GRAPHS

David Kauchak
CS 62 – Spring 2020

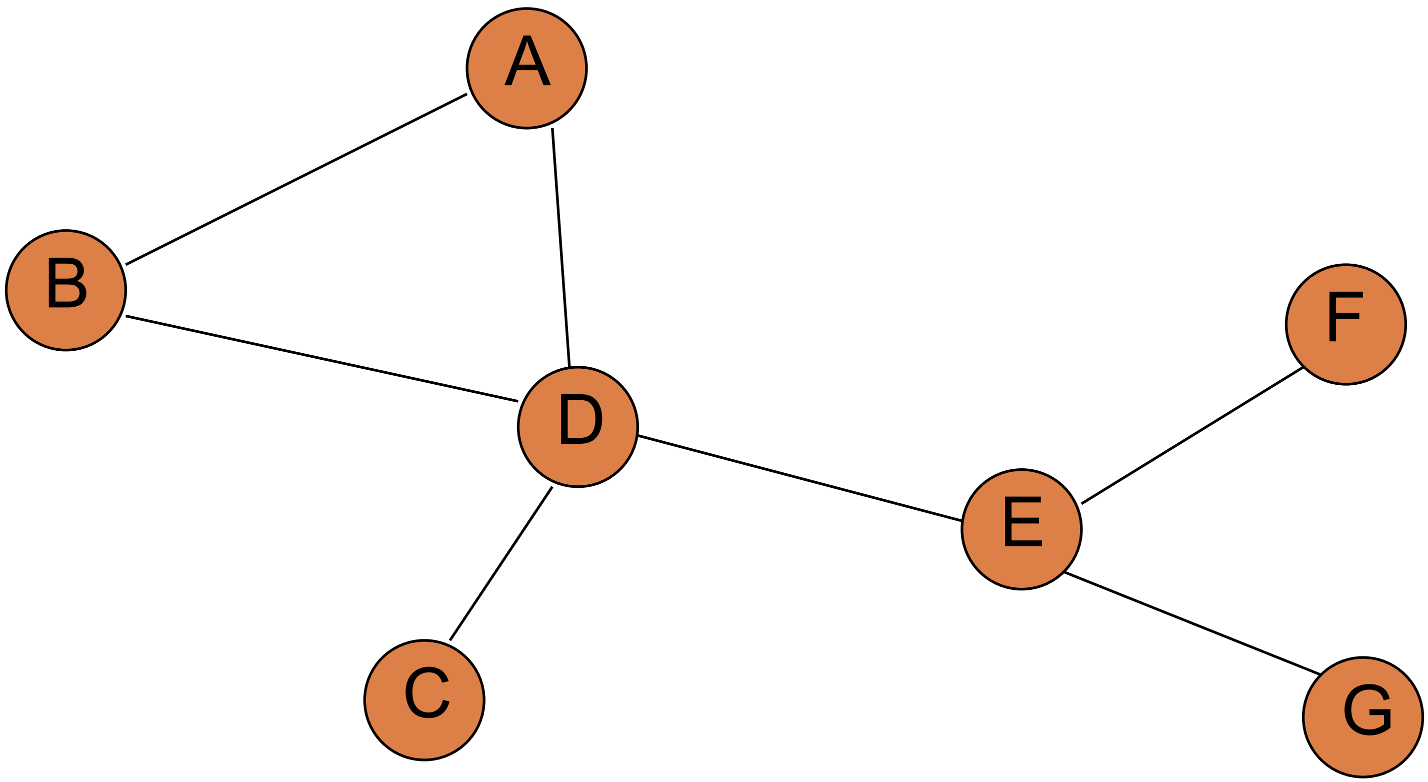
Admin

Last assignment out soon!

- Familiarize yourself with the problem
- Take a look at the starter code
- Probably won’t be able to start coding until Tue.

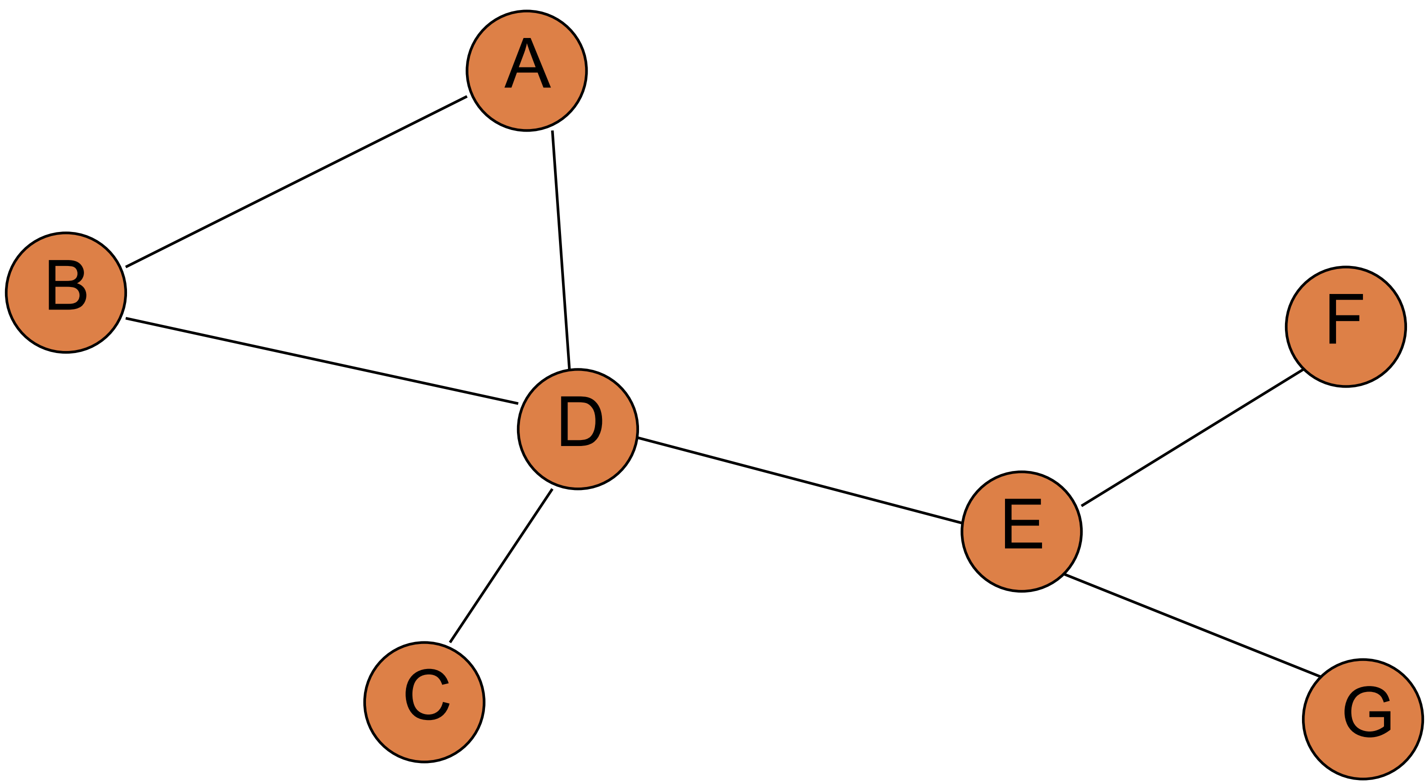
Graphs

A mathematical model consisting of a set of nodes/vertices and edges



Graphs

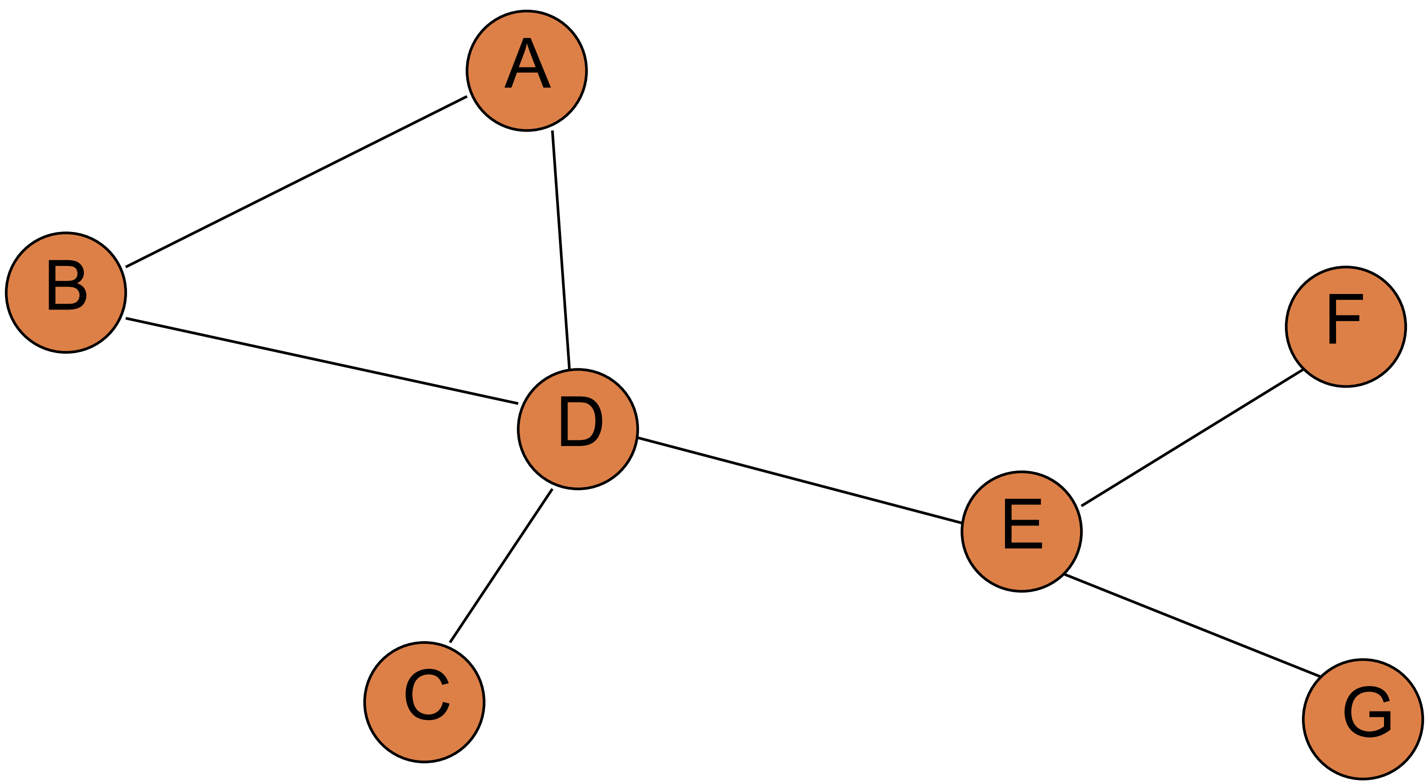
A graph is a set of vertices V and a set of edges (u,v) $\in$ E where u,v $\in$ V



Graphs

V = {A, B, C, D, E, F, G}

E = {(A,B), (A,D), (B,D), (C,D), (D,E), (E,F), (E,G)}



When do we see graphs in real life problems?

Transportation networks (flights, roads, etc.)

