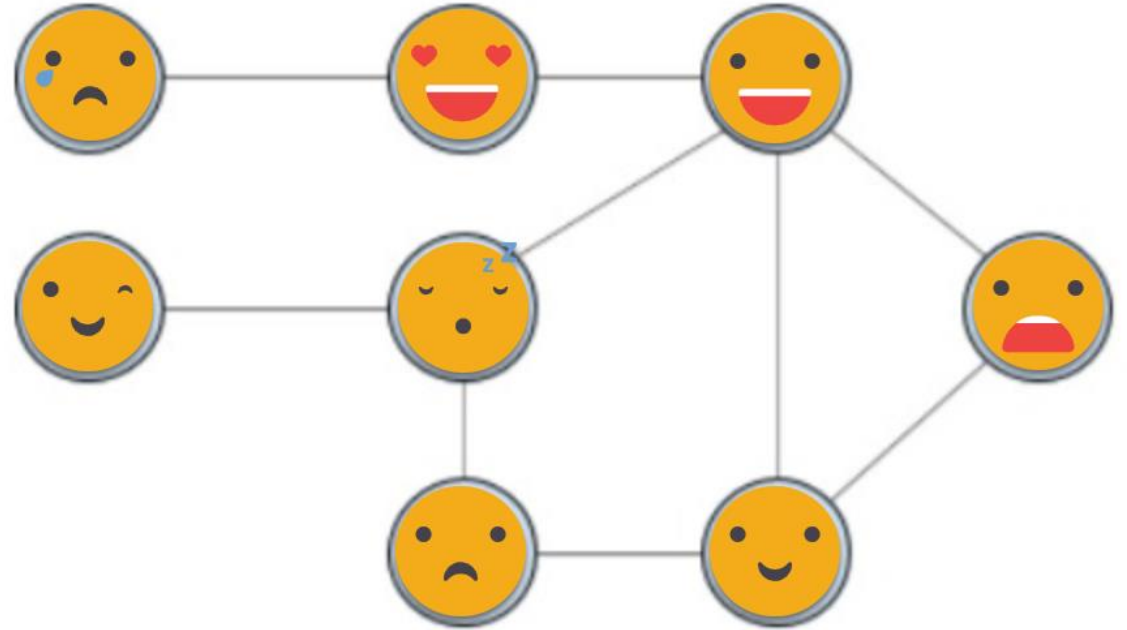
8

How many edges?

9

How many edges could be eliminated and still be able to connect all vertices without backtracking?

2 and we would be left with 7 edges.



Graphs

We can depict that two vertices are connected via an edge

(Audrey, Bill) or (A, B)

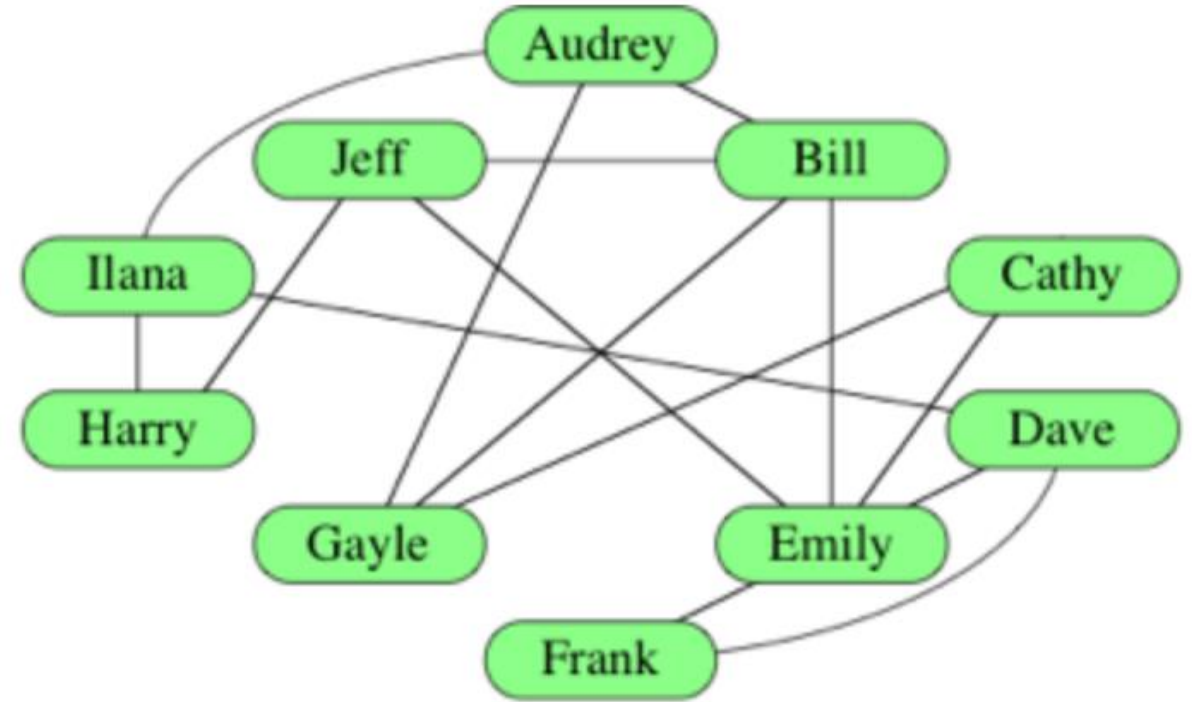
What is the edge between Jeff and Emily?

(J,E)

Is there a difference between
(A,B) and (B,A) or (J,E) and (E,J)?

No. The "knows each other" relationship indicates a bidirectional relationship.

All of this graph's edges are undirected.



Graphs

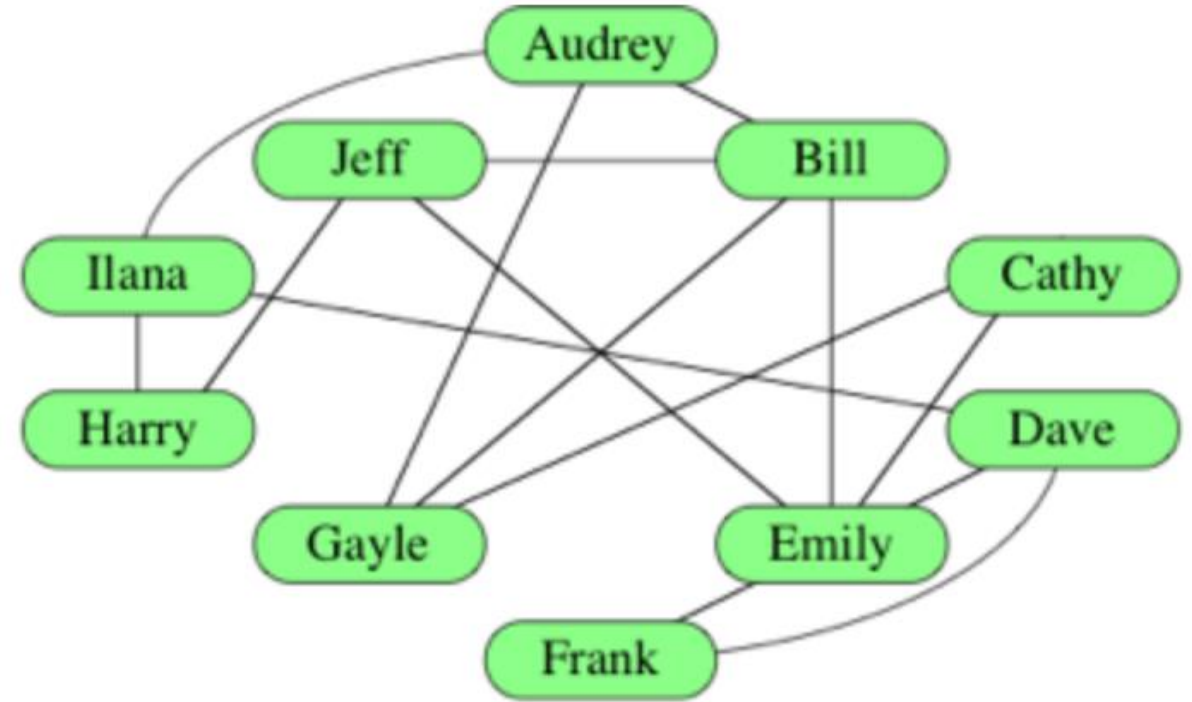
All of this graph's edges are undirected.

An undirected edge of (r,s) is the same as edge (s,r)

In an undirected graph, the edge between two vertices is incident on the two vertices.

The vertices connected by an edge are adjacent or neighbors.

The number of edges incident on a vertex is the degree of the vertex.

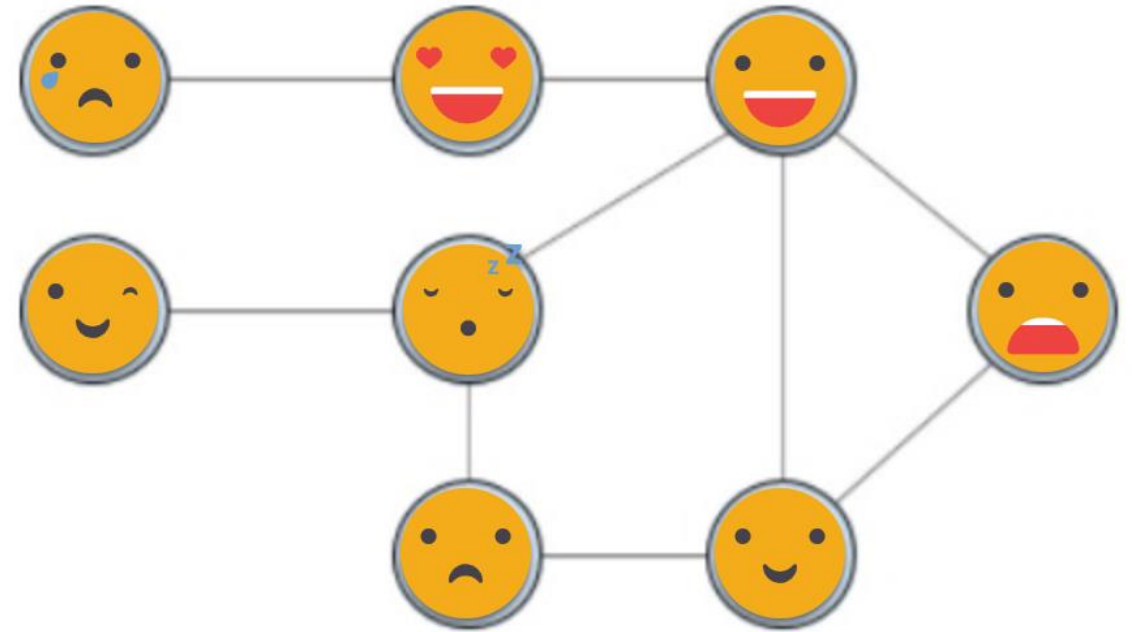


Graphs

Two vertices are called adjacent if they are connected by an edge.

Is 😊 adjacent to 😞 ? ❌

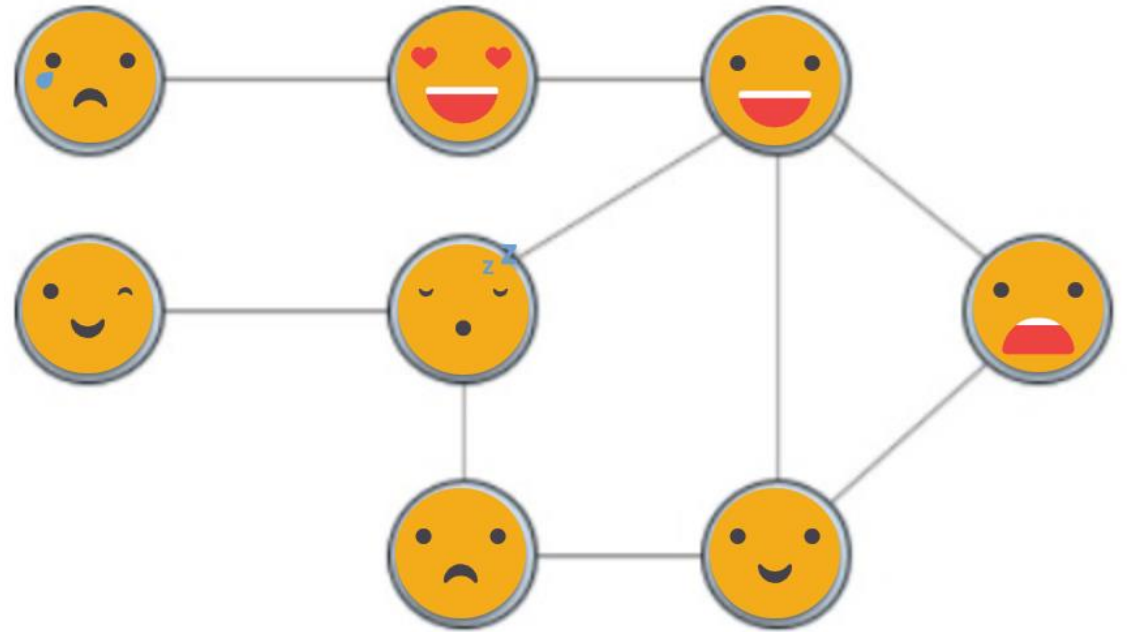
Is 😞 adjacent to 😊 ? ✅



Graphs

Two edges are called incident, if they share a vertex.