Communication networks

Web

Social networks

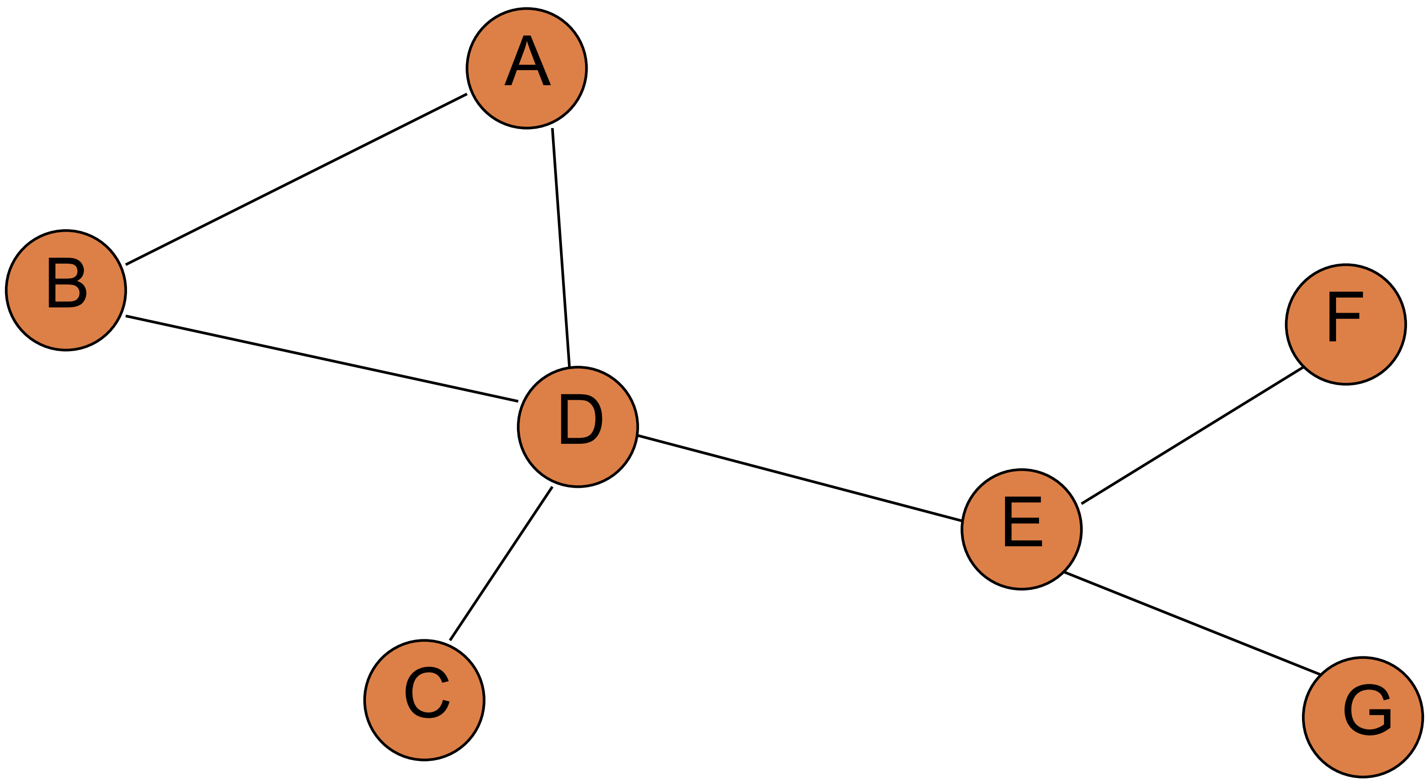
Circuit design

Bayesian networks

Graphs

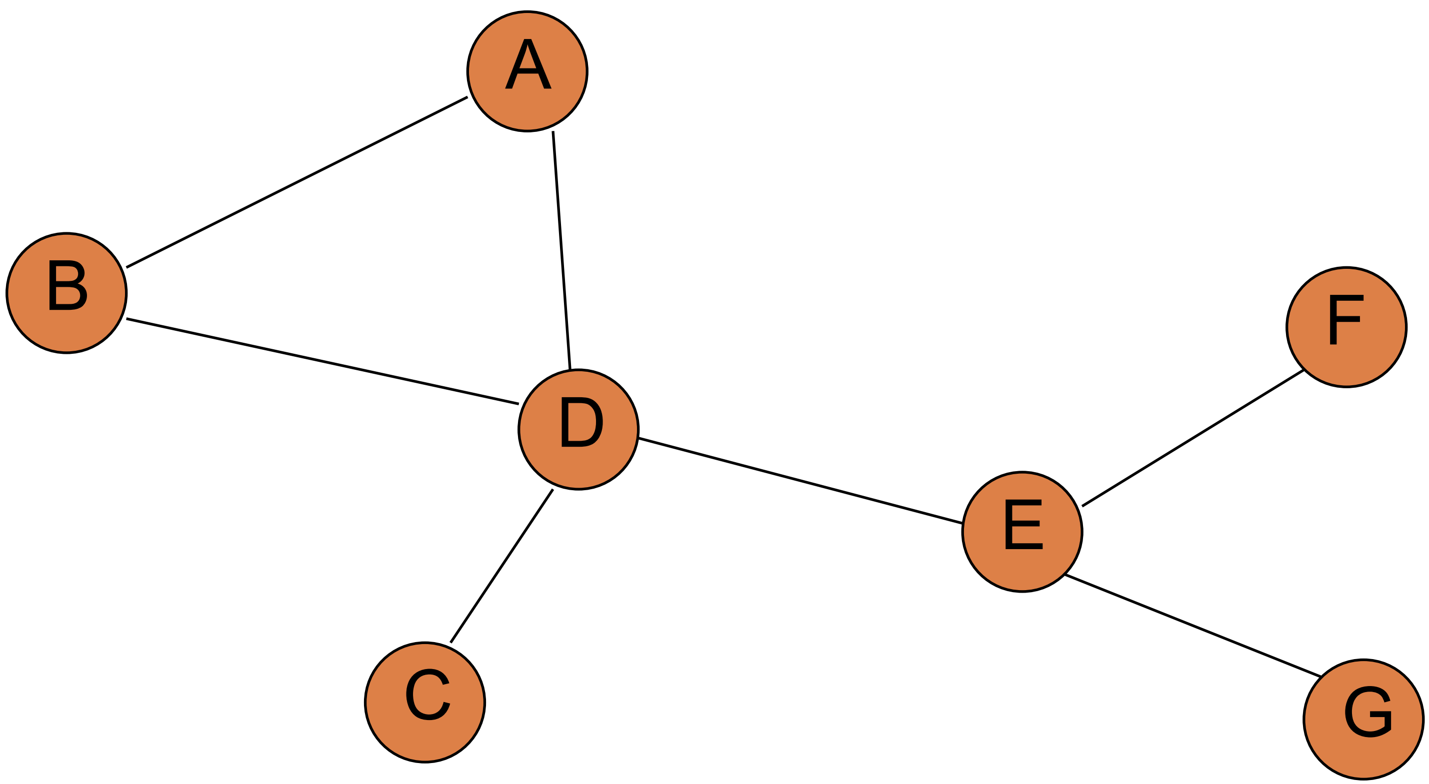
How do graphs differ?

What are graph characteristics we might care about?



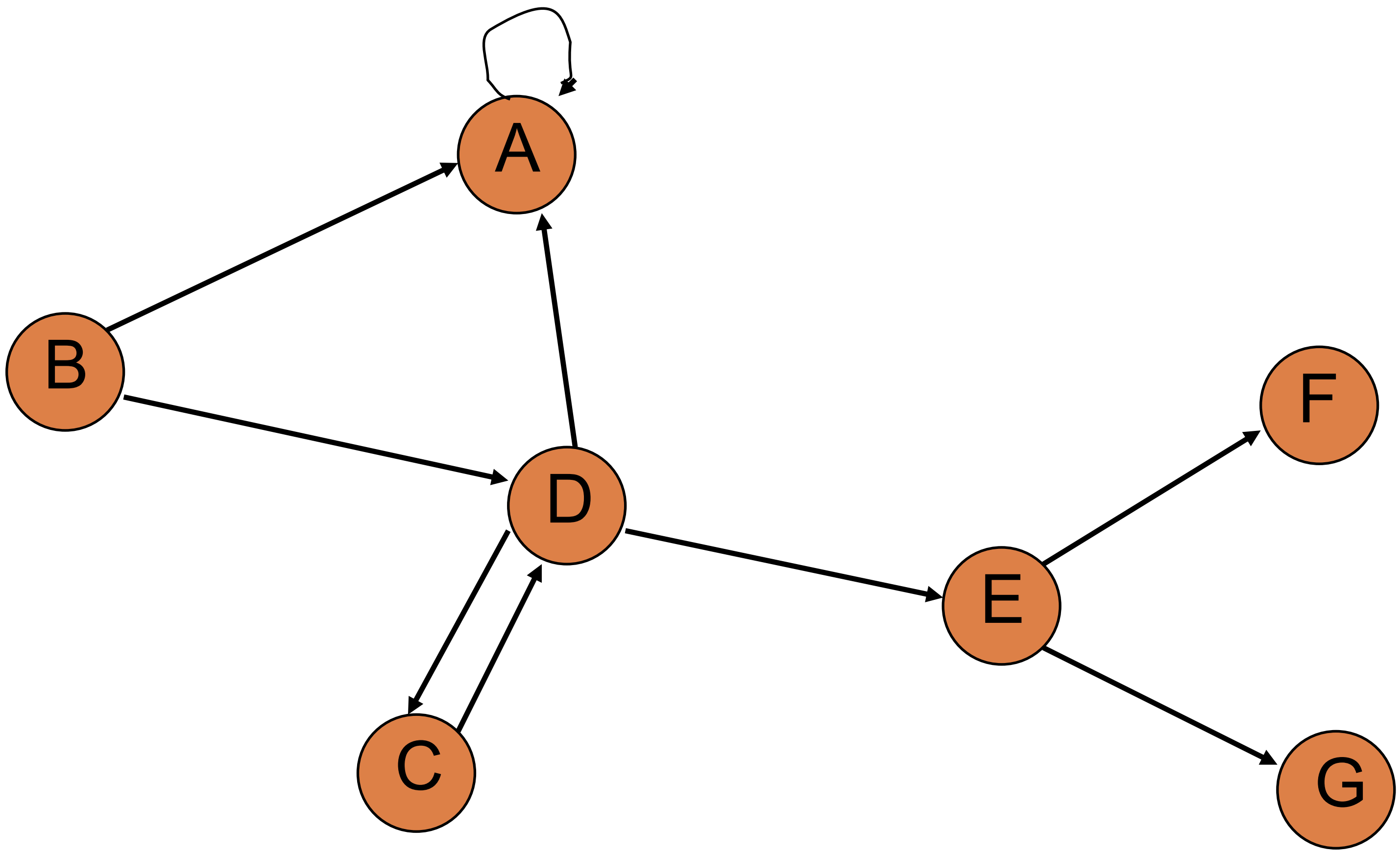
Different types of graphs

Undirected – edges do not have a direction



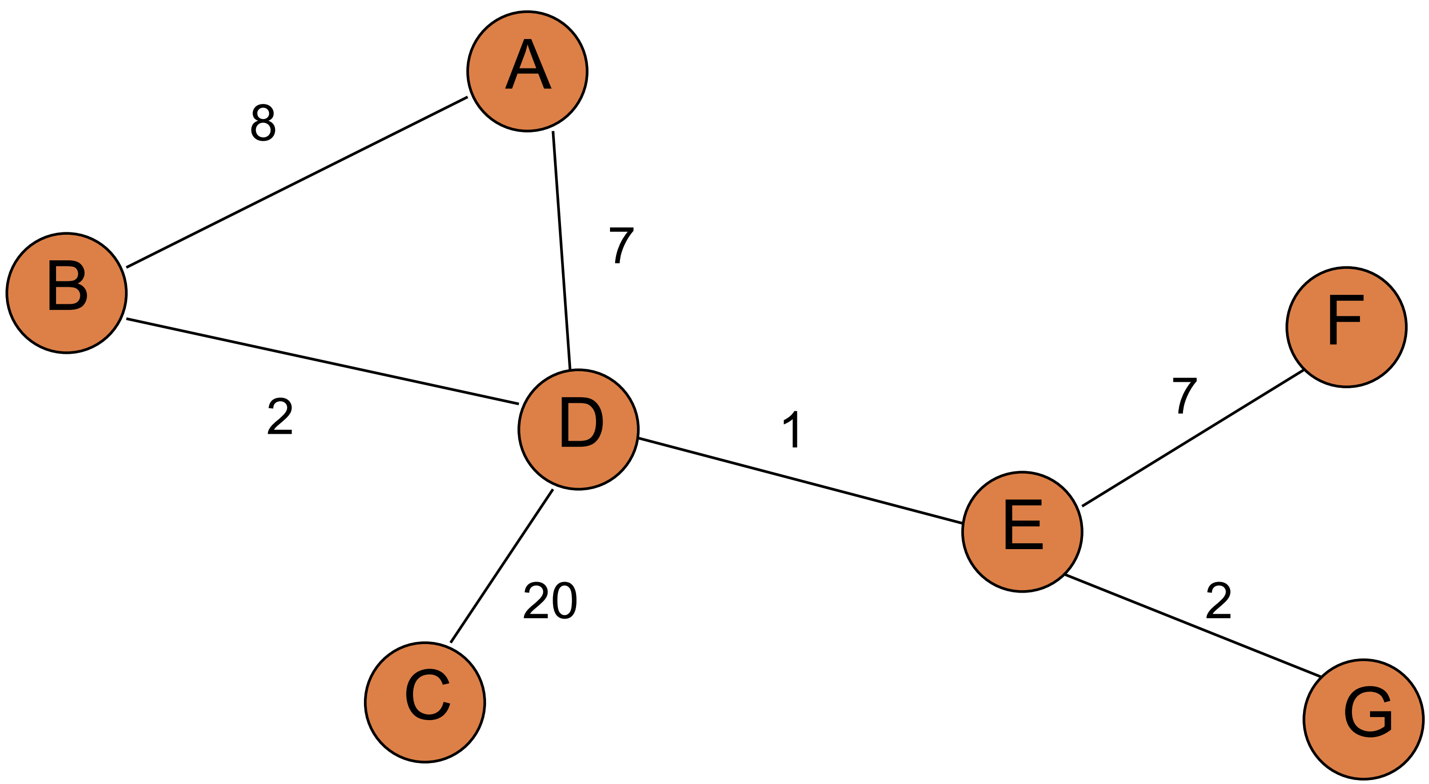
Different types of graphs

Directed – edges **do** have a direction



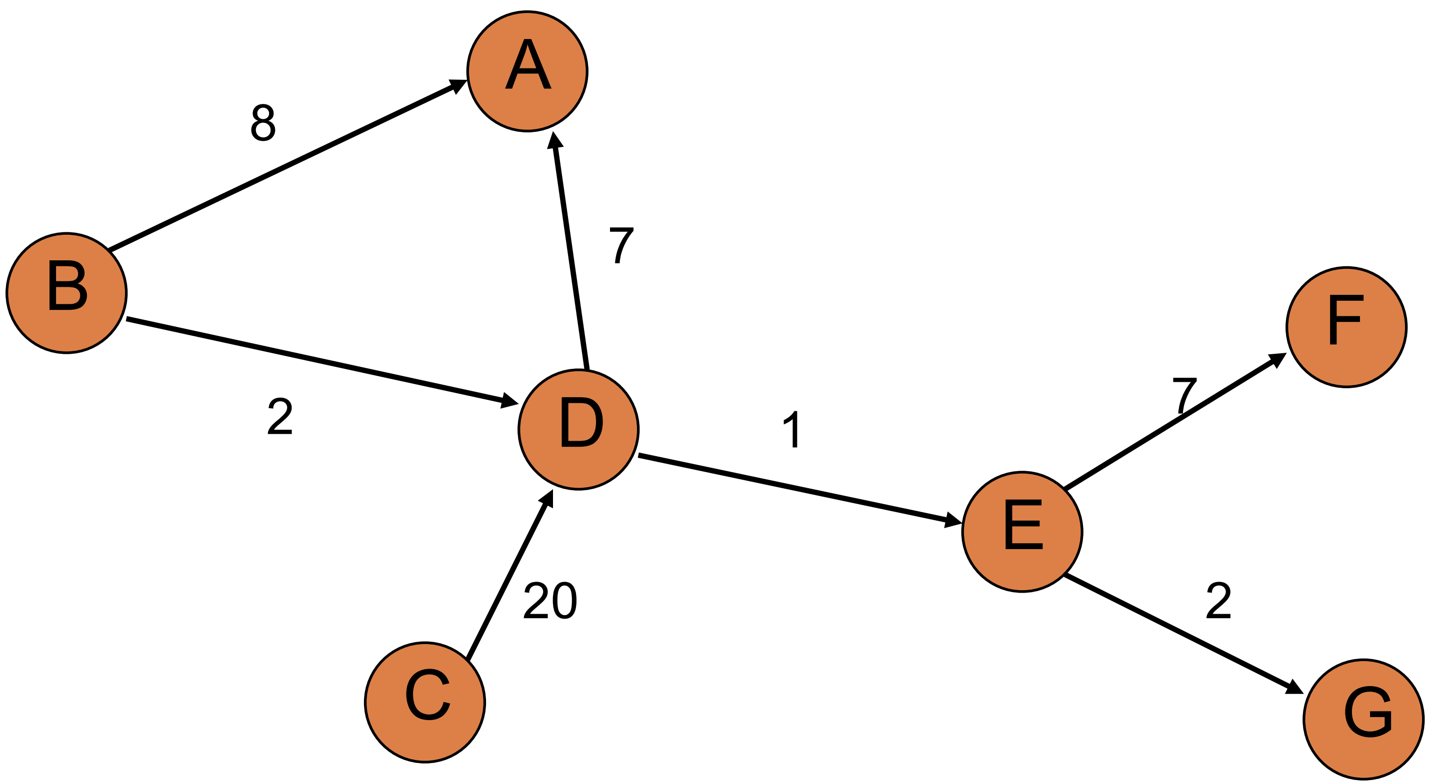
Different types of graphs

Weighted – edges have an associated weight



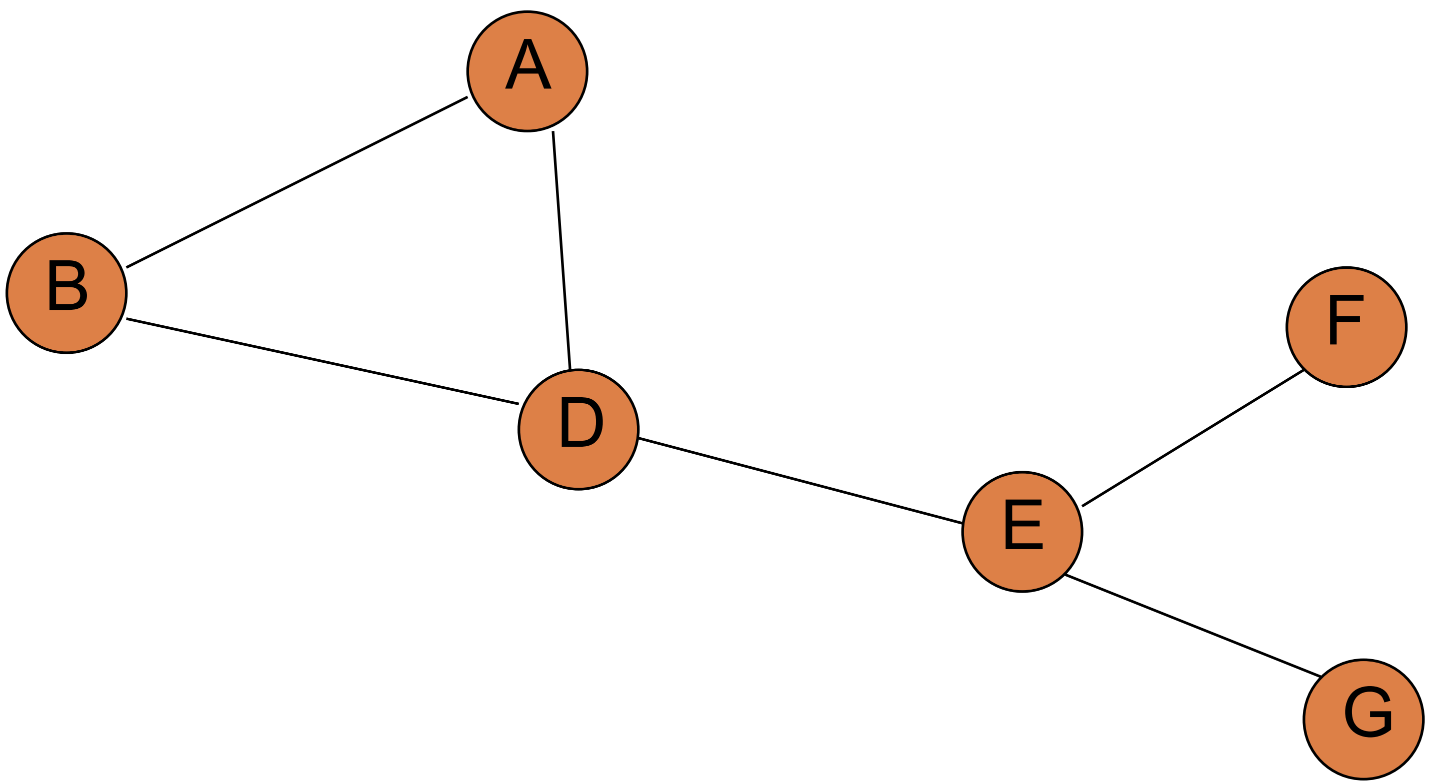
Different types of graphs

Weighted – edges have an associated weight



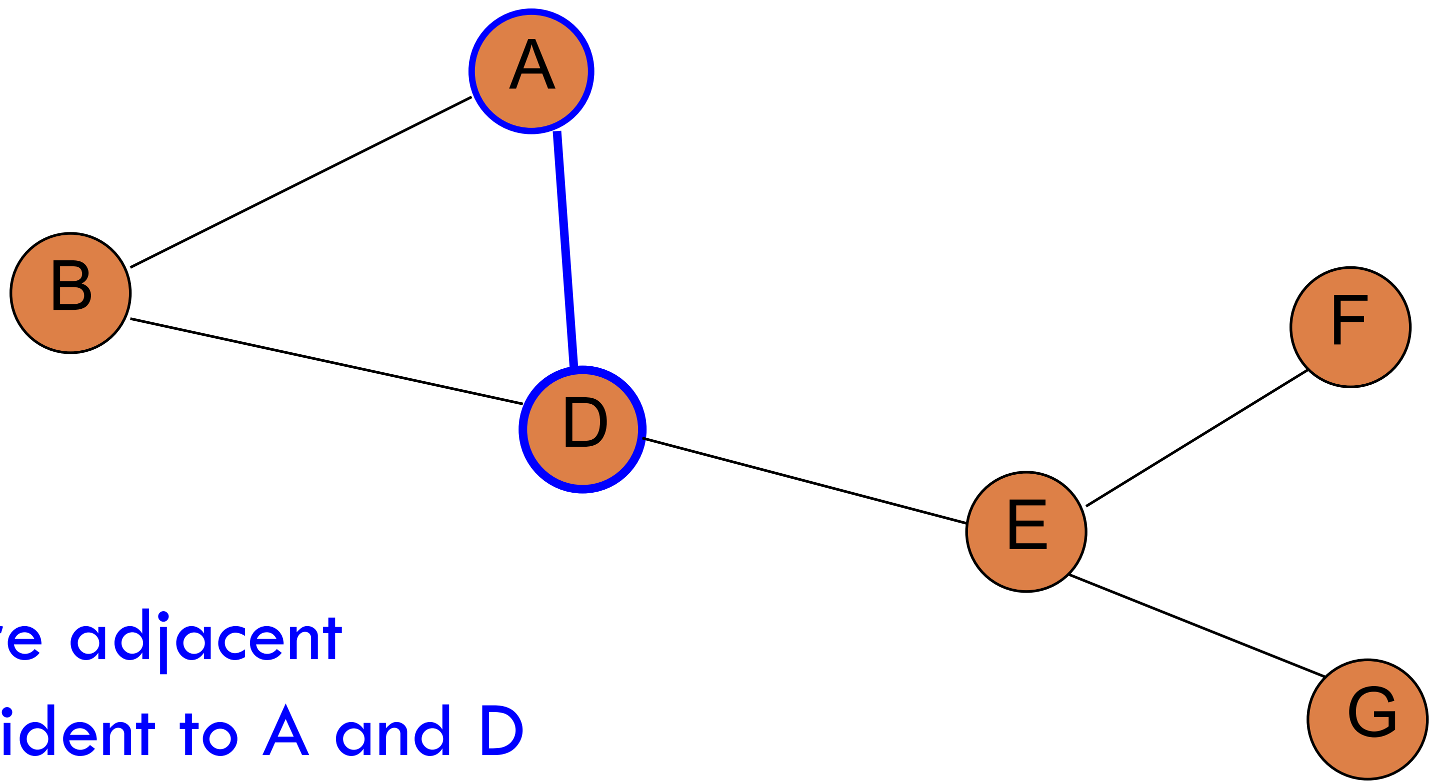
Terminology

When an edge connects two vertices, we say that the vertices are adjacent and that the edge is incident to both vertices



Terminology

When an edge connects two vertices, we say that the vertices are adjacent and that the edge is incident to both vertices

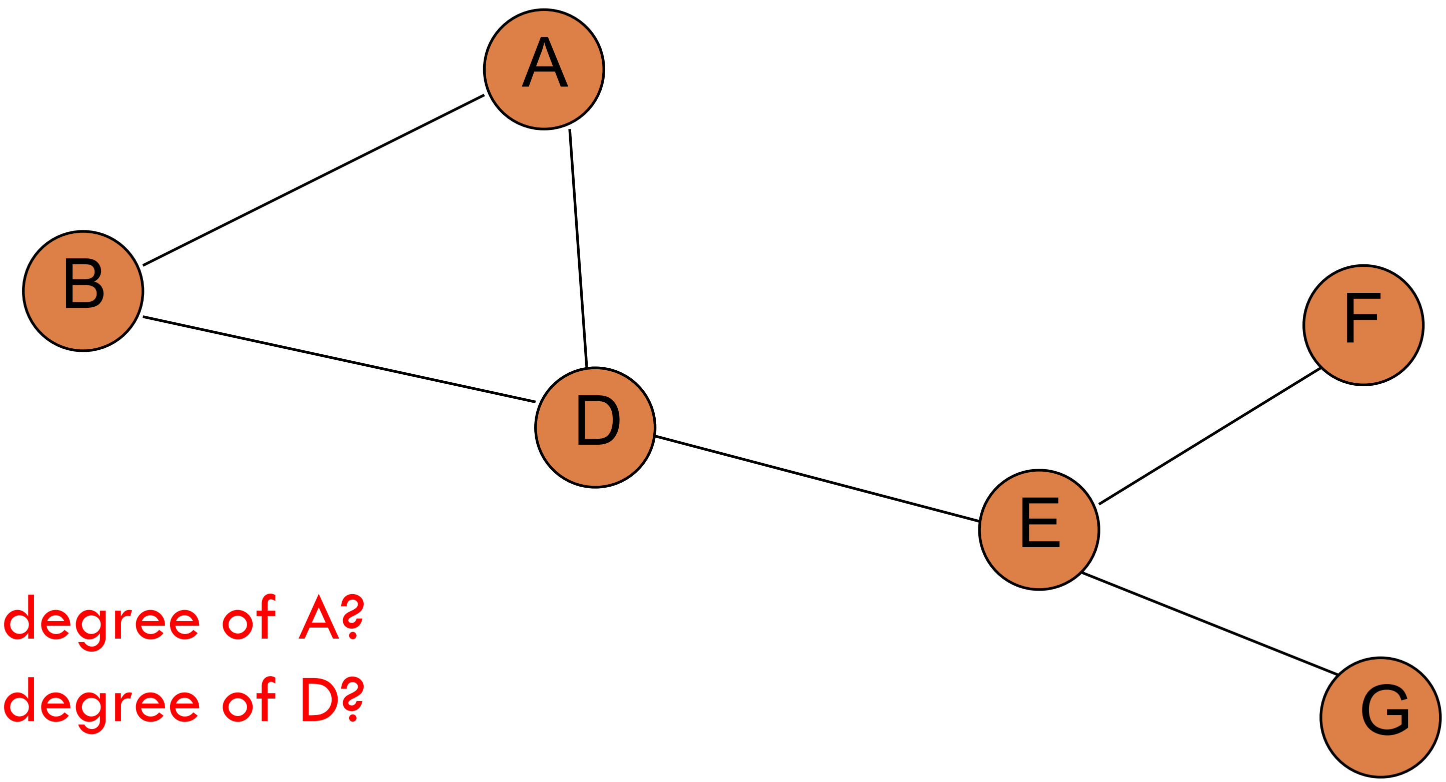


A and D are adjacent

(A, D) is incident to A and D

Terminology

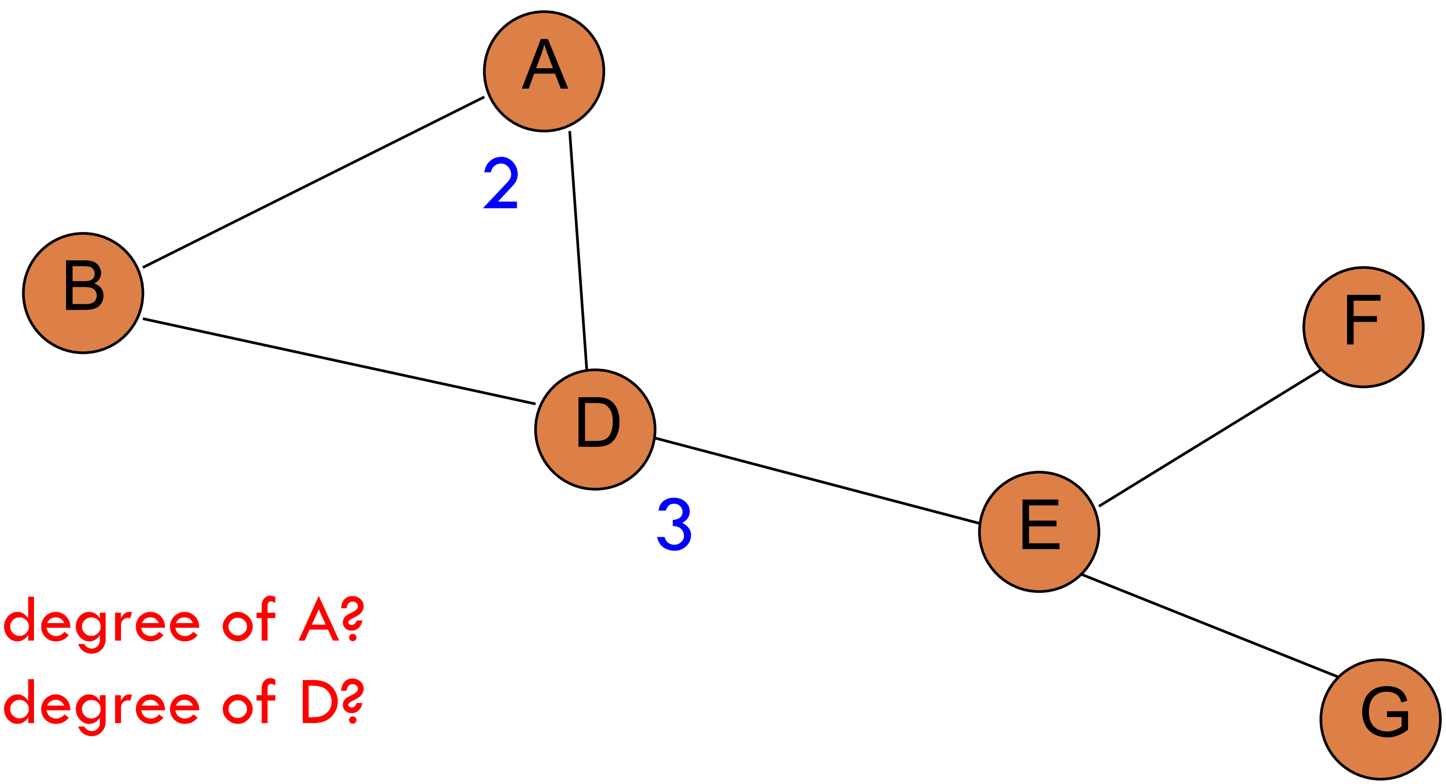
The degree of a vertex is the number of edges incident to it



What is the degree of A?
What is the degree of D?