Also, a vertex and an edge are called incident, if the vertex is one of the two vertices the edge connects.



What if Gayle wants to meet Harry?

Graphs

Gayle could ask Cathy because Cathy knows Emily who know Jeff who knows Harry.

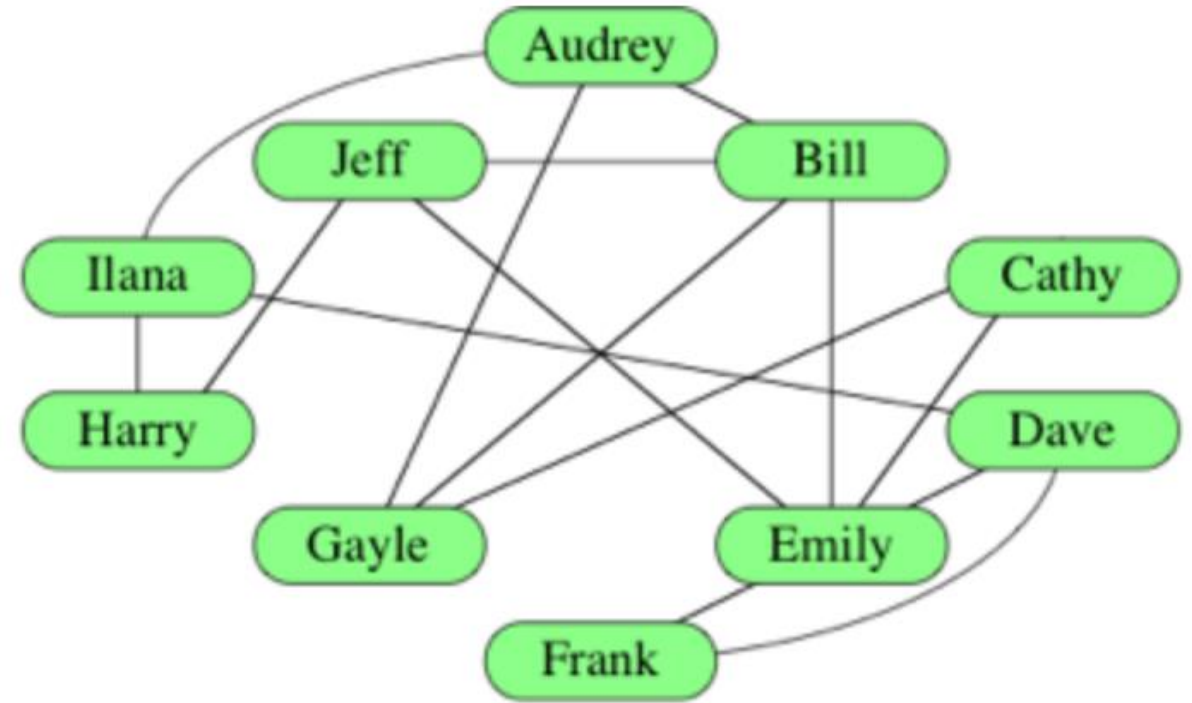
This is called the path between Gayle and Harry.

Is there a shorter route?

Yes – Gayle -> Audrey -> Ilana -> Harry

Is there another?

Yes – Gayle -> Bill -> Jeff -> Harry



Is there a path shorter than these two?

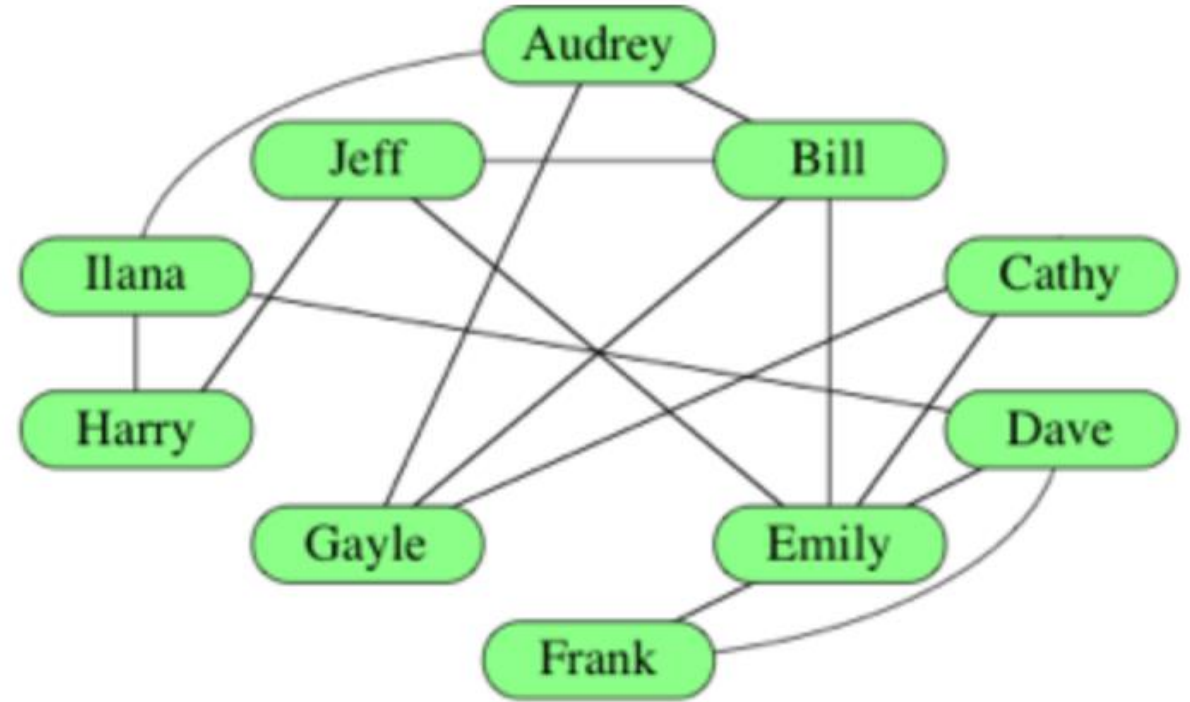
When a path goes from a particular vertex back to itself, that path is also called a

cycle

Are there any cycles in this graph?

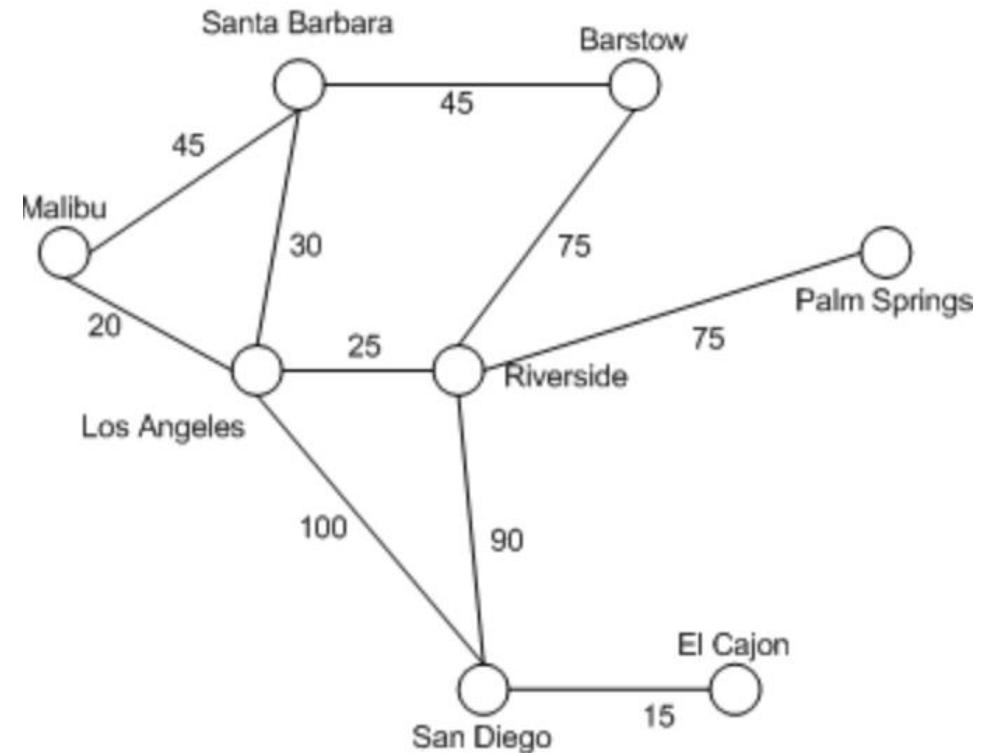
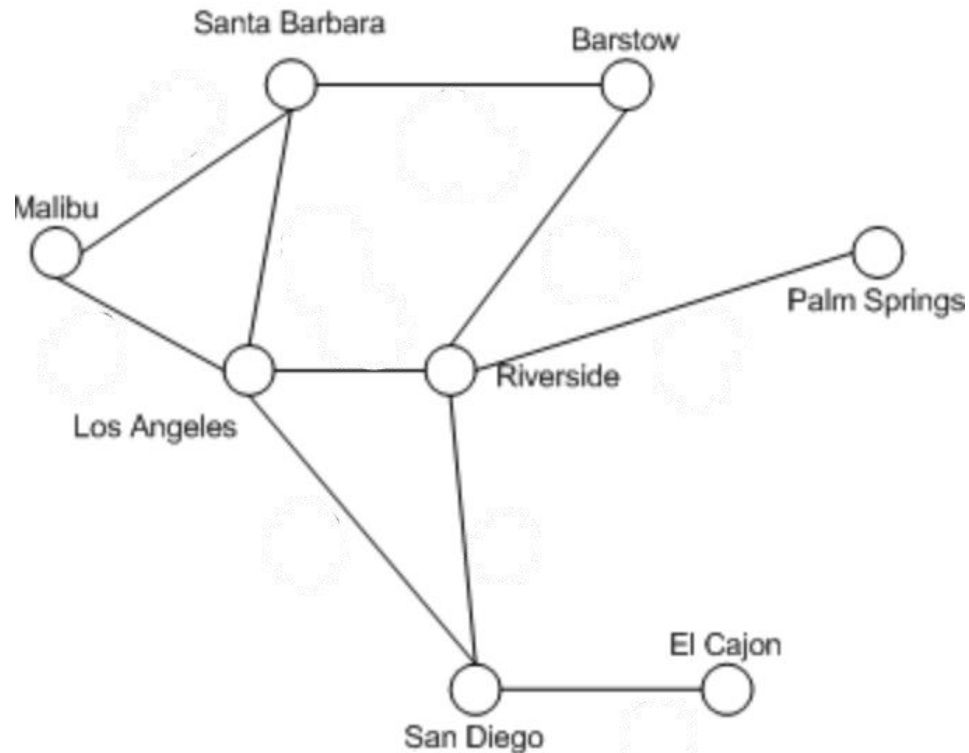
Is there a path from every vertex to every other vertex?

Graphs



Graphs

We can add numeric values to edges to indicate relationships between vertices.