Terminology

The degree of a vertex is the number of edges incident to it

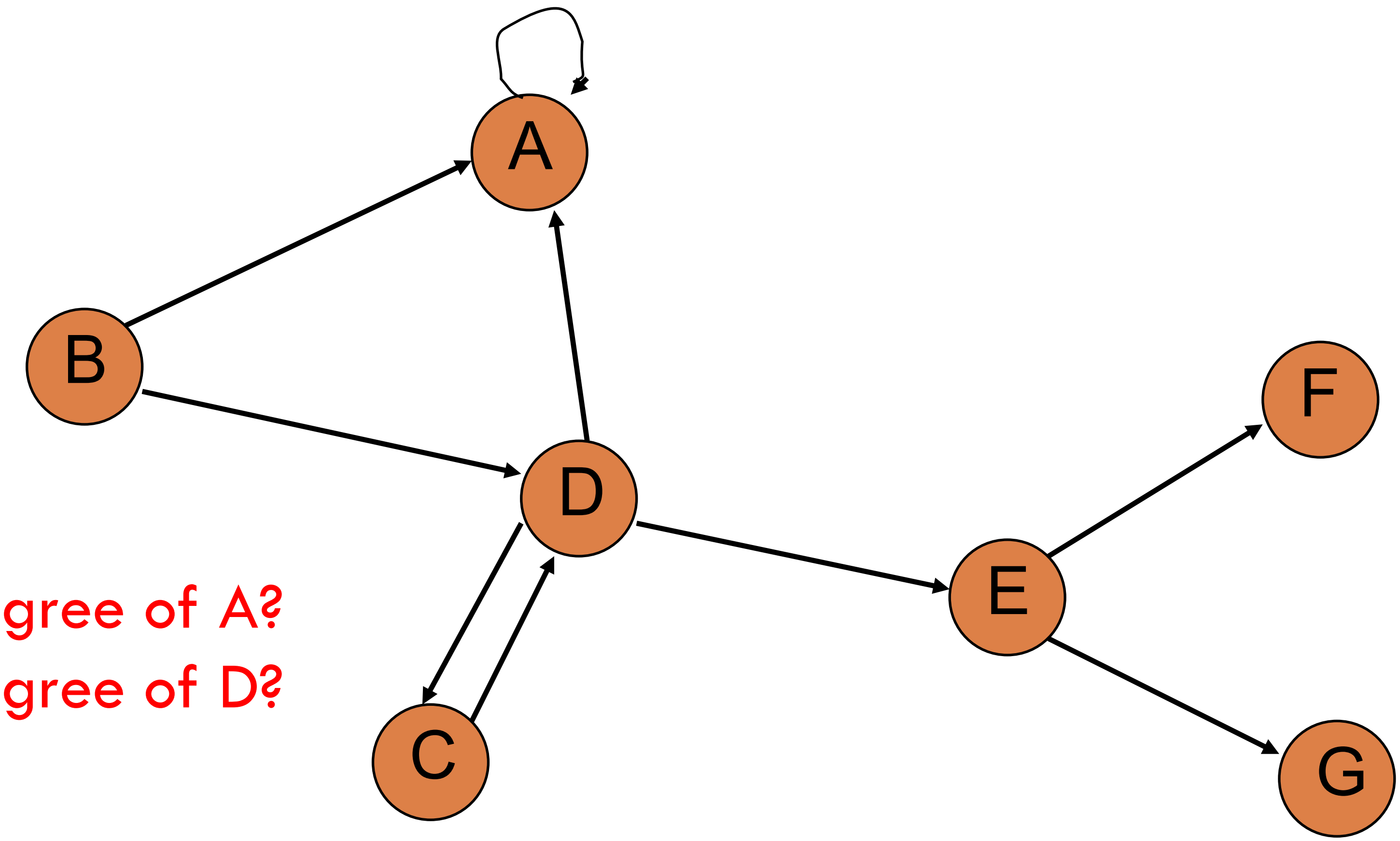


What is the degree of A?

What is the degree of D?

Terminology

The degree of a vertex is the number of edges incident to it

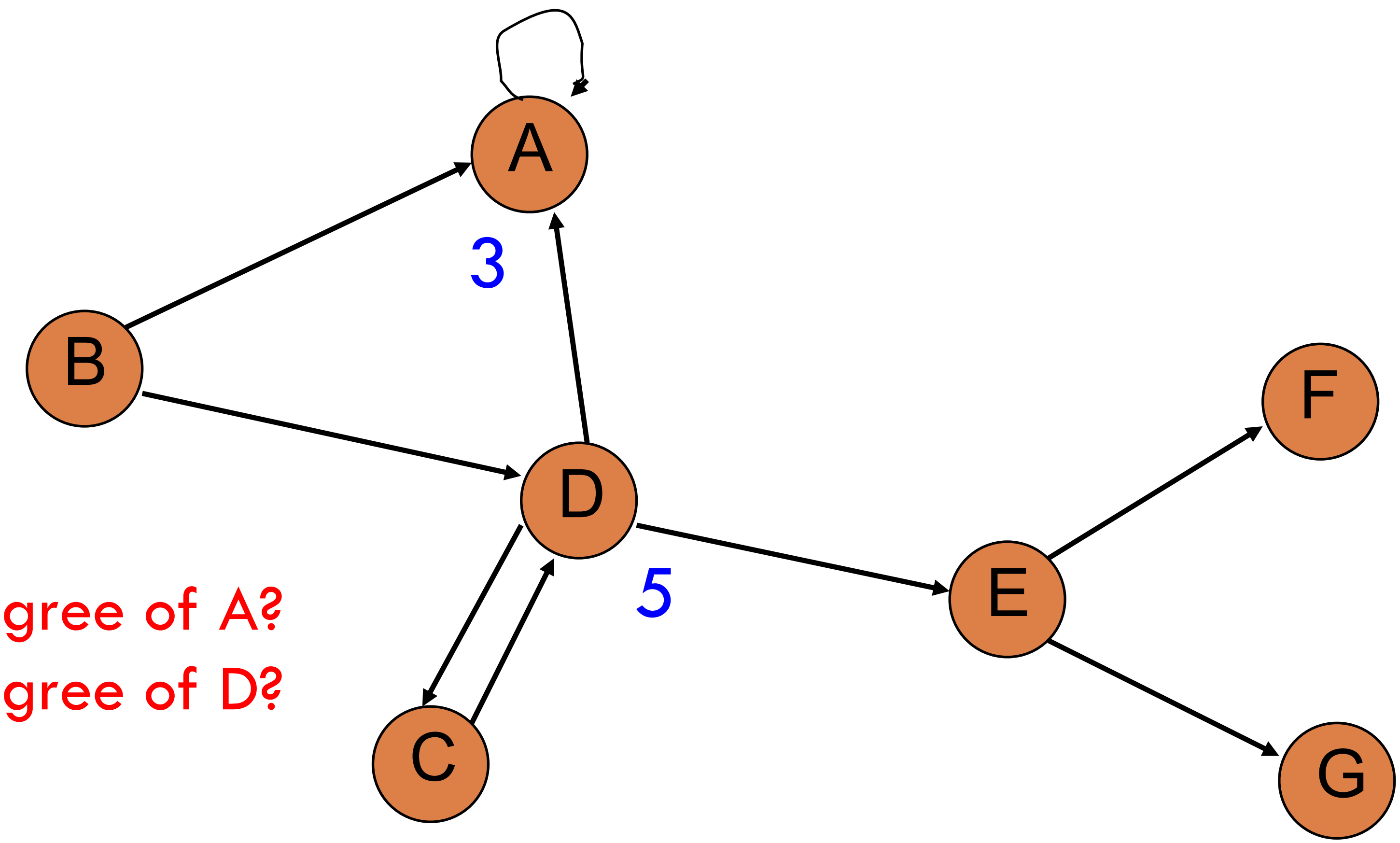


What is the degree of A?

What is the degree of D?

Terminology

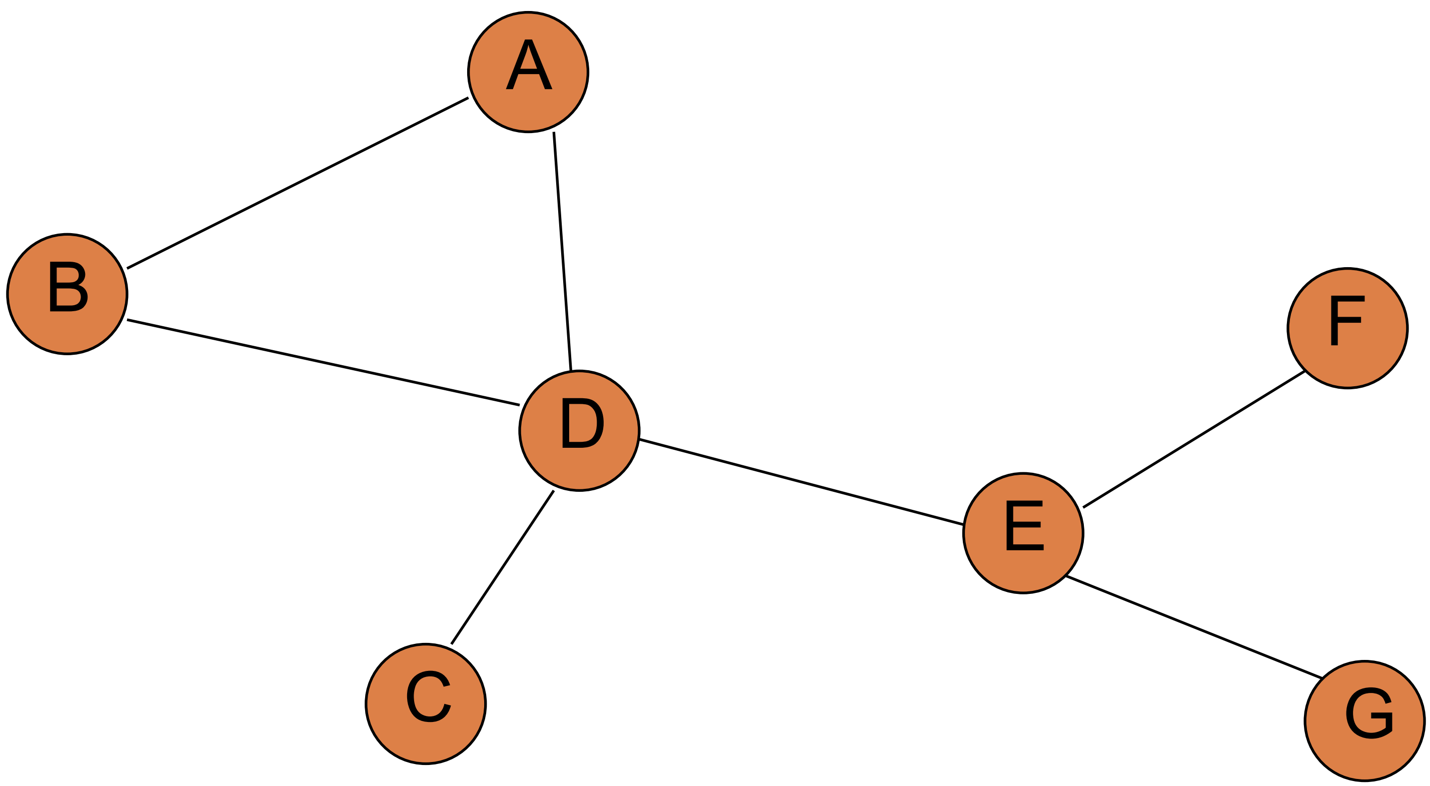
The degree of a vertex is the number of edges incident to it



What is the degree of A?
What is the degree of D?

Terminology

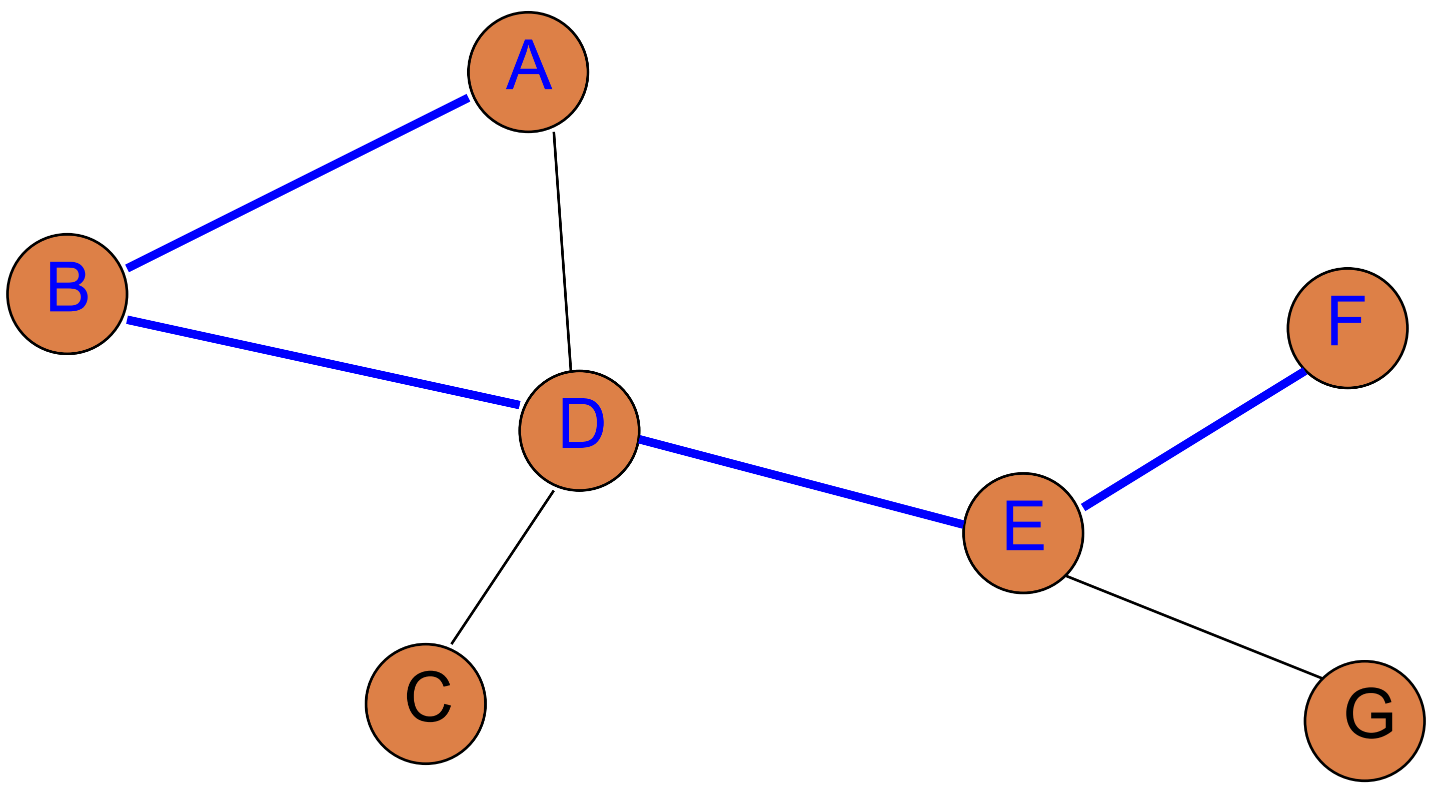
Path – A path is a sequence of vertices $p_1, p_2, \ldots p_k$ where there exists an edge $(p_i, p_{i+1}) \in E$ and no edge is repeated



Terminology

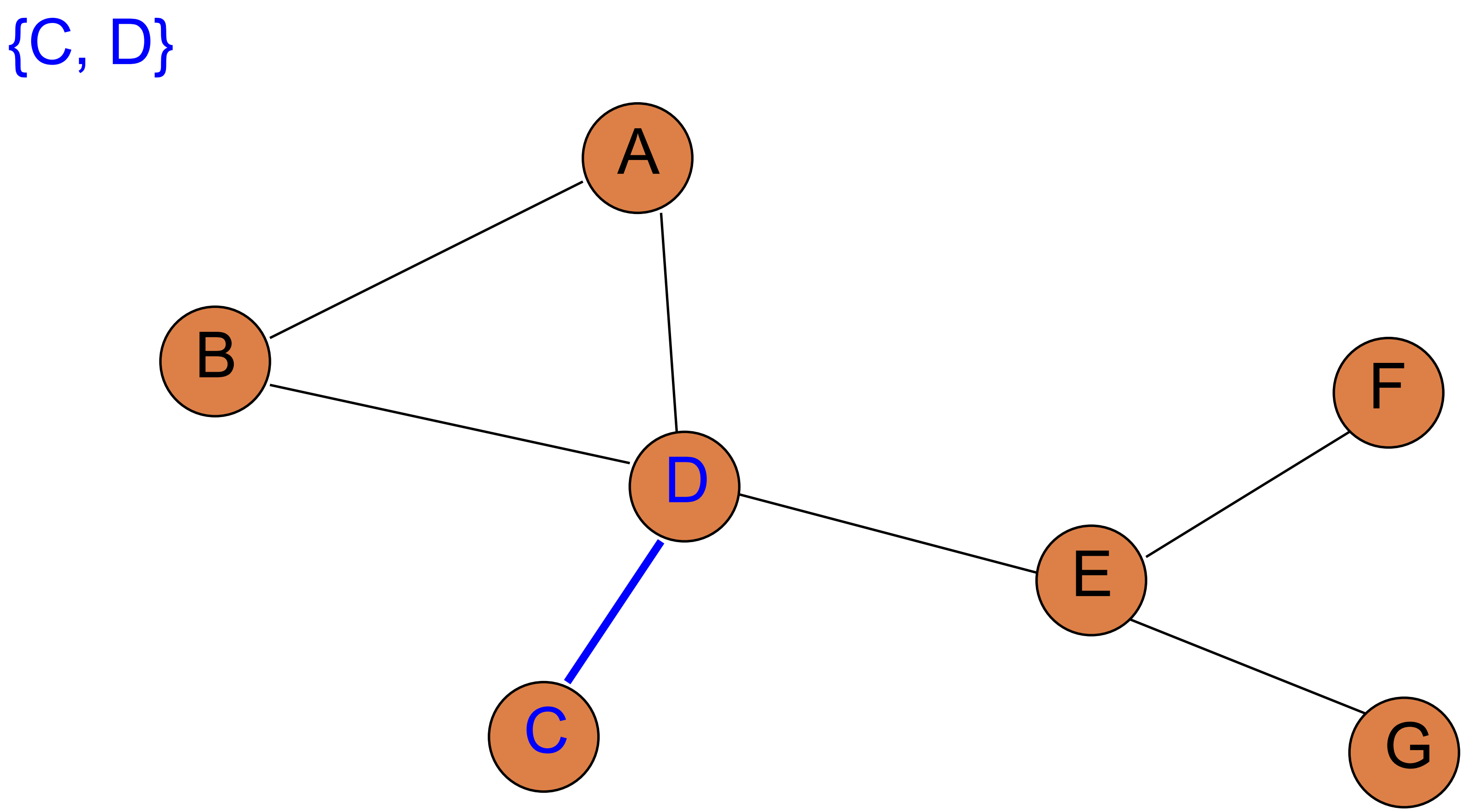
Path – A path is a sequence of vertices $p_1, p_2, \ldots p_k$ where there exists an edge $(p_i, p_{i+1}) \in E$ and no edge is repeated

{A, B, D, E, F}



Terminology

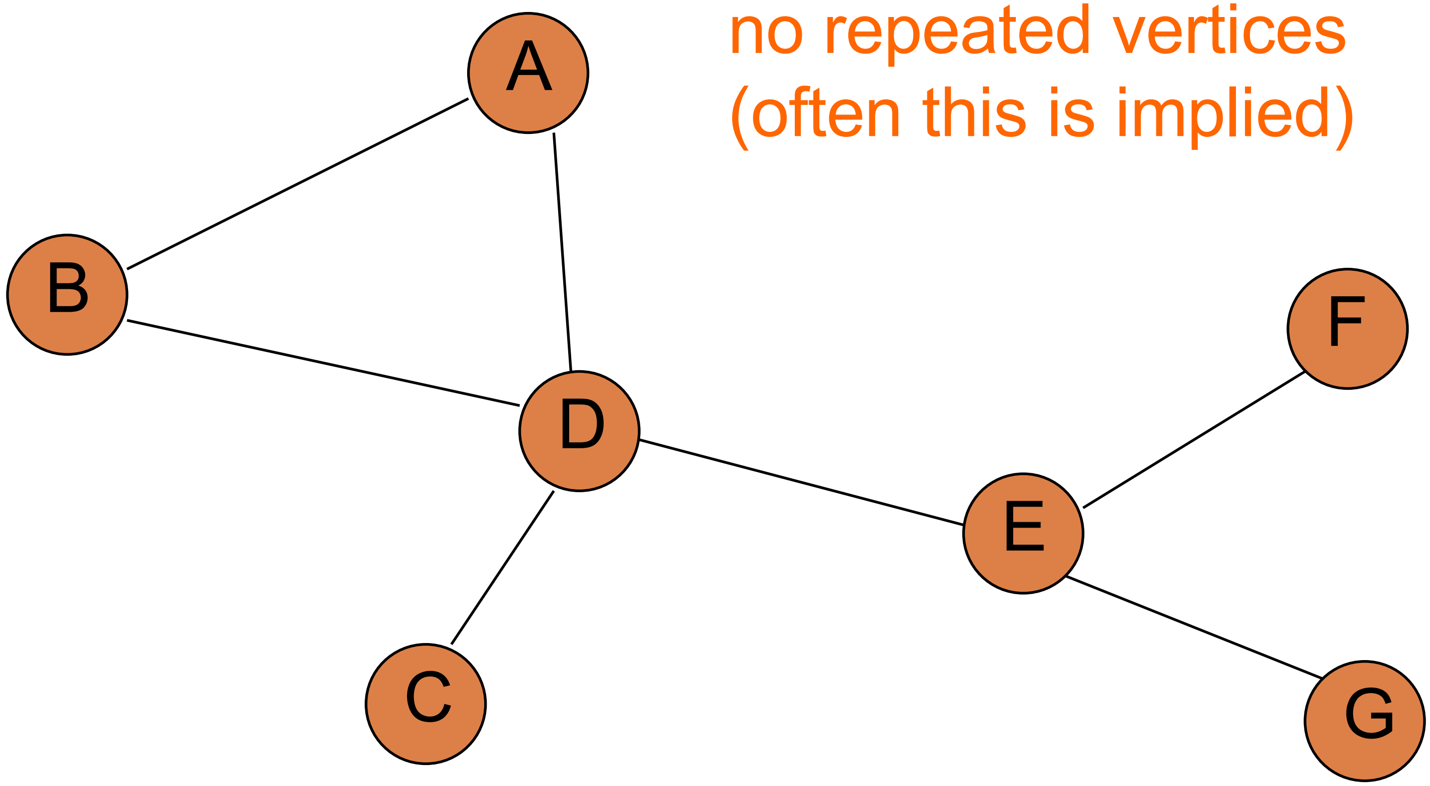
Path – A path is a sequence of vertices $p_1, p_2, \ldots p_k$ where there exists an edge $(p_i, p_{i+1}) \in E$ and no edge is repeated



Terminology

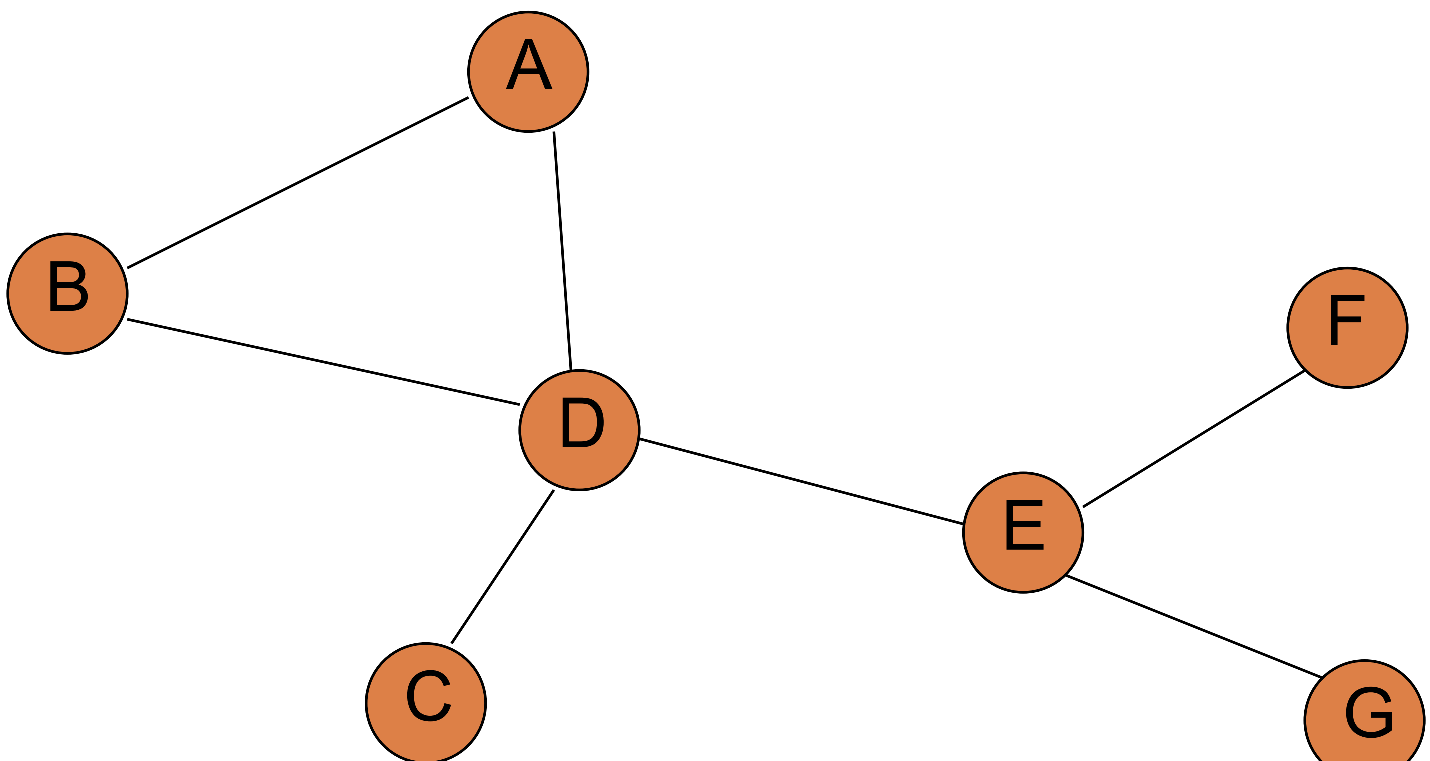
Path – A path is a sequence of vertices $p_1, p_2, \ldots p_k$ where there exists an edge $(p_i, p_{i+1}) \in E$ and no edge is repeated