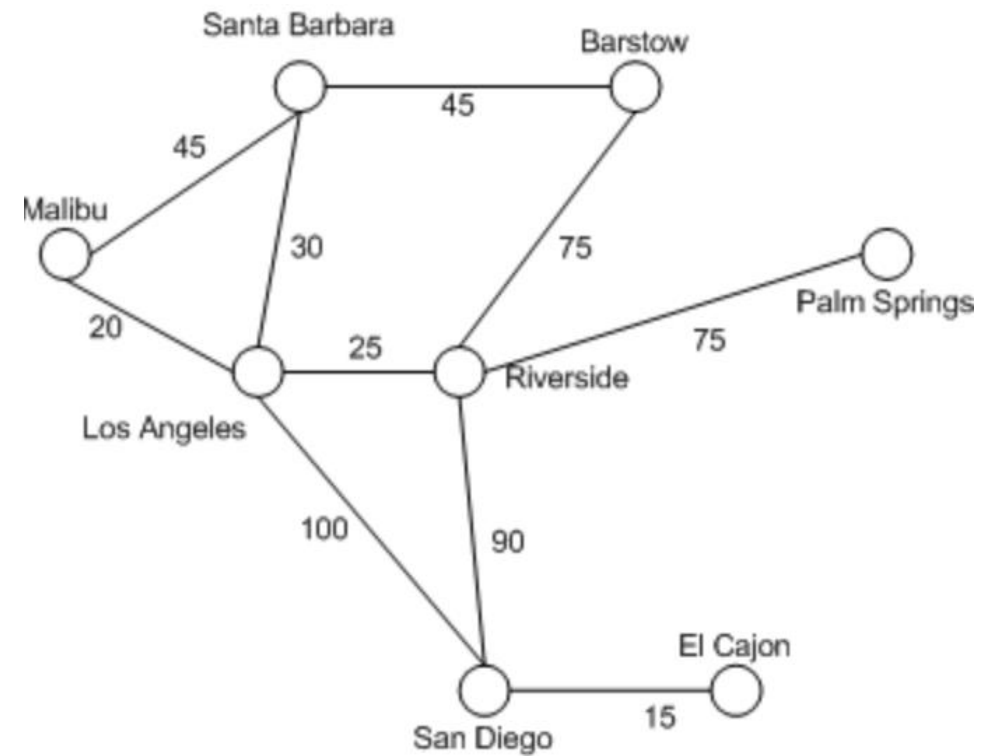


Graphs

In this graph, the number on the edge indicates the distance between each vertex.

Malibu is 45 units from Santa Barbara.

The unit is not specified here so we should not assume miles – could be minutes or kilometers or something else.



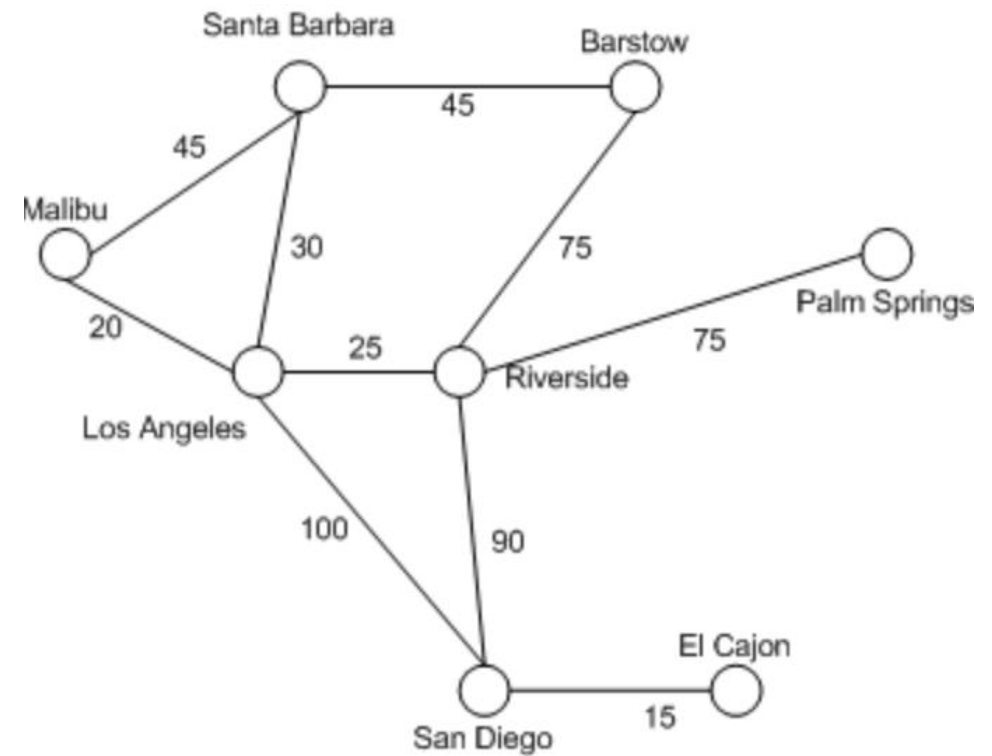
Graphs

When a number is put on an edge, that edge is said to have a **weight**.

A graph whose edges have weights is called a

weighted graph

The shortest path in a weighted graph is the path with the minimum sum of edge weights over all paths between two vertices.



Graphs

What is the shortest path between
Palm Springs and Malibu?

Palm Springs -> Riverside 75

Three choices from Riverside

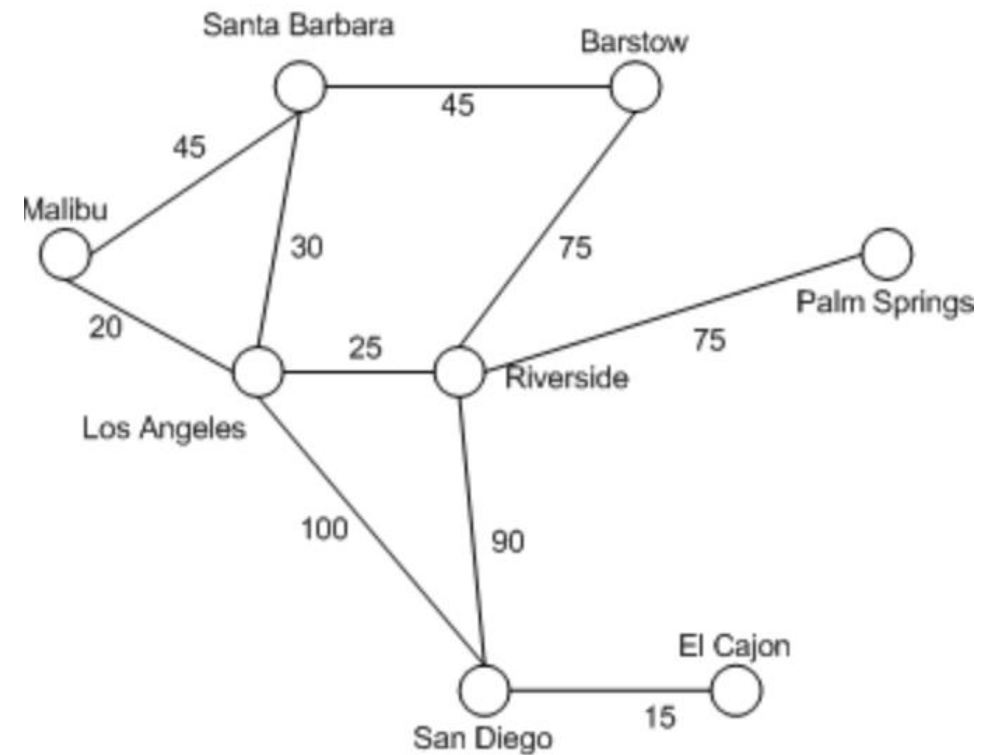
Barstow 75

Los Angeles 25

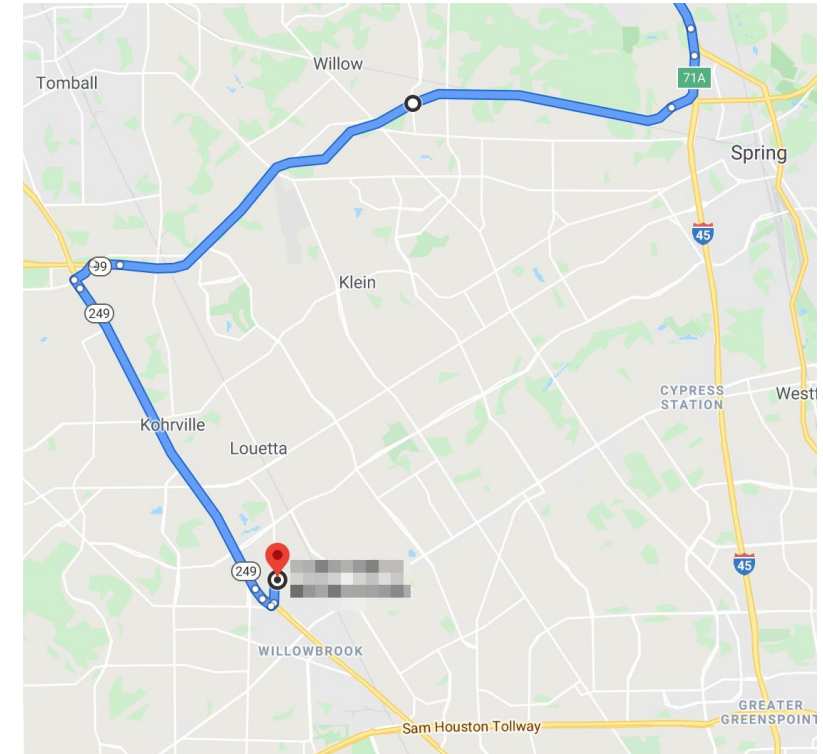
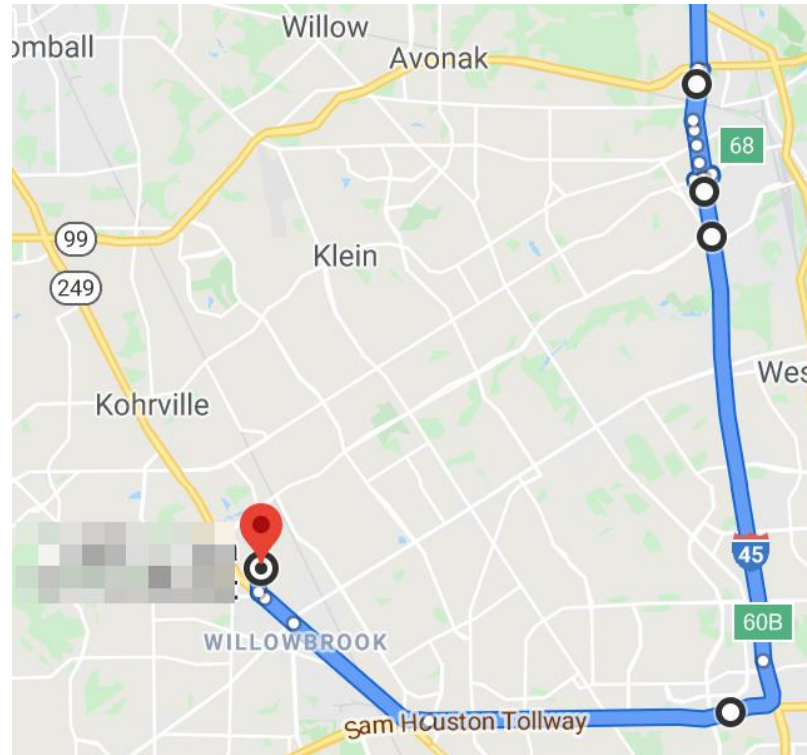
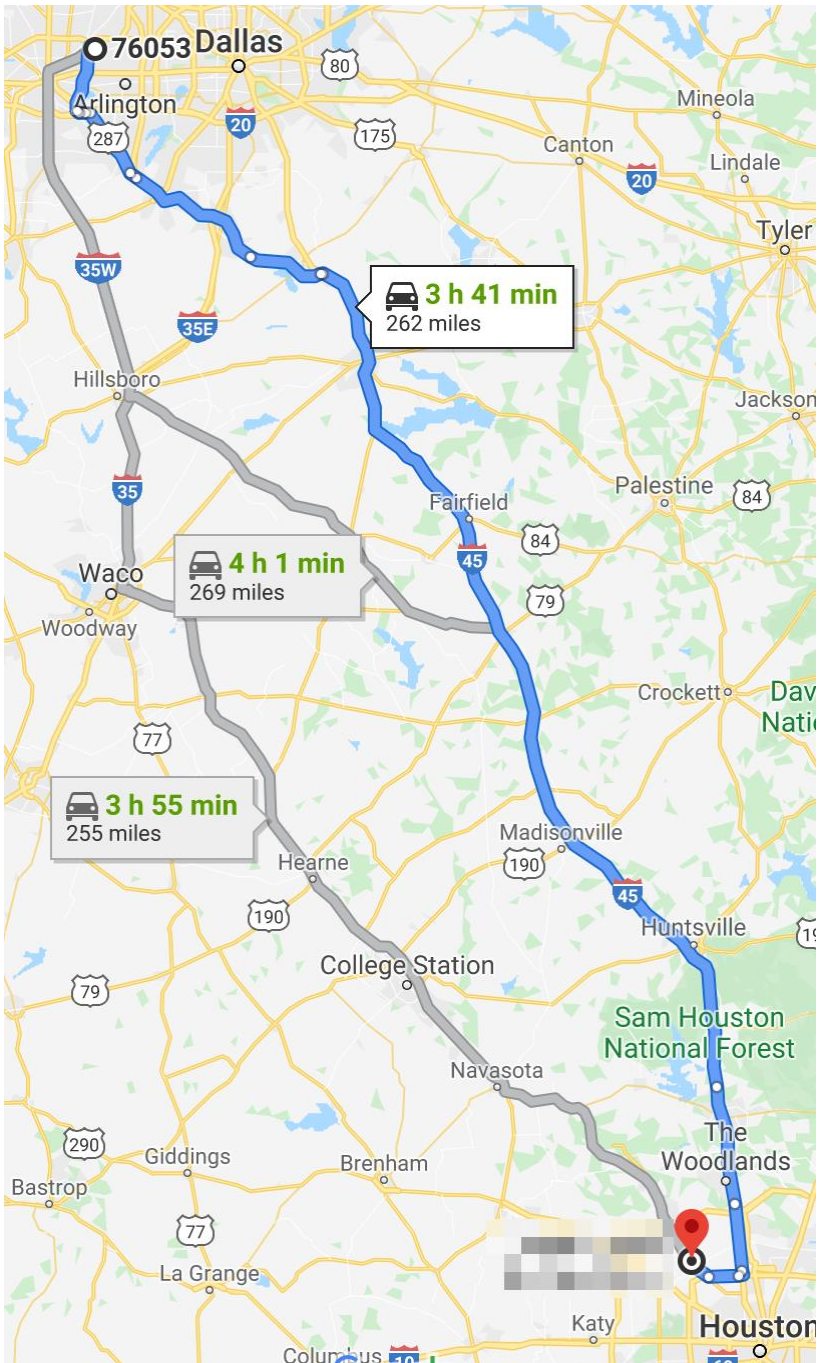
San Diego 90

Which route is "best"?

What happens when you factor in traffic, tollways, construction...?

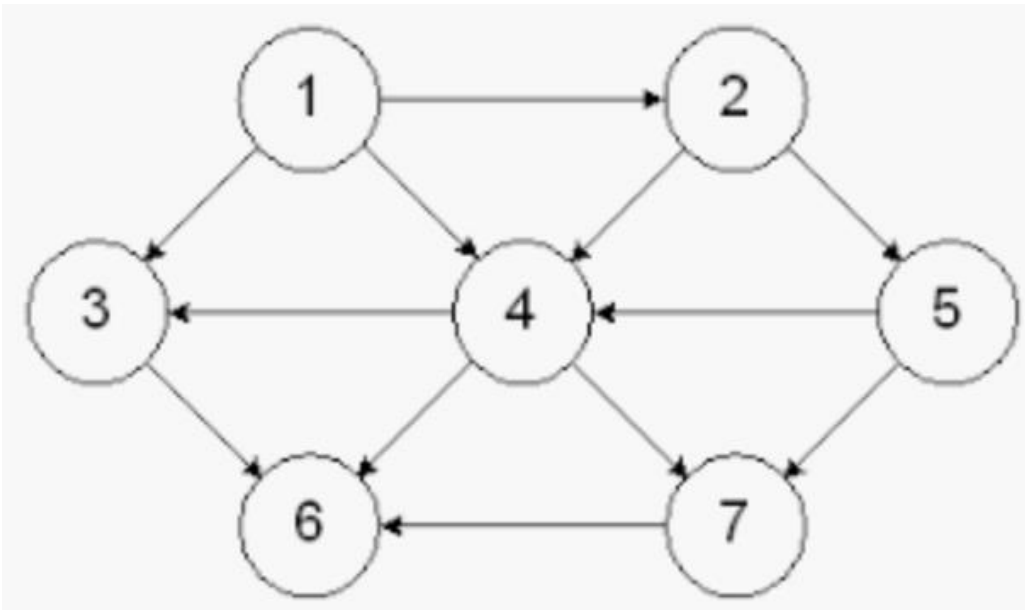


Graphs



Graphs

The relationship between vertices does not always go both ways. Notice that the edges now have arrows? This is called **directed graph**.



Does this directed graph have any cycles?

No.

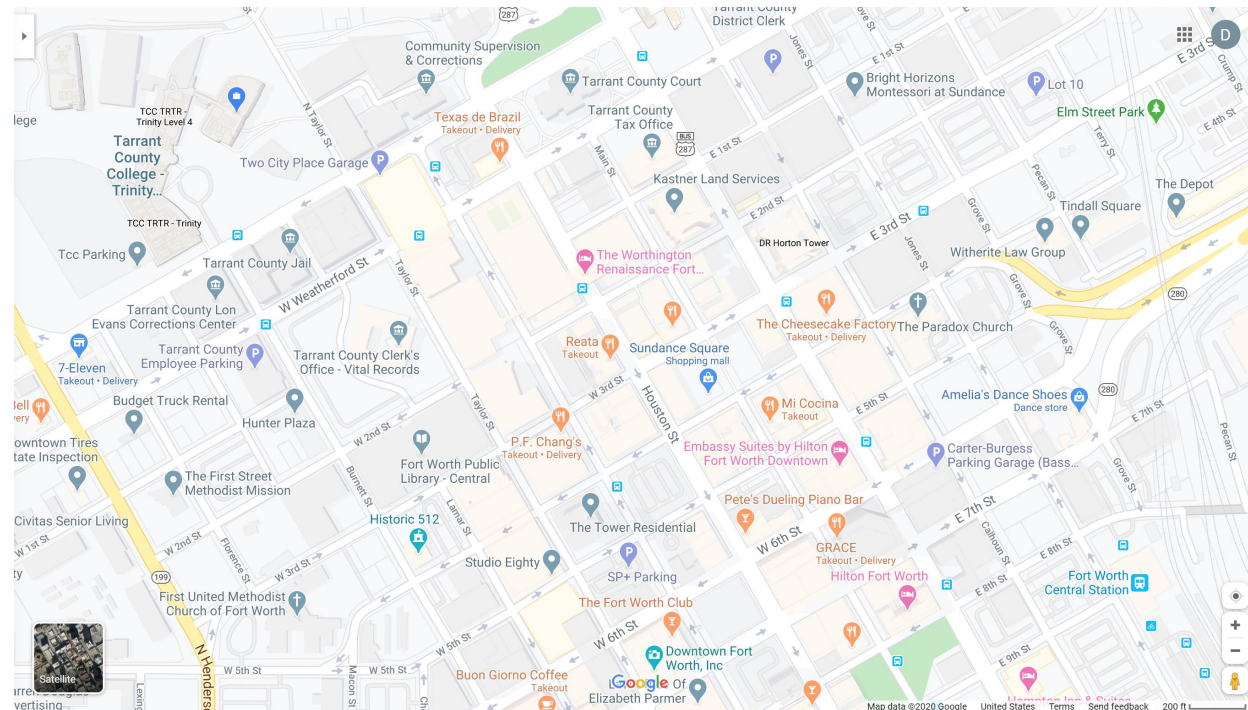
It is a directed acyclic graph (dag).

In a road map, for example, there could be one-way streets.



Graphs

If distances were added to this map, then it would be a weighted directed graph.



Graphs

Directed edges require some more detailed vocabulary for describing them.

A directed edge leaves one vertex and enters another.

When a directed edge leaves vertex A and enters vertex B , then we denote that with (A,B) and the order of the vertices matters.

The number of edges leaving a vertex is its out-degree.

The number of edges entering a vertex is its in-degree.

Graphs

How would you show the relationship between vertex 1 and vertex 3?

(1,3) or (3,1)

How would you show the relationship between vertex 4 and vertex 7?

(4,7) or (7,4)

What is the out-degree of vertex 4?

number of edges leaving a vertex

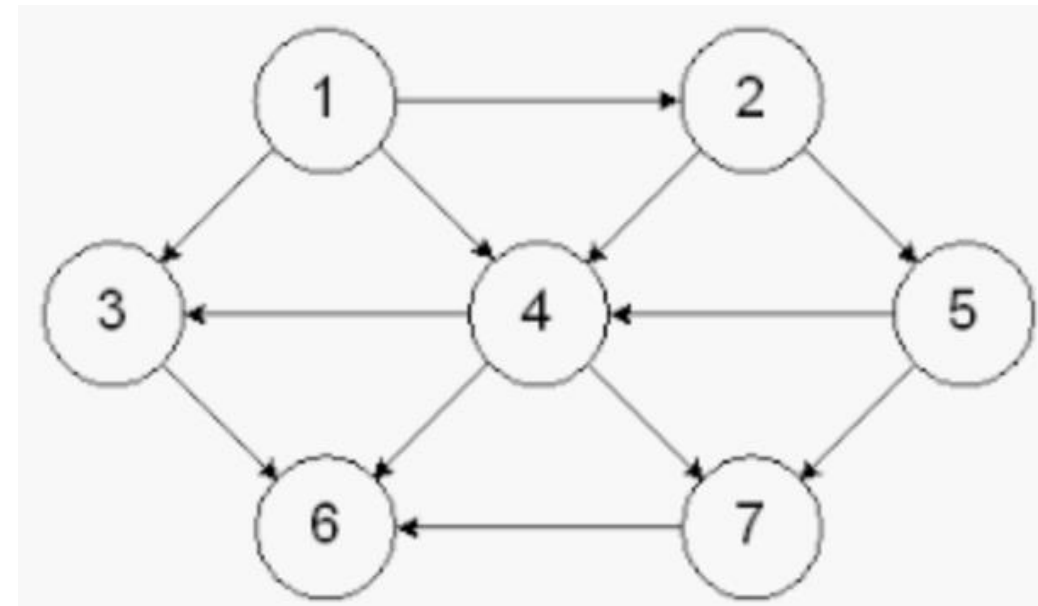
3

What is the in-degree of vertex 4?

number of edges entering a vertex

3

	In	Out
1		
2		
3		
4		
5		
6		
7		



Graphs

When working with graphs, we need to refer to the set of vertices and the set of edges.

Common notation is V for a vertex set and E for an edge set.

When representing a graph or running an algorithm on a graph, we often want to use the sizes of the vertex and edge sets in asymptotic notation.

Graphs

If we used strict set notation, then it would be

$$\Theta(|V|) \quad \text{or} \quad \Theta(\log_2 |E|)$$

But, when using asymptotic notation, you will typically see the $||$ (set notation) dropped.