A *simple* path contains no repeated vertices (often this is implied)



Terminology

Cycle – A path where the first and last node are the same

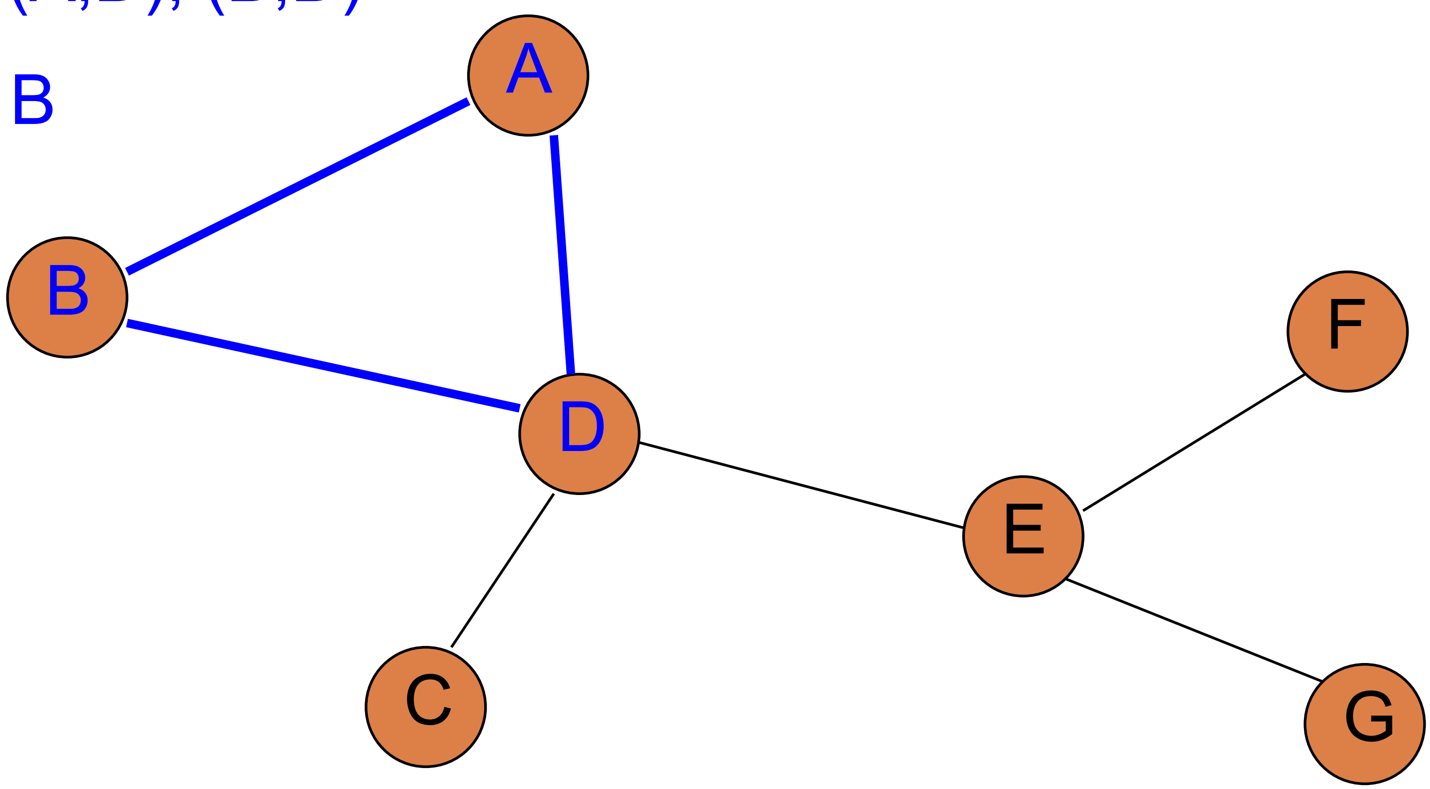


Terminology

Cycle – A path where the first and last node are the same

Edges: (A,B), (A,D), (B,D)

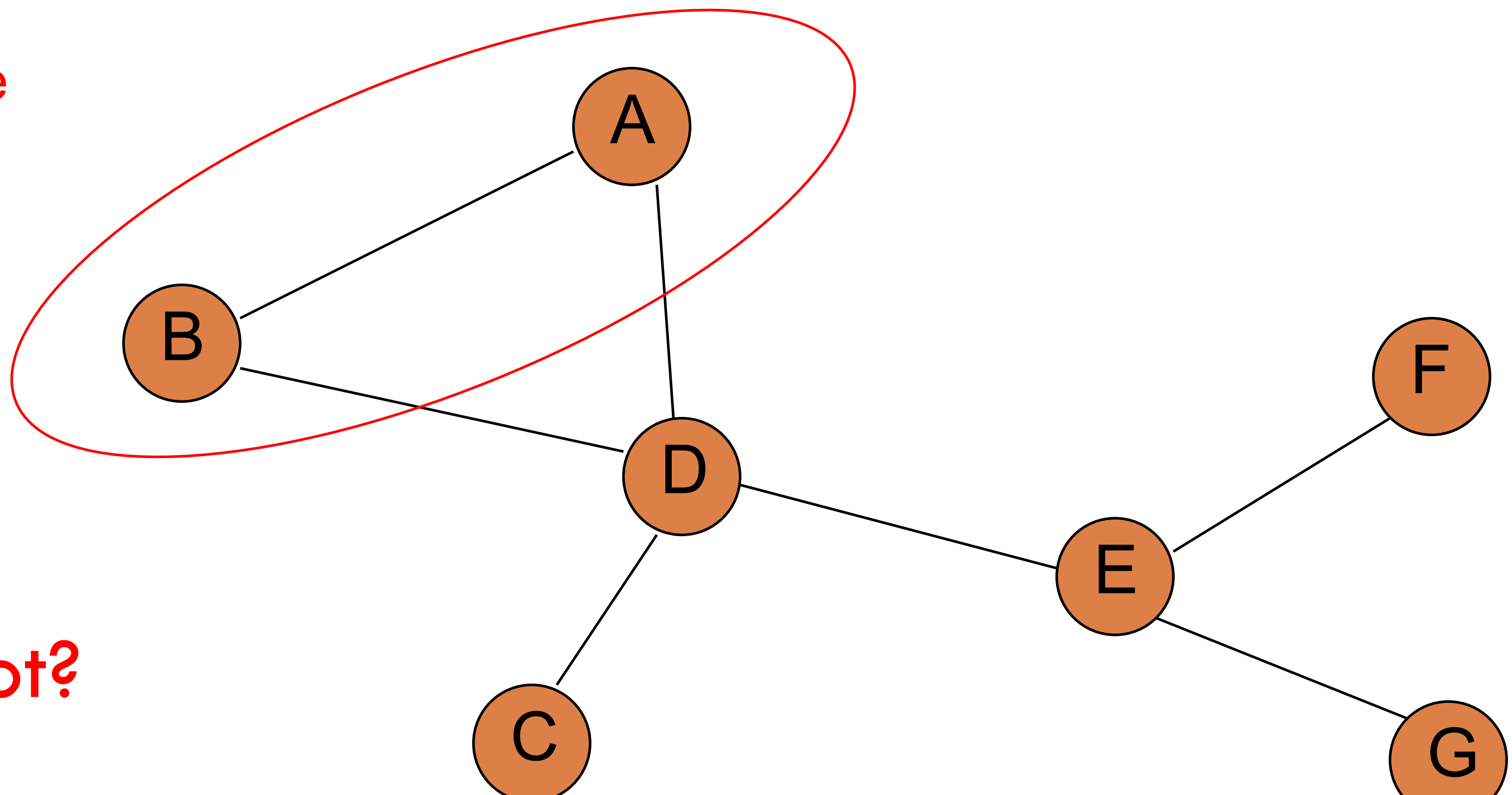
Path: B, A, D, B



Terminology

Cycle – A path where the first and last node are the same

not a cycle

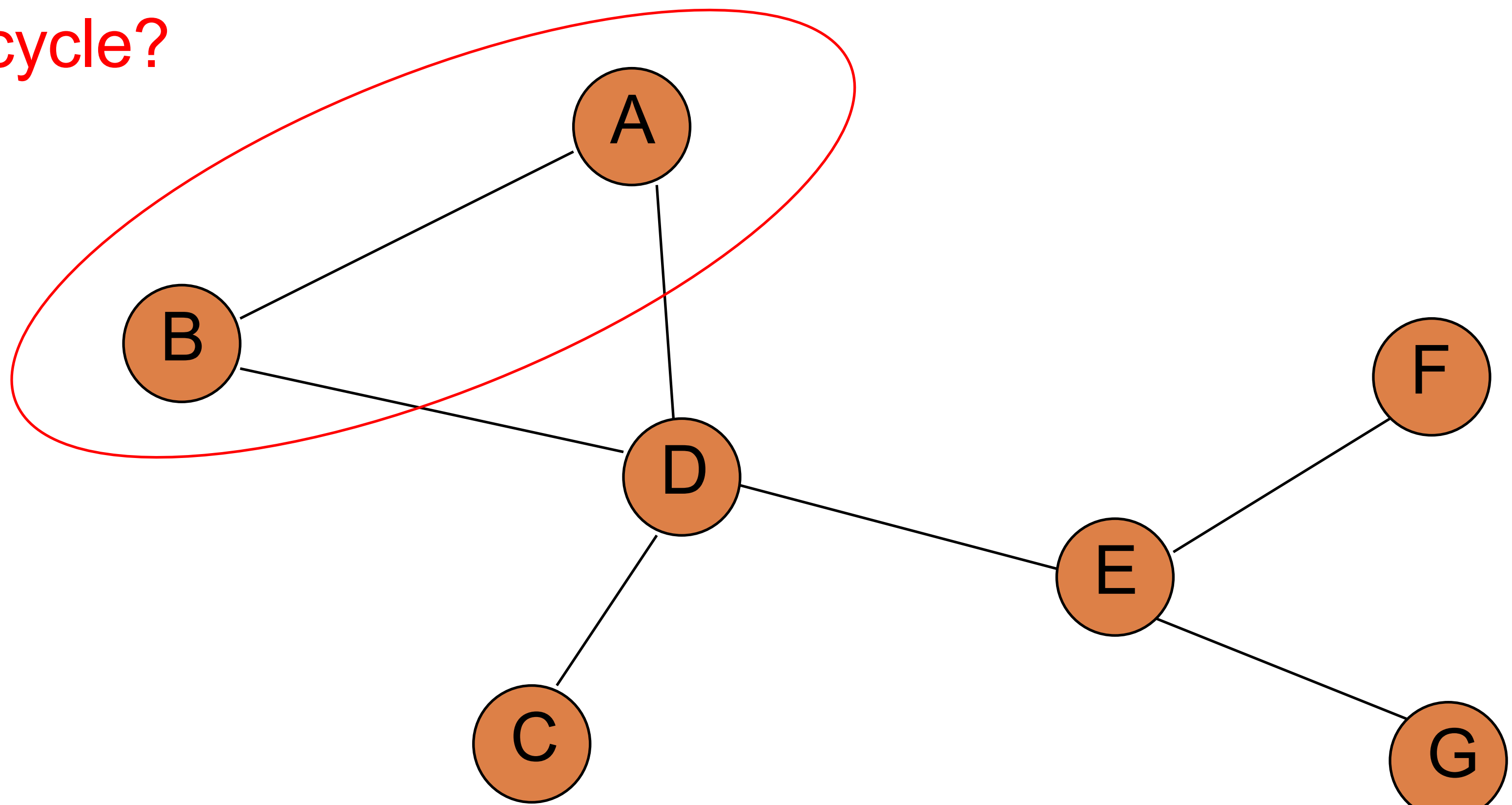


Why not?

Terminology

Cycle – A path where the first and last node are the same

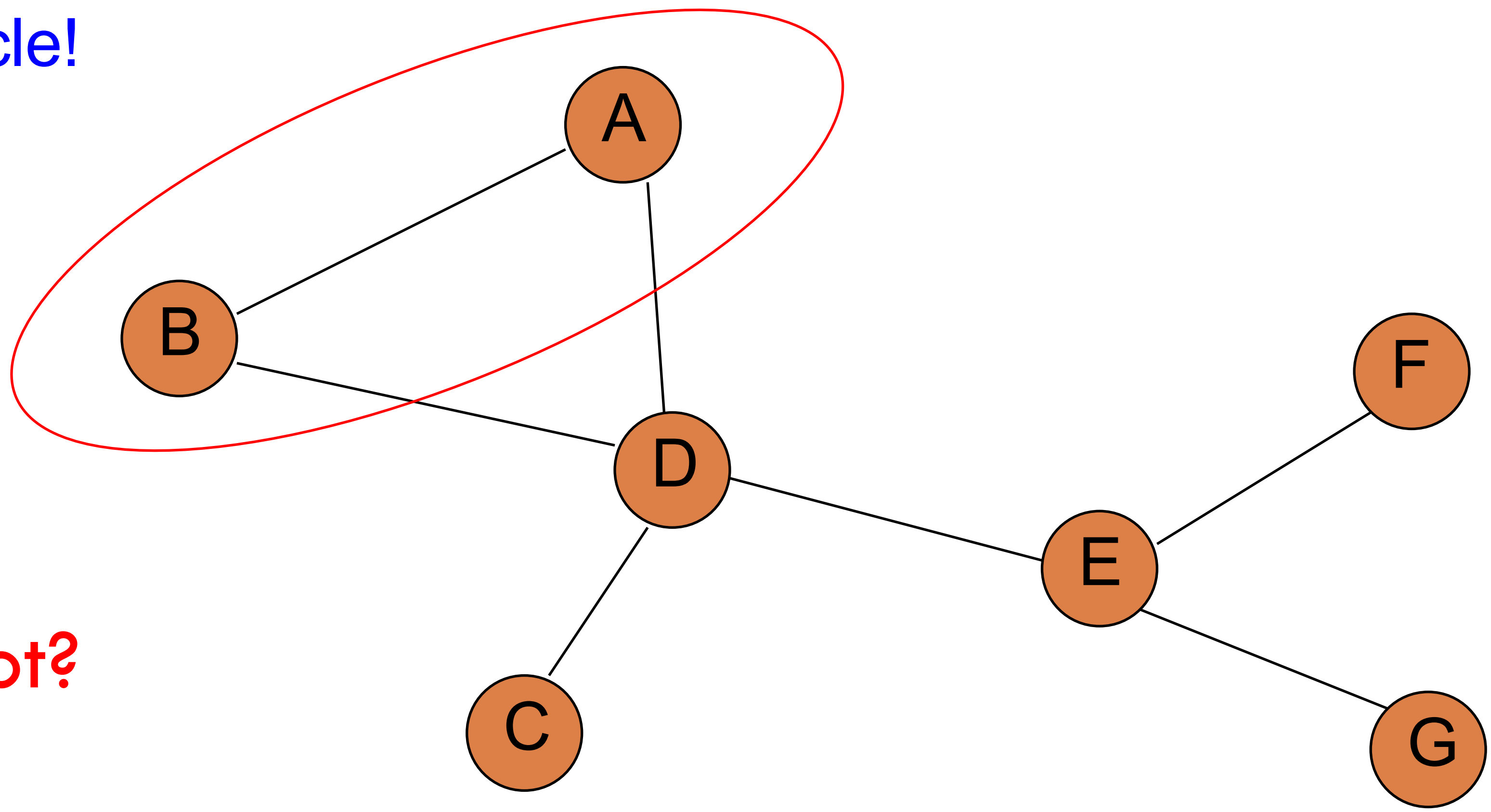
Is this a cycle?