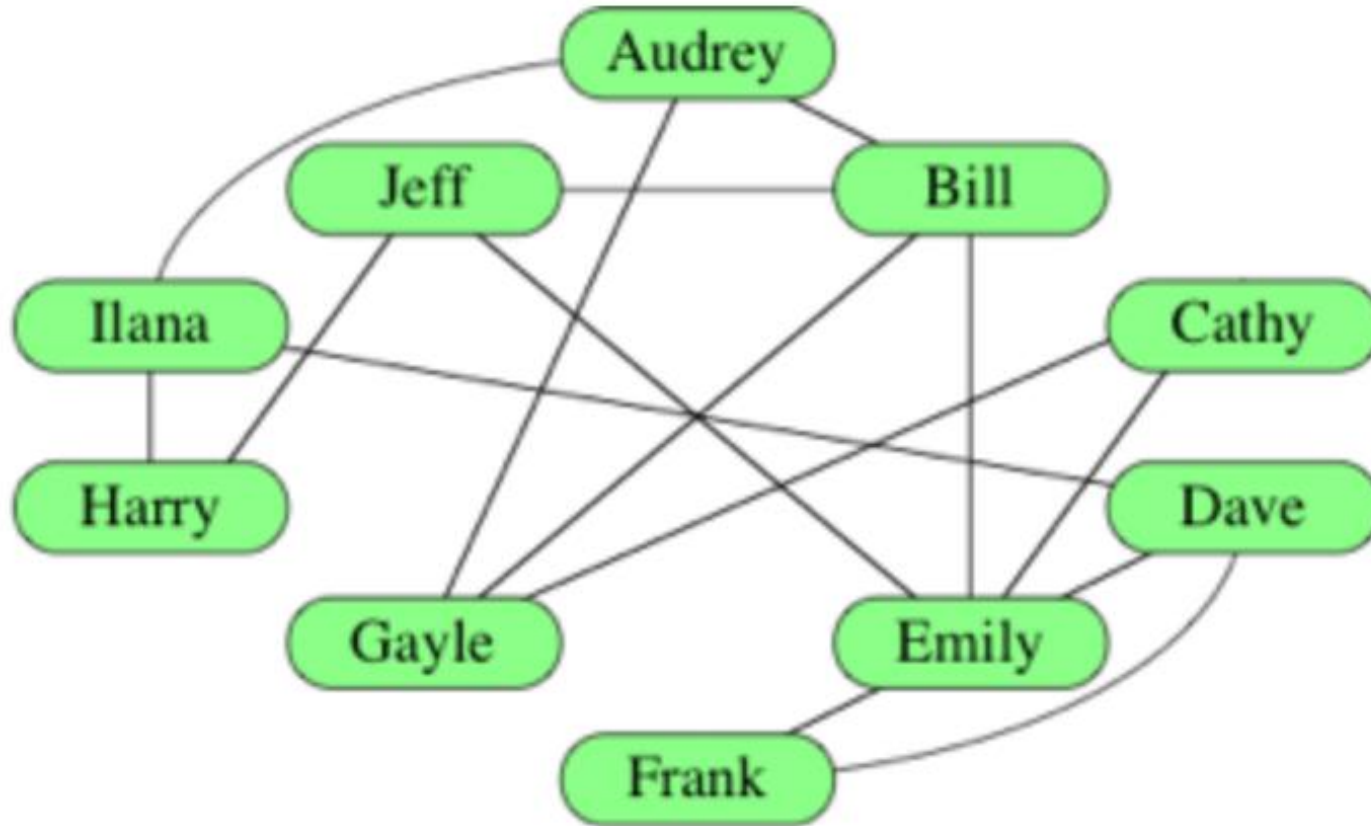
So you will see

$$\Theta(V) \quad \text{or} \quad \Theta(\log_2 E)$$

Graphs

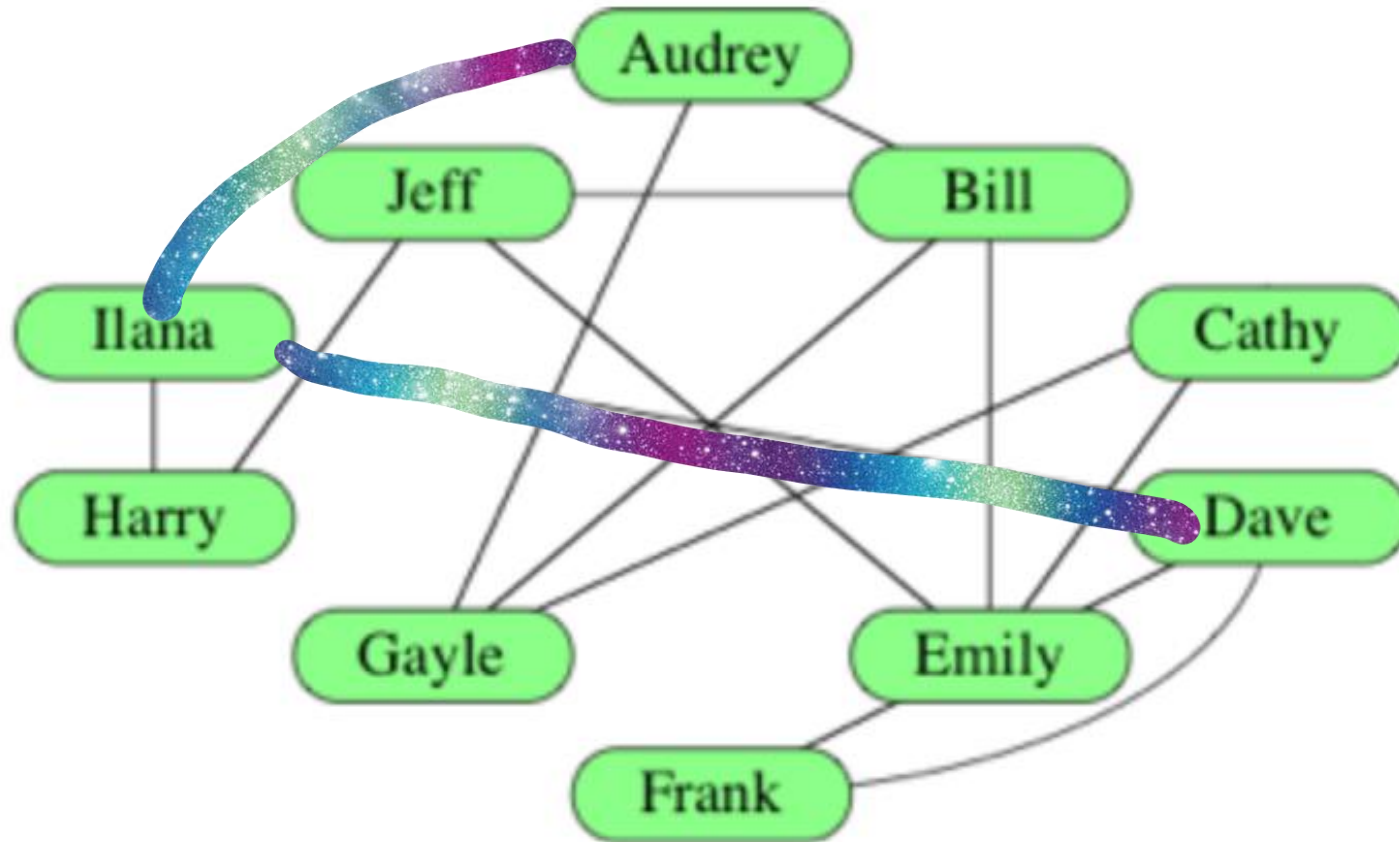
10

How many vertices are in the graph below?



Graphs

What is the shortest path between Audrey and Dave in this graph?



Remember that a cycle cannot contain repeated edges or vertices other than the starting one (Harry).



Harry \rightarrow Ilana \rightarrow Audrey \rightarrow Gayle \rightarrow Bill \rightarrow Jeff \rightarrow Harry

Harry \rightarrow Ilana \rightarrow Audrey \rightarrow Bill \rightarrow Emily \rightarrow Jeff \rightarrow Harry

Harry \rightarrow Jeff \rightarrow Gayle \rightarrow Harry

Harry \rightarrow Jeff \rightarrow Bill \rightarrow Emily \rightarrow Jeff \rightarrow Harry

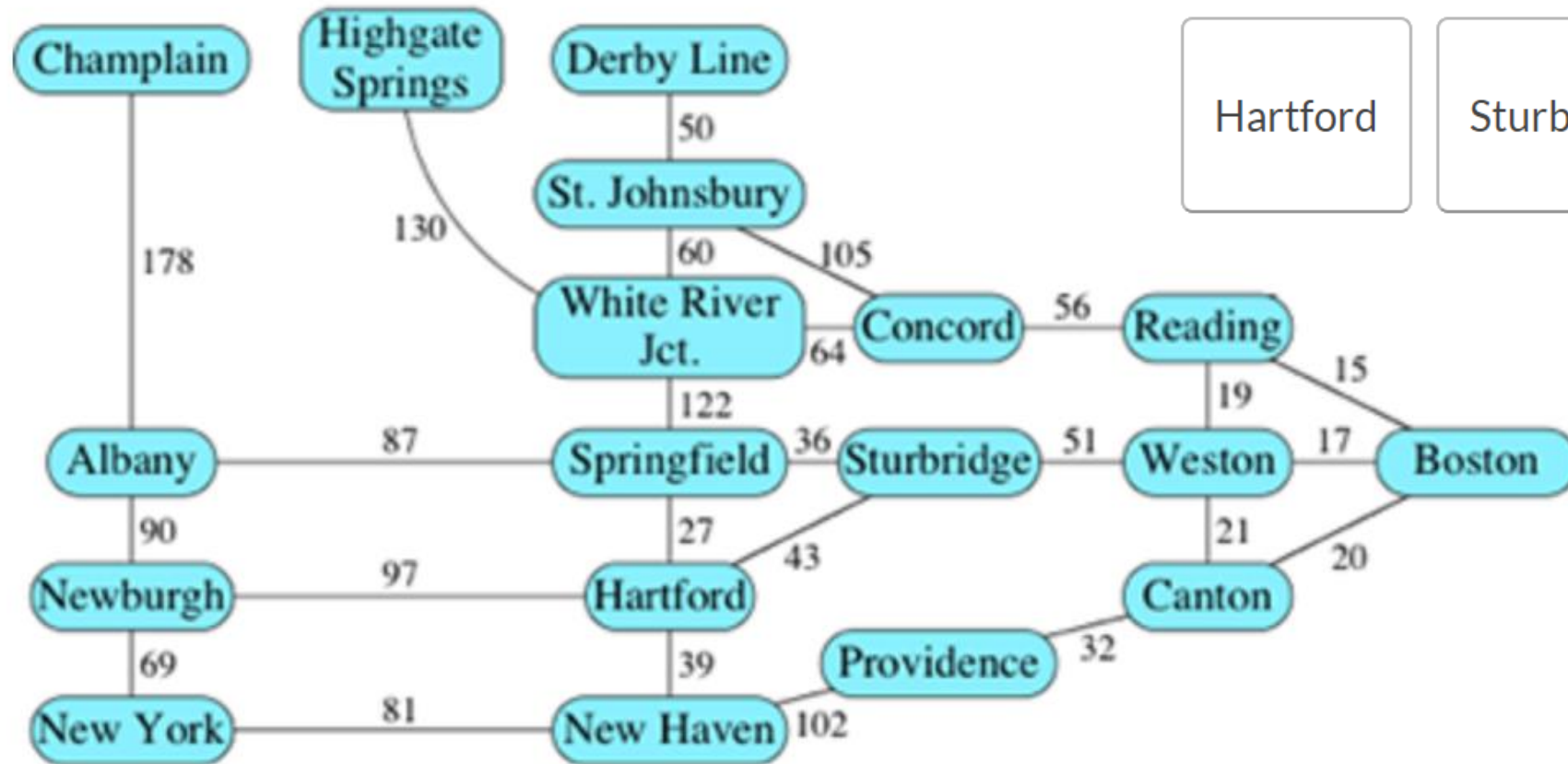
Harry → Jeff → Bill → Audrey → Ilana → Harry

Harry → Jeff → Bill → Gayle → Audrey → Ilana → Harry

Harry \rightarrow Ilana \rightarrow Audrey \rightarrow Ilana \rightarrow Harry

Graphs

What is the shortest path between Hartford and Canton?



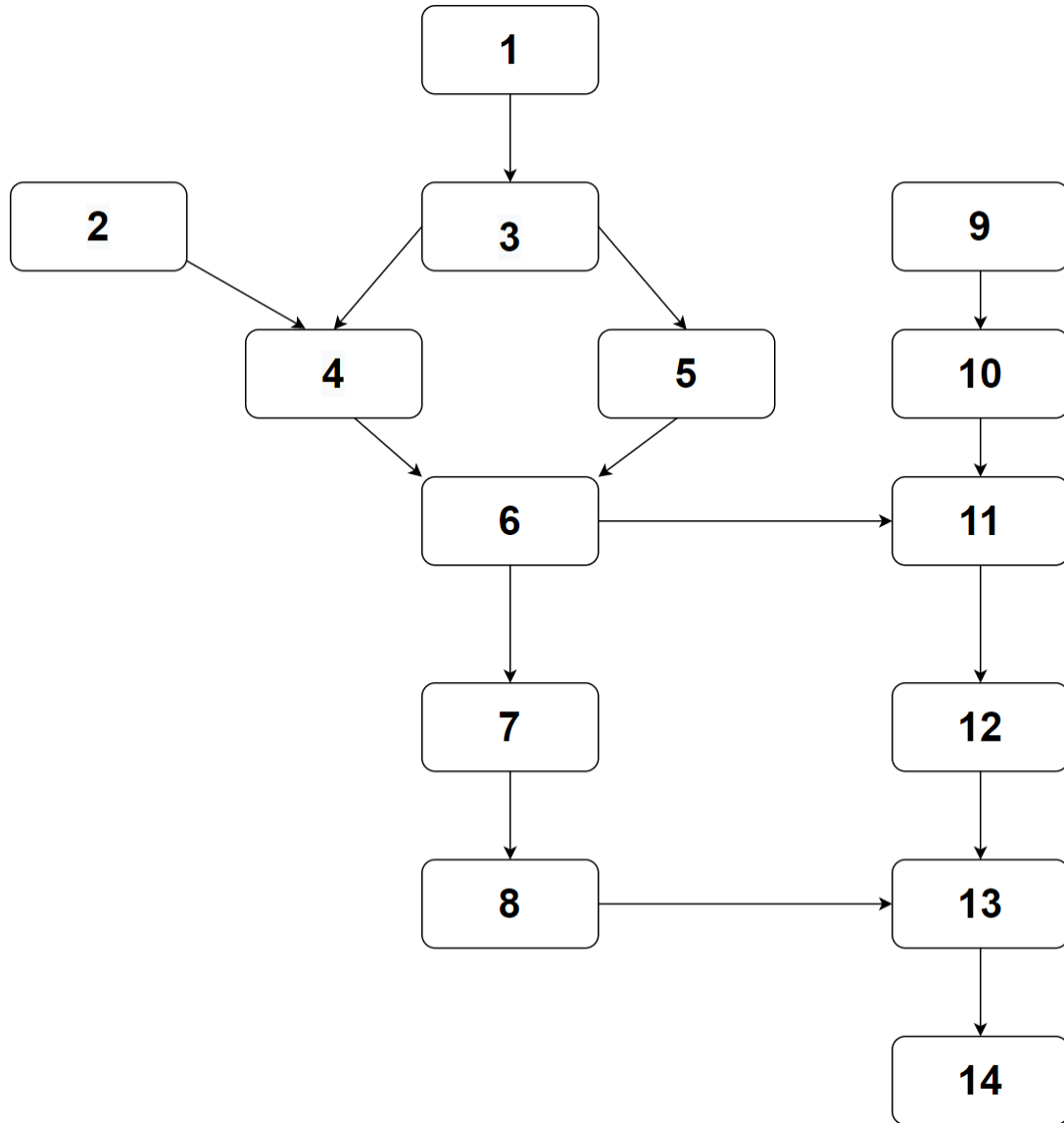
Hartford

Sturbridge

Weston

Canton

Graphs



Vertex	In-degree	Out-degree
1		
3		
2		
4		
5		
6		

Graphs

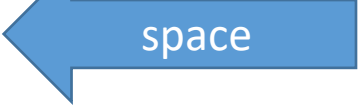
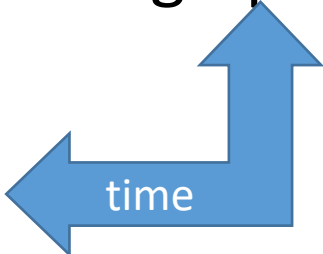
There are several ways to represent graphs, each with its advantages and disadvantages.

Some situations, or algorithms that we want to run with graphs as input, call for one representation and others call for a different representation.

We will learn three ways to represent graphs.

Graphs

There are 3 criteria we will use when choosing how to represent a graph

1. How much memory space is needed for a representation 
2. How long it takes to determine whether a given edge is in the graph
3. How long it takes to find the neighbors of a given vertex. 

Graphs

In order to determine how much memory, or space, we need in each representation, we will use asymptotic notation.

We can use asymptotic notation for purposes other than expressing running times.

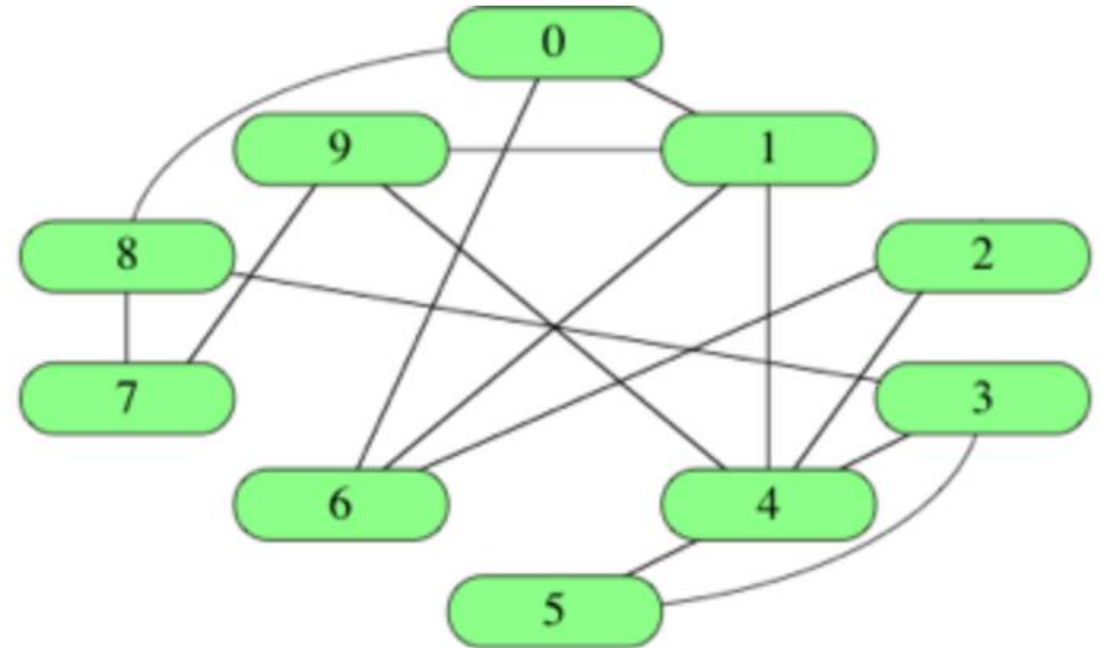
It's really a way to characterize *functions* and a function can describe a running time, an amount of space required, or some other resource.

Graphs

Let's start by identifying vertices by numbers rather than names.

We typically number the $|V|$ vertices from 0 to $|V|-1$

Remember that $|V|$ means the set of vertices.



Graphs

One simple way to represent a graph is an

edge list

which is just a list of $|E|$ edges.

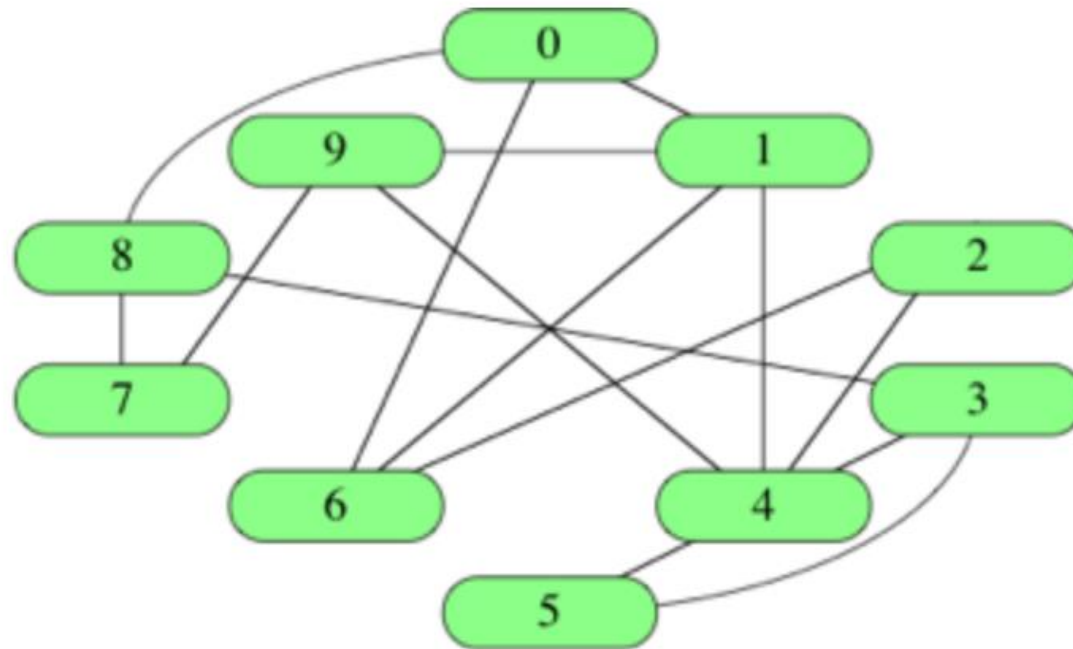
To represent an edge, create an array of two vertex numbers. In an OOP language, an array of objects could be used. An array of structs could also be used to create an edge list. A 2D array could be used also.

The total space needed by an edge list is $\Theta(E)$.

Graphs

This graph could be represented with an edge list like this...

$\{\{0,1\}, \{4,5\}, \{2,4\}, \{0,6\}, \{1,4\}, \{3,8\}, \{1,6\}, \{2,6\}, \{3,4\}, \{0,8\}, \{3,5\}, \{4,9\}, \{7,8\}, \{1,9\}, \{7,9\}\}$



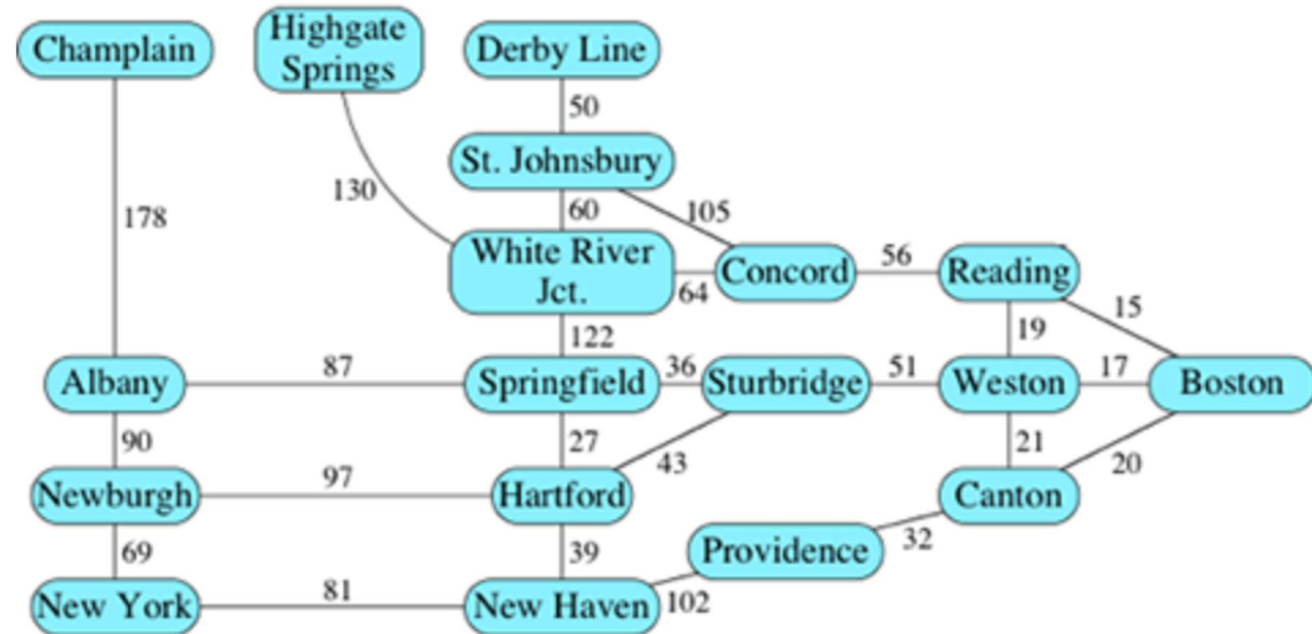
Graphs

What if the edges had weight?

If we are using objects in OOP, then we could add an attribute to the class.

If we are using arrays, we could add another dimension to the array.

If we are using structs, then we could add another member to the struct.



Graphs

Edge lists are simple but...

What if we wanted to find whether the graph contains a particular edge?

How would we find it?

Are the edges in any particular order?

$\{\{0,1\}, \{4,5\}, \{2,4\}, \{0,6\}, \{1,4\}, \{3,8\}, \{1,6\}, \{2,6\}, \{3,4\}, \{0,8\}, \{3,5\}, \{4,9\}, \{7,8\}, \{1,9\}, \{7,9\}\}$

A linear search would require $O(E)$ run time.