Terminology

Cycle – A path where the first and last node are the same

Not a cycle!

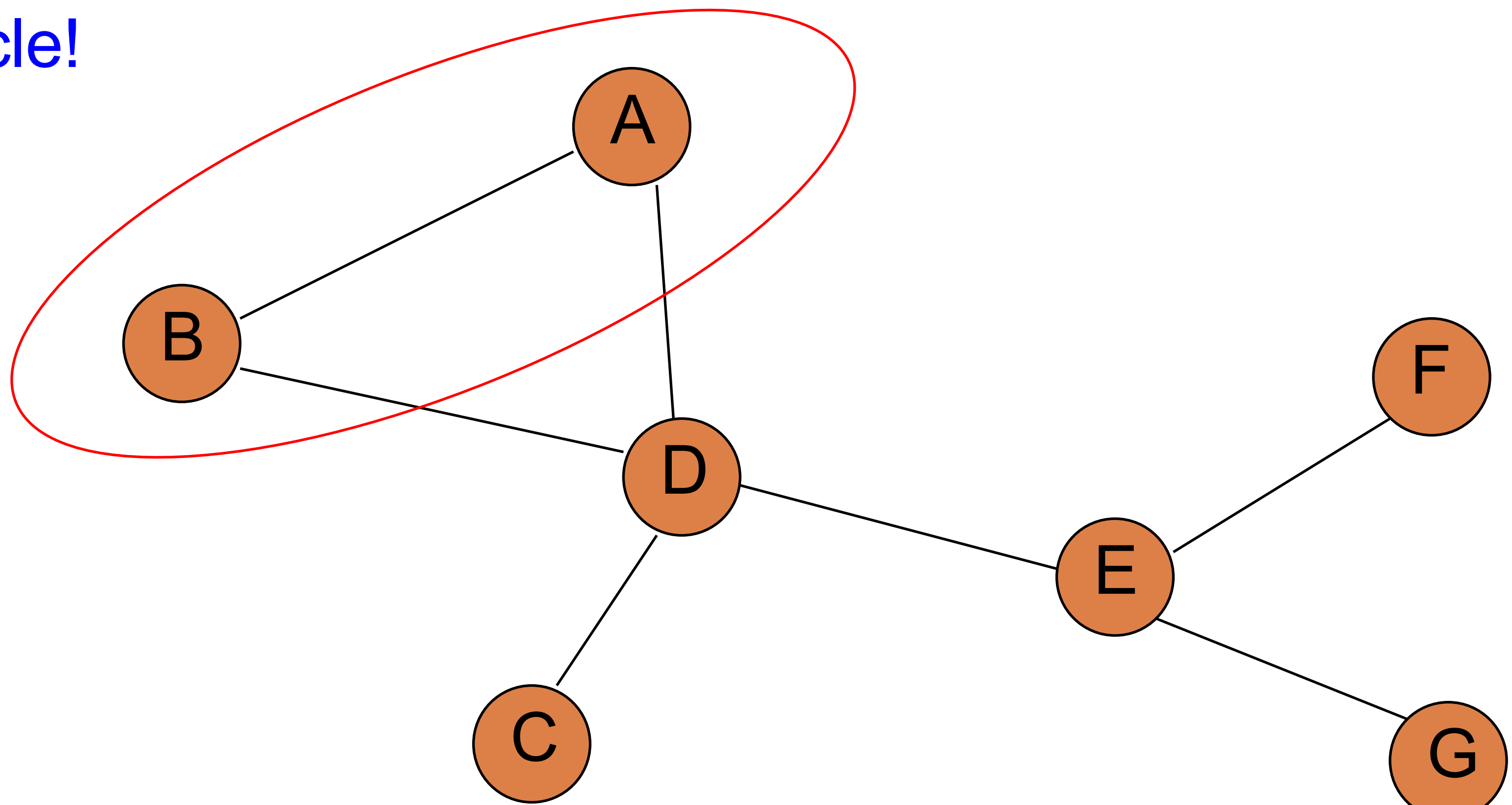


Why not?

Path – A path is a sequence of vertices $p_1, p_2, \ldots p_k$ where there exists an edge $(p_i, p_{i+1}) \in E$ and no edge is repeated

Cycle – A path where the first and last node are the same

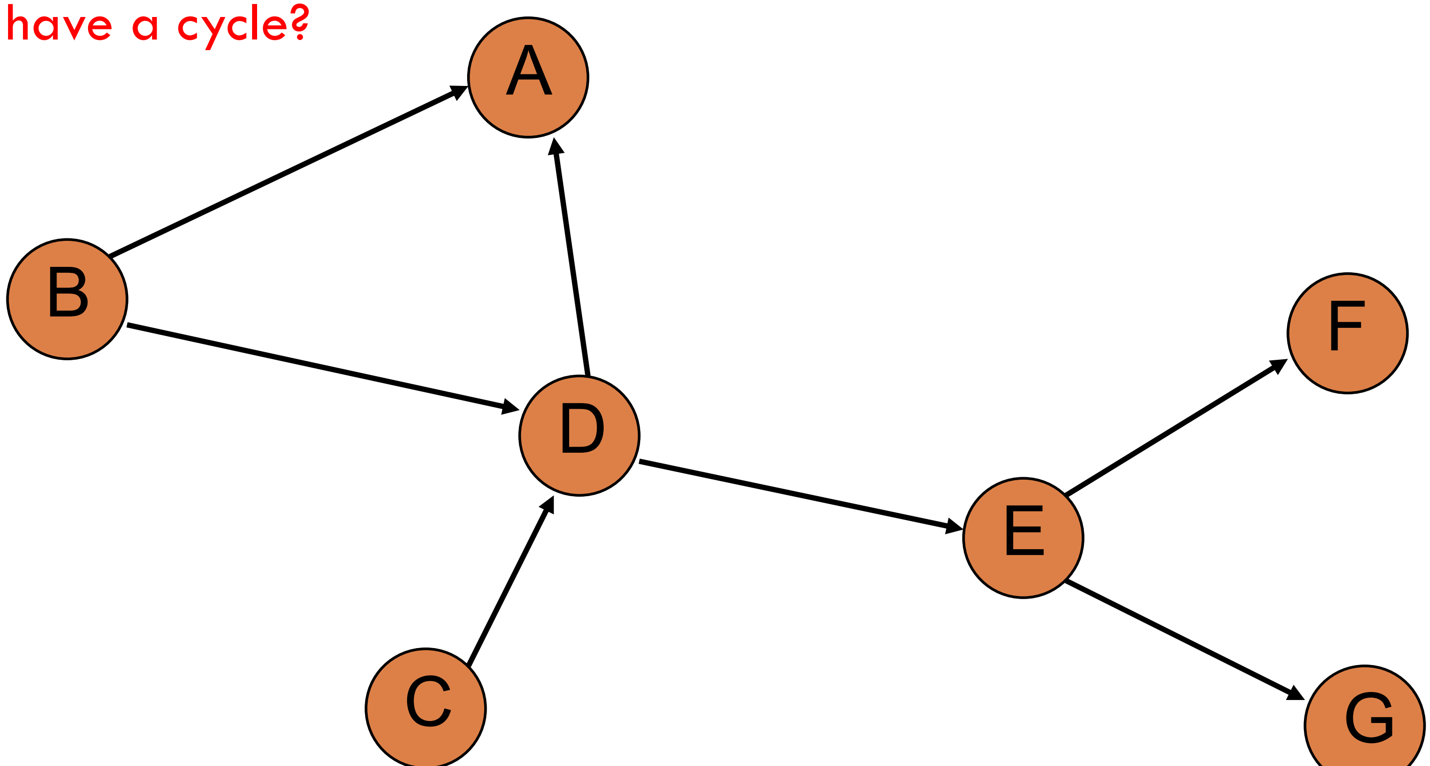
Not a cycle!



Terminology

Cycle – A path where the first and last node are the same

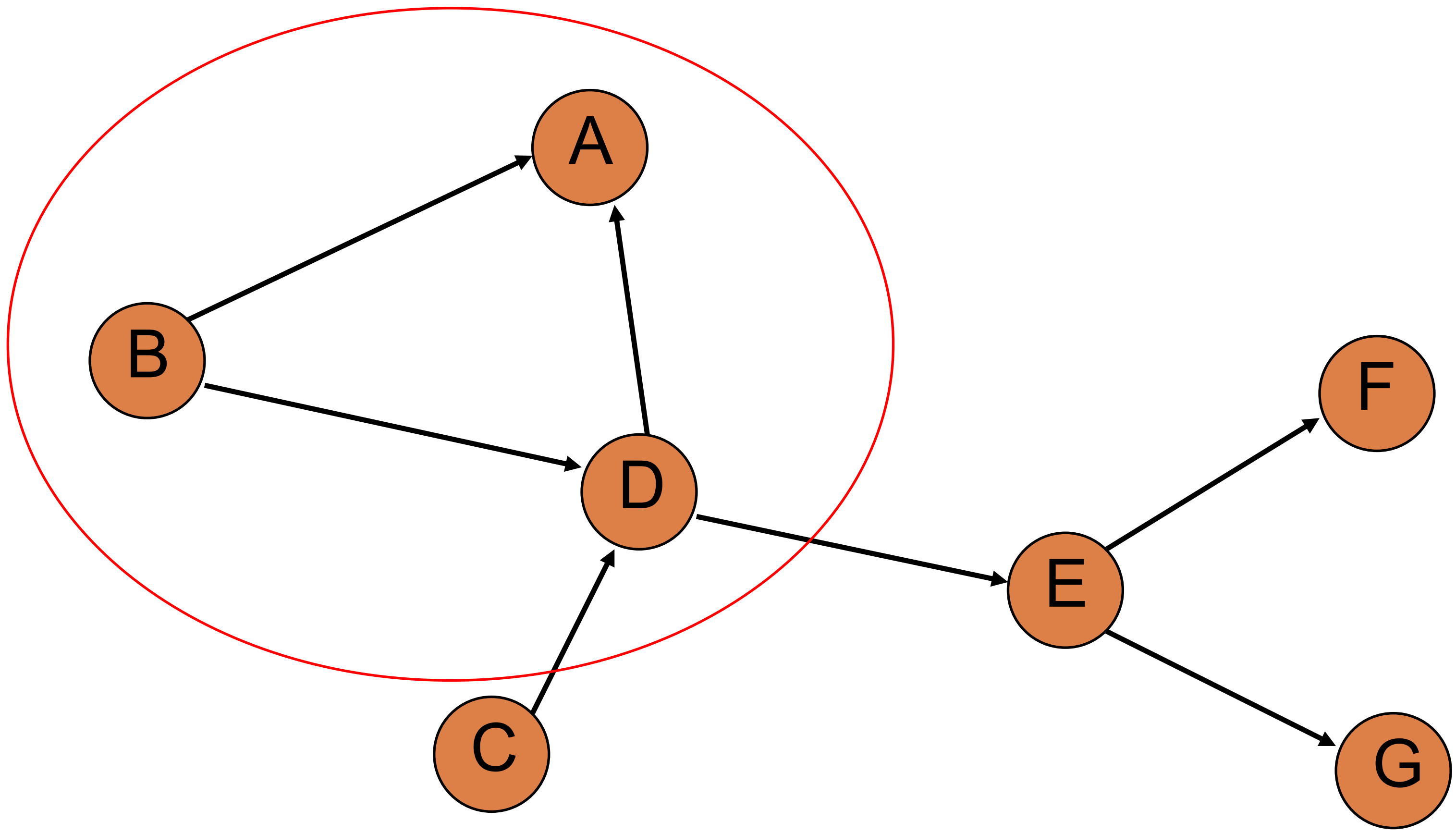
Does this graph have a cycle?