Graphs

$\{\{0,1\}, \{4,5\}, \{2,4\}, \{0,6\}, \{1,4\}, \{3,8\}, \{1,6\}, \{2,6\}, \{3,4\}, \{0,8\}, \{3,5\}, \{4,9\}, \{7,8\}, \{1,9\}, \{7,9\}\}$

A linear search would require $O(E)$ run time.

What could we change about the list to make the runtime $O(\log_2 E)$?

Binary search has a runtime of $O(\log_2 n)$...

Could we organize/sort our edge list so that a binary search could be used?

Graphs

Could we organize/sort our edge list so that a binary search could be used?

We could sort the list by the 1st vertex and then the 2nd vertex.

{ {0,1}, {0,6}, {0,8}, {1,4}, {1,6}, {1,9}, {2,4}, {2,6}, {3,4}, {3,5}, {3,8}, {4,5}, {4,9}, {7,8}, {7,9} }

How would the binary search work?

It would still pick the middle element – in this example, array element {2,6} would be in the middle. To decide whether to choose the right half of the array or the left requires a little more work.

Graphs

$\{\{0,1\}, \{0,6\}, \{0,8\}, \{1,4\}, \{1,6\}, \{1,9\}, \{2,4\}, \{2,6\}, \{3,4\}, \{3,5\}, \{3,8\}, \{4,5\}, \{4,9\}, \{7,8\}, \{7,9\}\}$

If $\{2,6\}$ is the middle element, how do we decide whether to go left or right?

Edge $(V1, V2)$ is less than another edge $(V3, V4)$

if $(V1 < V3)$

OR

$(V1 == V3) \text{ AND } (V2 < V4)$

Edge (V1, V2) is less than another edge (V3,V4)

Graphs

if (V1 < V3)

OR

(V1 == V3) AND (V2 < V4)

{0,1}, {0,6}, {0,8}, {1,4}, {1,6}, {1,9}, {2,4}, {2,6}, {3,4}, {3,5}, {3,8}, {4,5}, {4,9}, {7,8}, {7,9} }

Is edge {0,8} less than edge {2,6} ?

V1 = 0 and V2 = 8 and V3 = 2 and V4 = 6

if (0 < 2) TRUE

OR

(0 == 2) AND (8 < 6) F AND T = FALSE

FALSE OR FALSE is FALSE so edge {0,8}

is to the left of/less than {2,6}

Is edge {2,4} less than edge {2,6}?

V1 = 2 and V2 = 4 and V3 = 2 and V4 = 6

if (2 < 2) FALSE

OR

(2 == 2) AND (4 < 6) TRUE

FALSE OR TRUE is TRUE so edge {2,4} is to the left of/less than {2,6}

Edge (V1, V2) is less than another edge (V3,V4)

Graphs

if (V1 < V3)

OR

(V1 == V3) AND (V2 < V4)

{0,1}, {0,6}, {0,8}, {1,4}, {1,6}, {1,9}, {2,4}, {2,6}, {3,4}, {3,5}, {3,8}, {4,5}, {4,9}, {7,8}, {7,9} }

Is edge {4,9} less than edge {2,6}?

V1 = 4 and V2 = 9 and V3 = 2 and V4 = 6

if (4 < 2) FALSE

OR

(4 == 2) AND (9 < 6) F AND F = FALSE

FALSE OR FALSE is FALSE so edge {4,9}

is NOT less so it is to the right of {2,6}

Is edge {8,4} less than edge {2,6}?

V1 = 8 and V2 = 4 and V3 = 2 and V4 = 6

if (8 < 2) FALSE

OR

(8 == 2) AND (4 < 6) F AND T = FALSE

FALSE OR FALSE is FALSE so edge {8,4} is NOT
to less so it is to the right of {2,6}

Edge (V1, V2) is less than another edge (V3,V4)

Graphs

if (V1 < V3)

OR

(V1 == V3) AND (V2 < V4)

{{0,1}, {0,6}, {0,8}, {1,4}, {1,6}, {1,9}, {2,4}, {2,6}, {3,4}, {3,5}, {3,8}, {4,5}, {4,9}, {7,8}, {7,9} }

Is edge {7,9} to the left or right?

V1 = 7 and V2 = 9 and V3 = 2 and V4 = 6

7 < 2 is FALSE so {7,9} is to the right

{{3,4}, {3,5}, {3,8}, {4,5}, {4,9}, {7,8}, {7,9}}

New middle is {4,5}

7 < 4 is FALSE so {7,9} is to the right

{{4,9}, {7,8}, {7,9}}

New middle is {7,8}

7 \nless 7 but 7==7 AND 9 < 8 FALSE so right

{{7,9}}

New middle is {7,9}

FOUND IT

Graphs

Adjacency Matrices

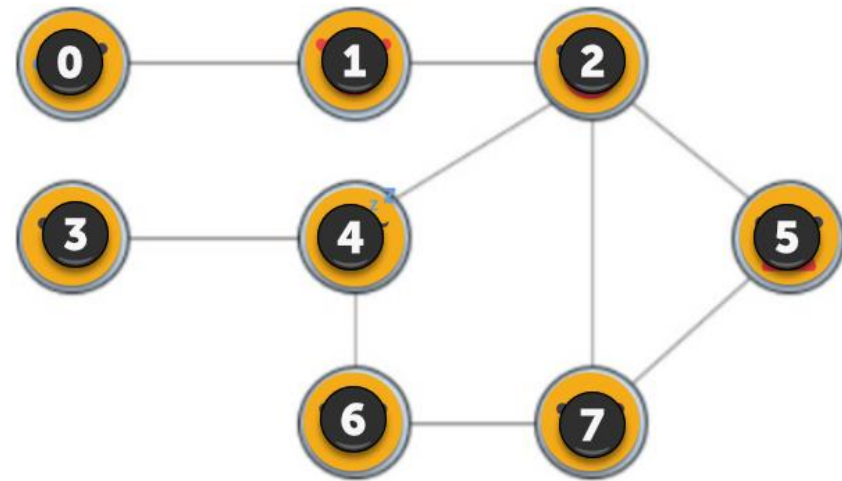
For a graph with $|V|$ vertices, an **adjacency matrix** is a $|V| \times |V|$ matrix of zeroes and ones.

The entry in row i and column j is **1 if and only if** the edge (i,j) is in the graph. A value of 0 indicates there is no edge between i and j .

To indicate an edge weight, put the weight in the row i , column j entry and reserve a special value (-1 for example) to indicate an absent edge.

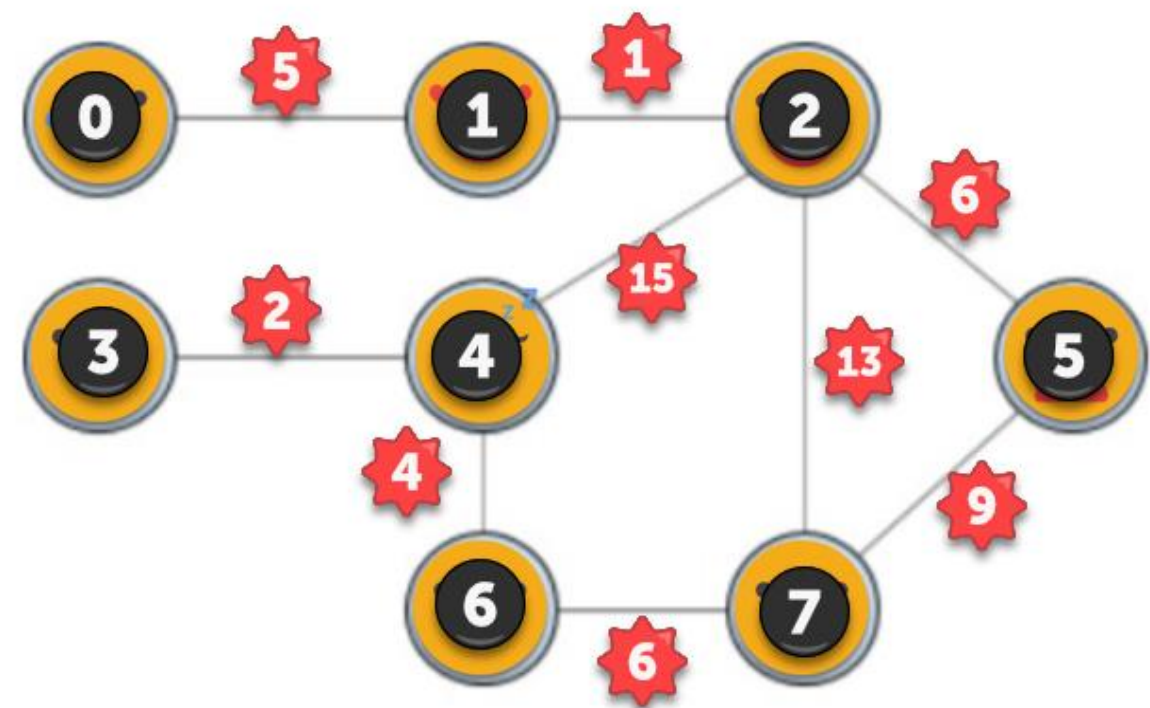
Graphs

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



Graphs

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



Graphs

We can determine if an edge is present in the adjacency matrix just by looking up the vertices of the edge – an edge's i,j value.

If we create a 2D array named AM , then we could just look up $AM[i][j]$

This look up takes a constant amount of time.

Sounds pretty good?

Graphs

Disadvantages of adjacency matrix

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

Adjacency matrix requires $\Theta(V^2)$ space to store a graph

Our array was mostly zeroes – we filled in 18 of 64 cells

Our graph is **sparse** – relatively few edges

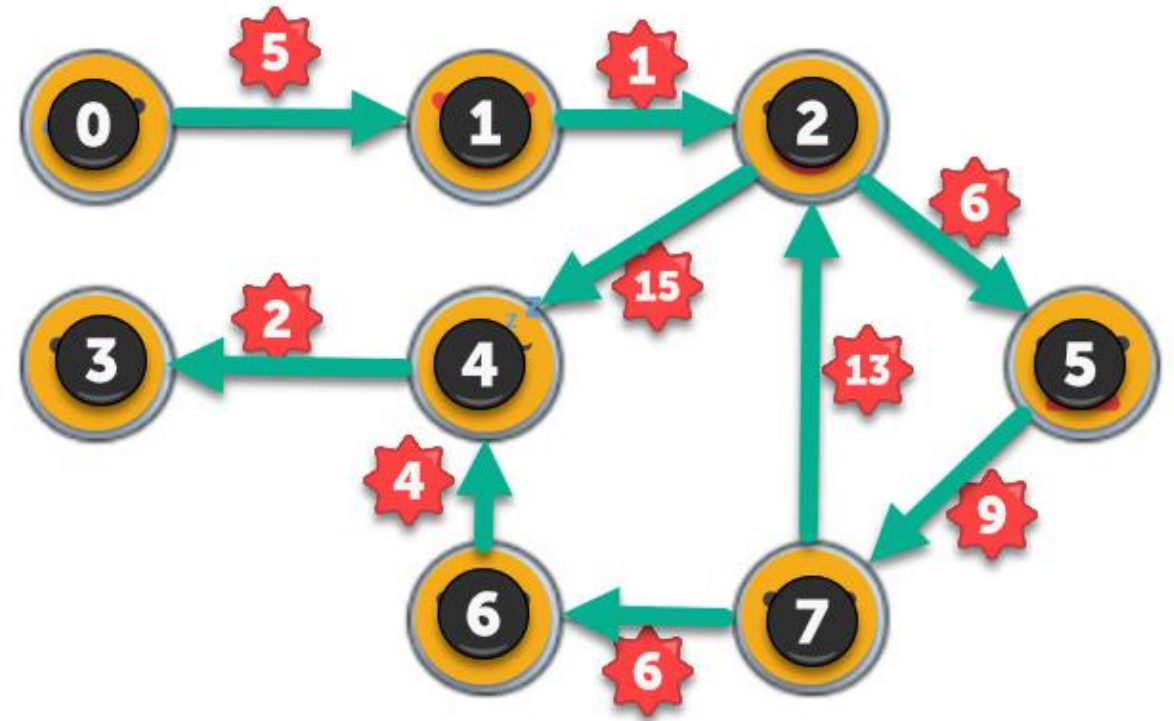
To find which vertices are adjacent to each other, we have to look at all $|V|$ entries in row i , even if only a few vertices are adjacent to vertex i .

Graphs

Undirected graph's adjacency matrix is symmetric

Directed graph's adjacency matrix need not be symmetric

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



Graphs

Adjacency Lists

We can combine adjacency matrices with edge lists.

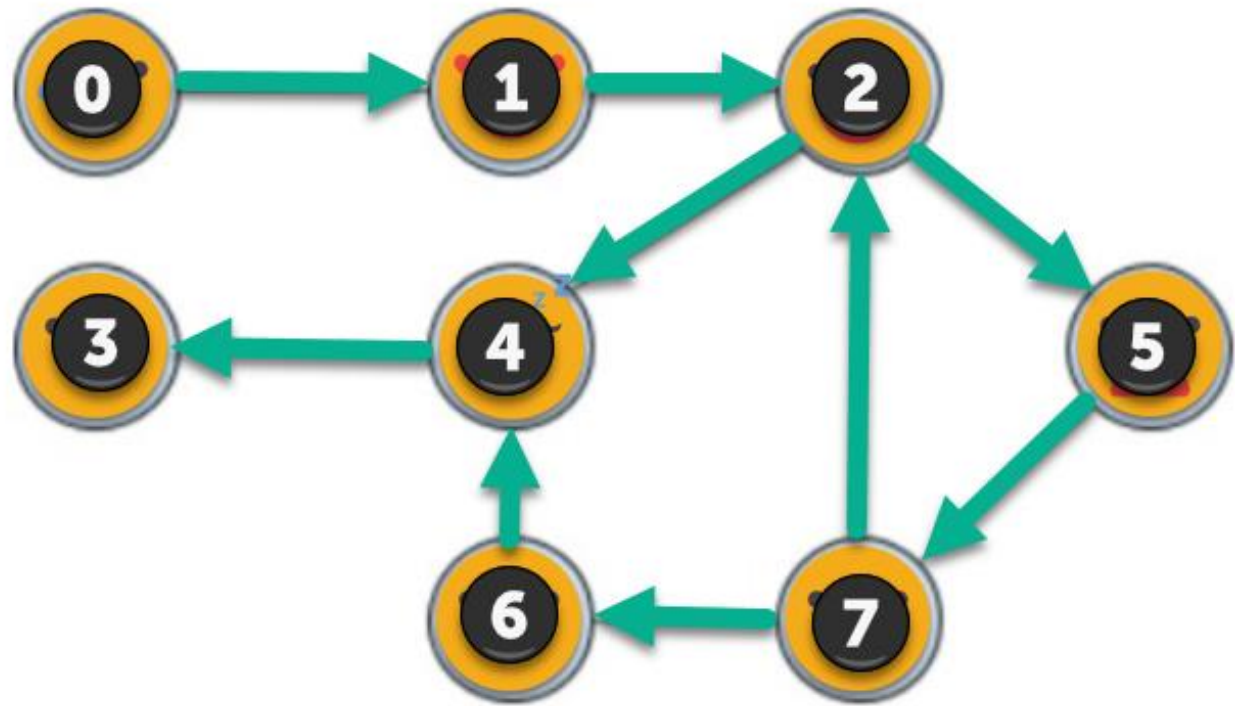
For each vertex i , store an array of the vertices adjacent to it.

We will need an array of $|V|$ adjacency lists where there is one adjacency list per vertex.

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

Graphs

Adjacency Lists



Graphs

Adjacency Lists

Vertex numbers in an adjacency list are not required to appear in any particular order.

There is a benefit to listing them in increasing order despite the cost of doing the ordering at the time of creating the adjacency list.

Graphs

Adjacency Lists

How much time does it take to find a vertex's adjacency list?

A constant amount of time since we are just going to an array index.

To find if an edge(x,y) is in a graph, we go to x 's adjacency list in constant time and then look for y in x 's adjacency list.

Graphs

Adjacency Lists

So how long does the worst case search take for edge (x,y) ?

How many possible entries could any one vertex have?

Let's recall this definition

The number of edges incident on a vertex is the degree of the vertex.

Use d to represent the degree of a vertex

Graphs

Adjacency Lists

So how long does the worst case take?

$\Theta(d)$

where d is the degree of vertex x because that's how long vertex x 's adjacency list is.

What is the highest value (degree) possible for any vertex in $|V|$?

Graphs

Adjacency Lists

What is the highest value (degree) possible for any vertex in $|V|$?

Any vertex in a graph could be adjacent (share an edge with) to every other vertex.

so the degree of any vertex could be as high as $|V| - 1$ and as low as 0^*

*a vertex can be isolated with no incident edges

Graphs

Adjacency Lists

How much space does an adjacency list use?

Every vertex ($|V|$) can have a list of adjacent vertices ($|V|-1$)

For an undirected graph, the adjacency list will contain $2|E|$ elements.

For a directed graph, the adjacency list will contain $|E|$ edges – one element per directed edge.

Graphs

Adjacency Lists

How much space does an adjacency list use?

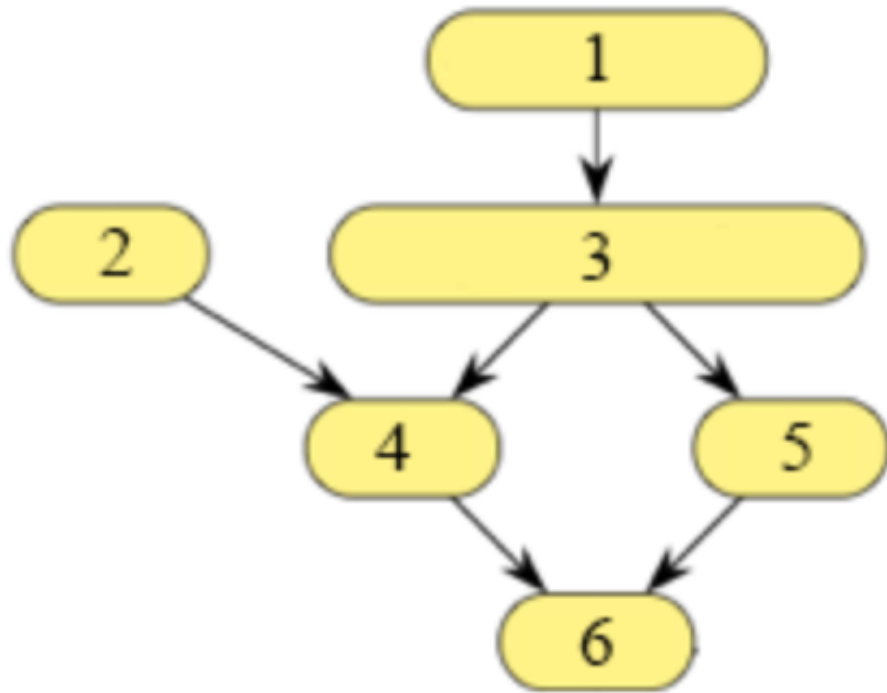
An adjacency list is a list of lists.

Each list corresponds to a vertex u and contains a list of edges (x, y) that originate from x .

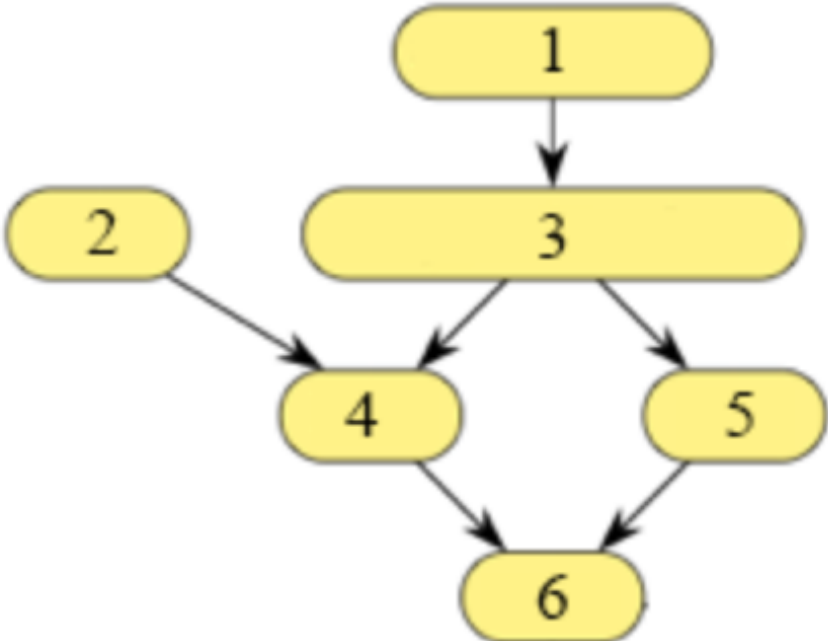
An adjacency list takes up $\Theta(V + E)$ space.

Worst case is when the graph is dense and $E = \Theta(V^2)$

Given the following directed graph, how would you represent it with an edge list?

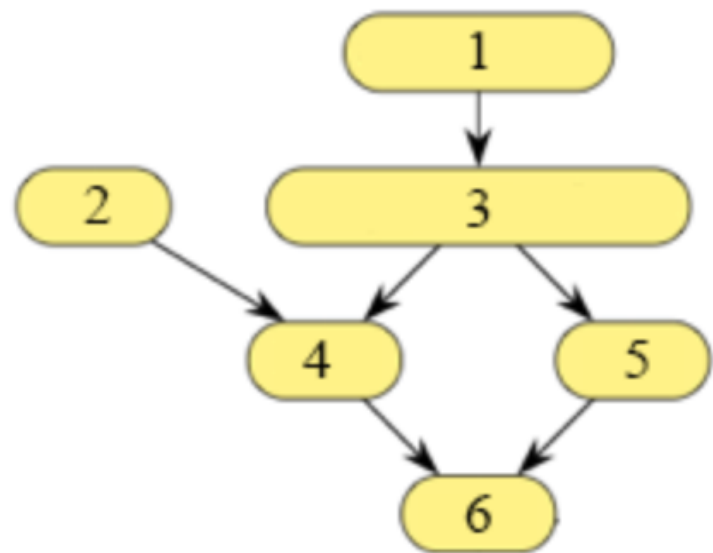


Given the following directed graph, how would you represent it with an adjacency matrix?



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

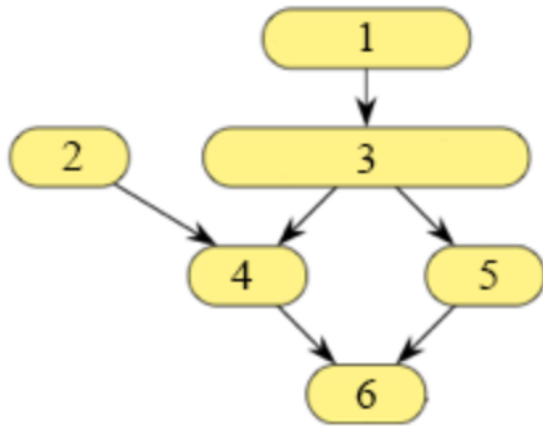
Given the following directed graph, how would you represent it with an adjacency list?



1	→							
2	→							
3	→							
4	→							
5	→							
6	→							

For each vertex i , write the vertices that are adjacent to it, one in each cell. Leave cells empty that you don't need.

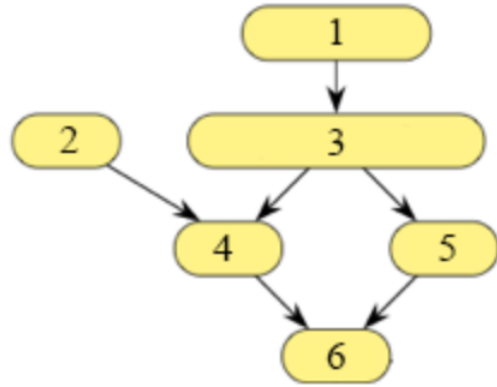
We've seen three ways to store graphs - edge lists, adjacency matrices, and adjacency lists. For an directed graph like the one shown below, how much space do we need for each type of storage?



Assuming E is the number of edges and V is the number of vertices, categorize the space below:

	$\Theta(E)$	$\Theta(V + E)$	$\Theta(V^2)$
edge list	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>
adjacency matrix	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
adjacency list	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>

We've seen three ways to store graphs - edge lists, adjacency matrices, and adjacency lists. For an directed graph like the one shown below, how much time would it take to search for a particular edge through each way of storing the graph?



Assuming E is the number of edges, V is the number of vertices, and d is the degree of each vertex, categorize the time taken below:

	$O(1)$	$O(d)$	$O(E)$
edge list (unordered)	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
adjacency matrix	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>
adjacency list	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>