Terminology

Cycle – A path where the first and last node are the same

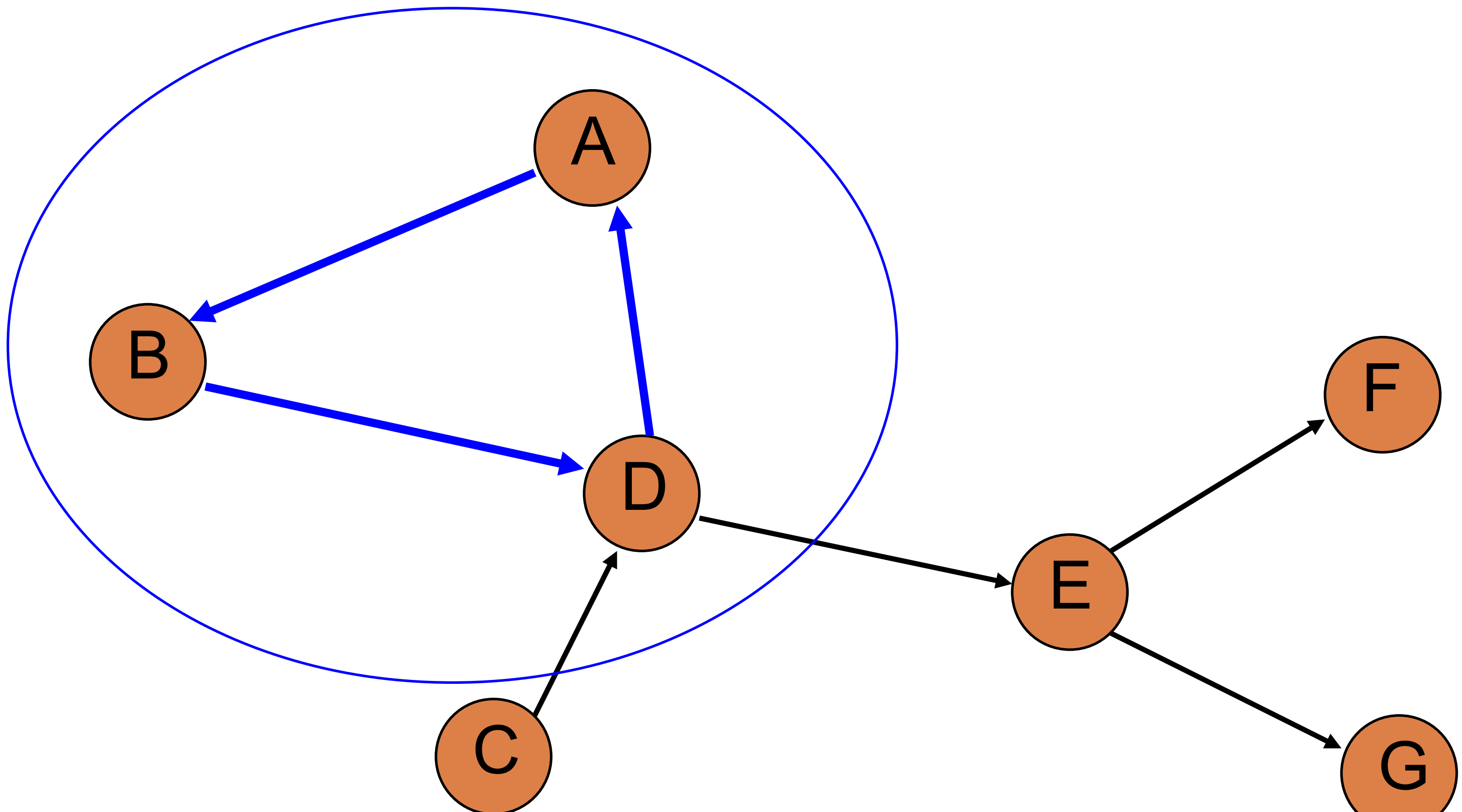
not a cycle



Terminology

Cycle – A path $p_1, p_2, \ldots p_k$ where $p_1 = p_k$

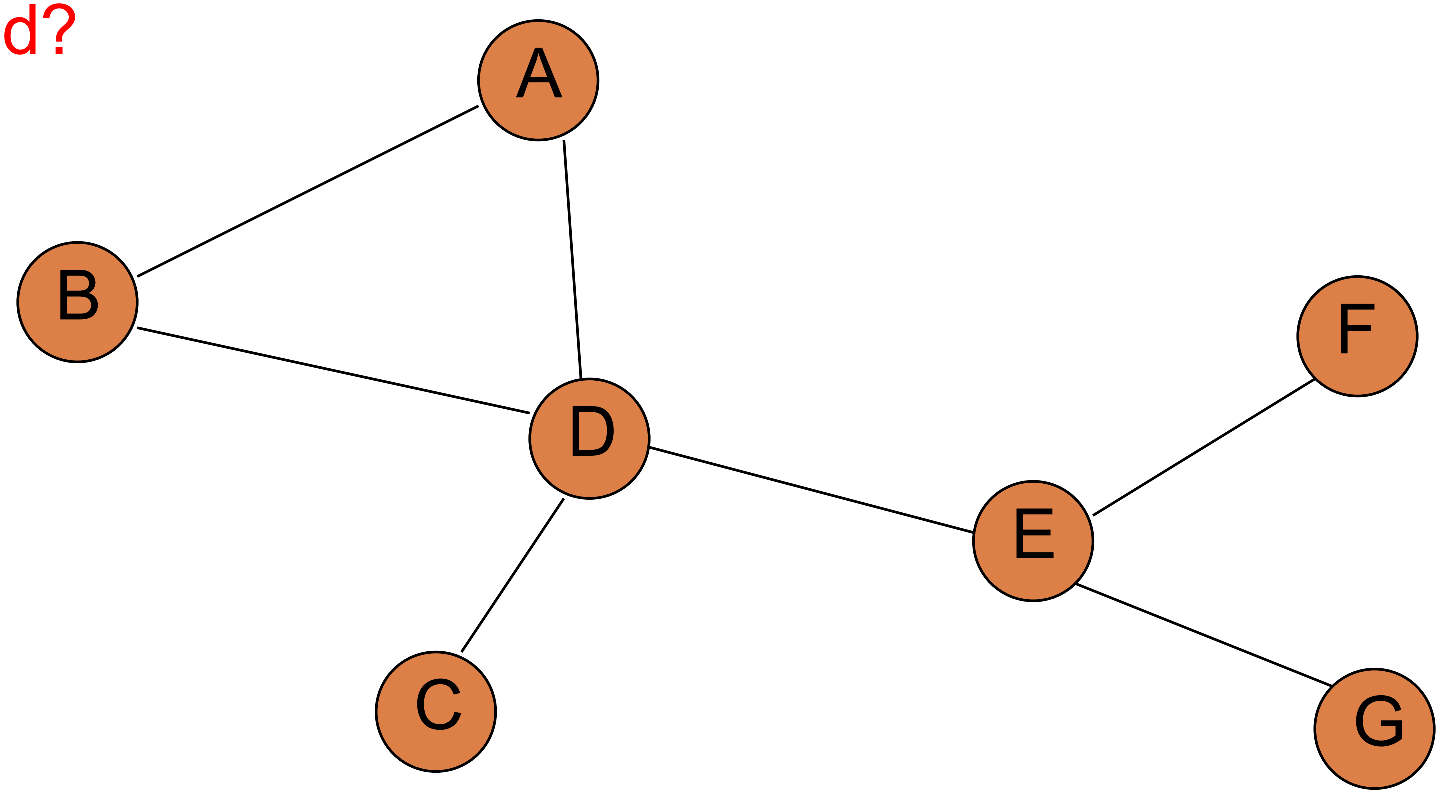
This would be a cycle



Terminology

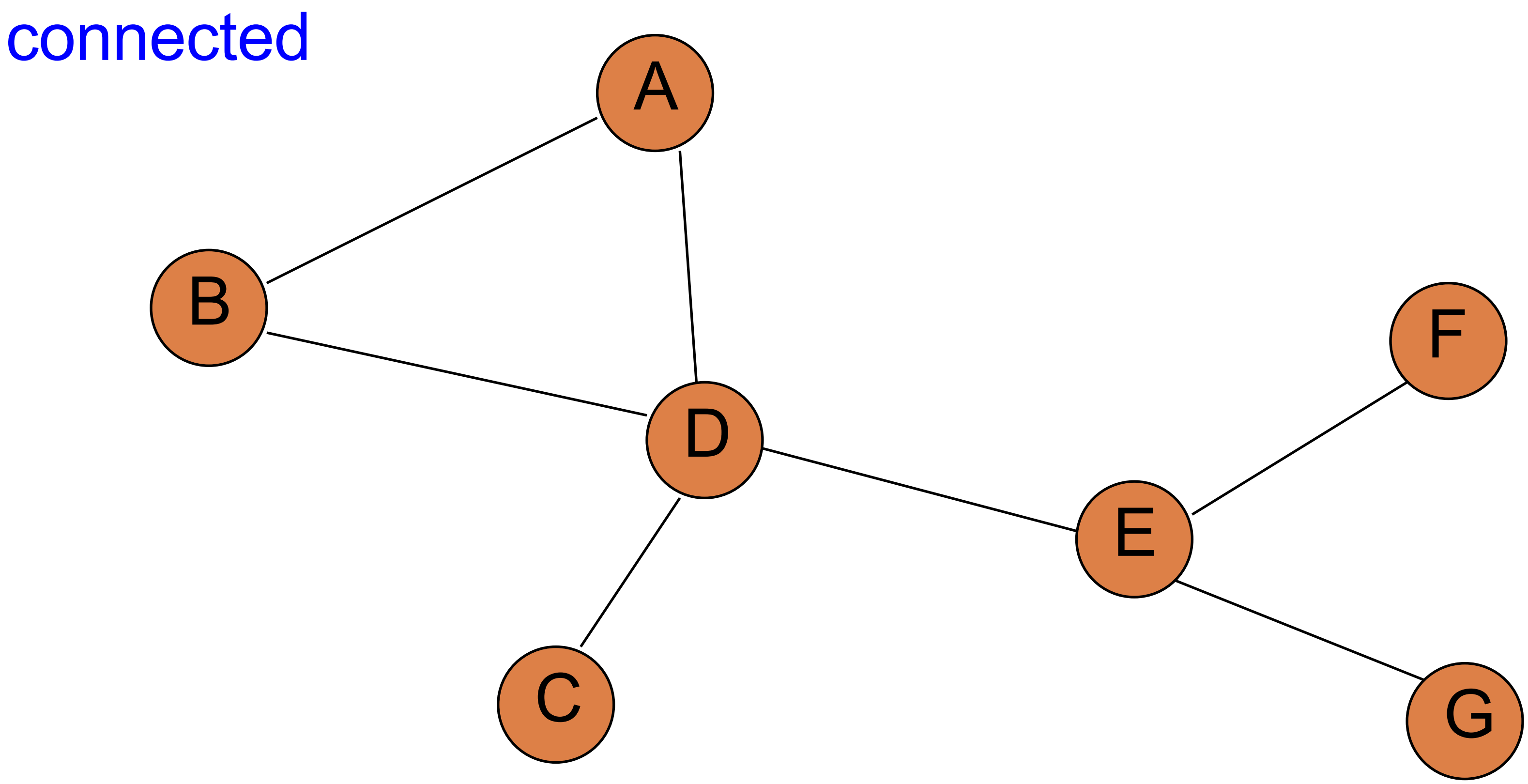
Connected – every pair of vertices is connected by a path

Is this graph connected?



Terminology

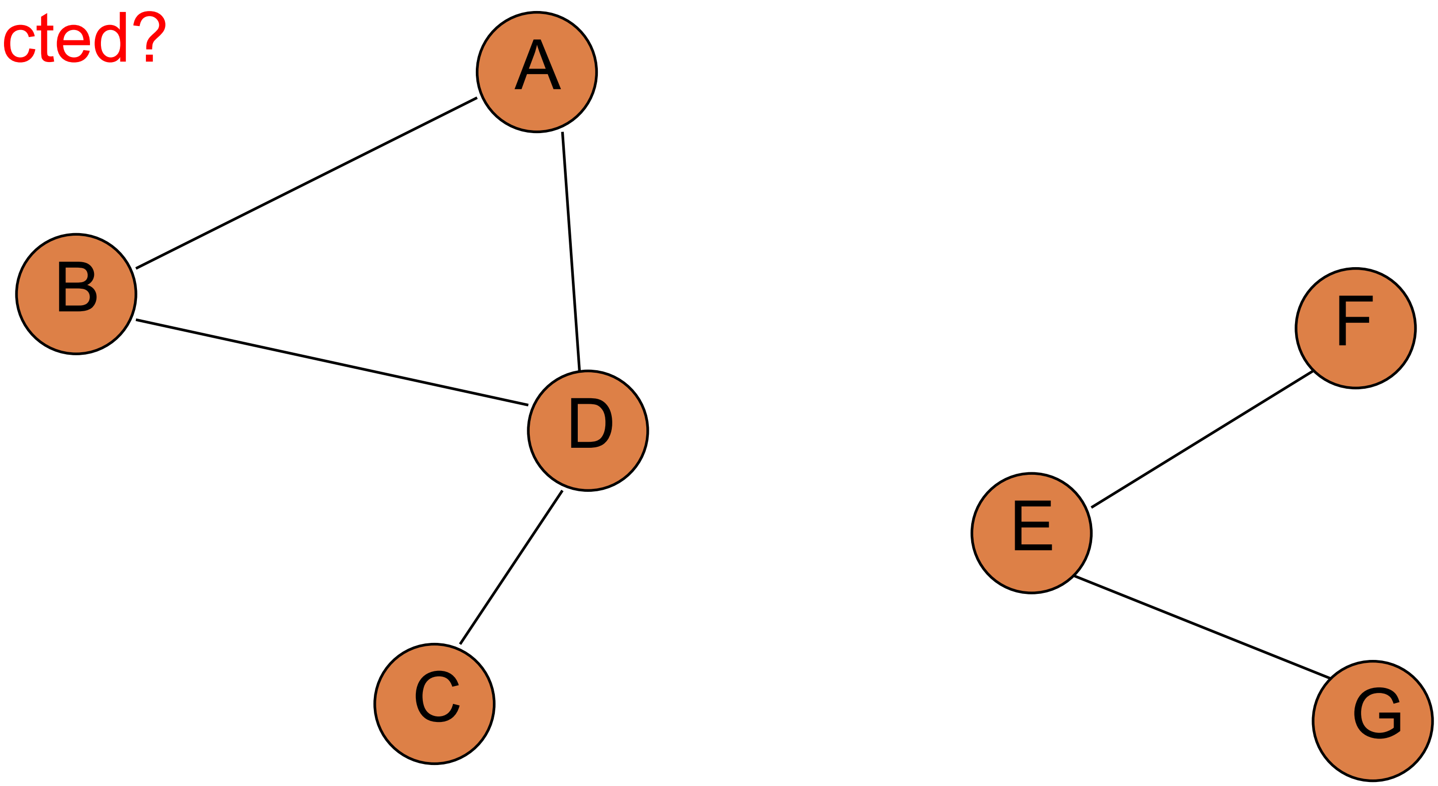
Connected – every pair of vertices is connected by a path



Terminology

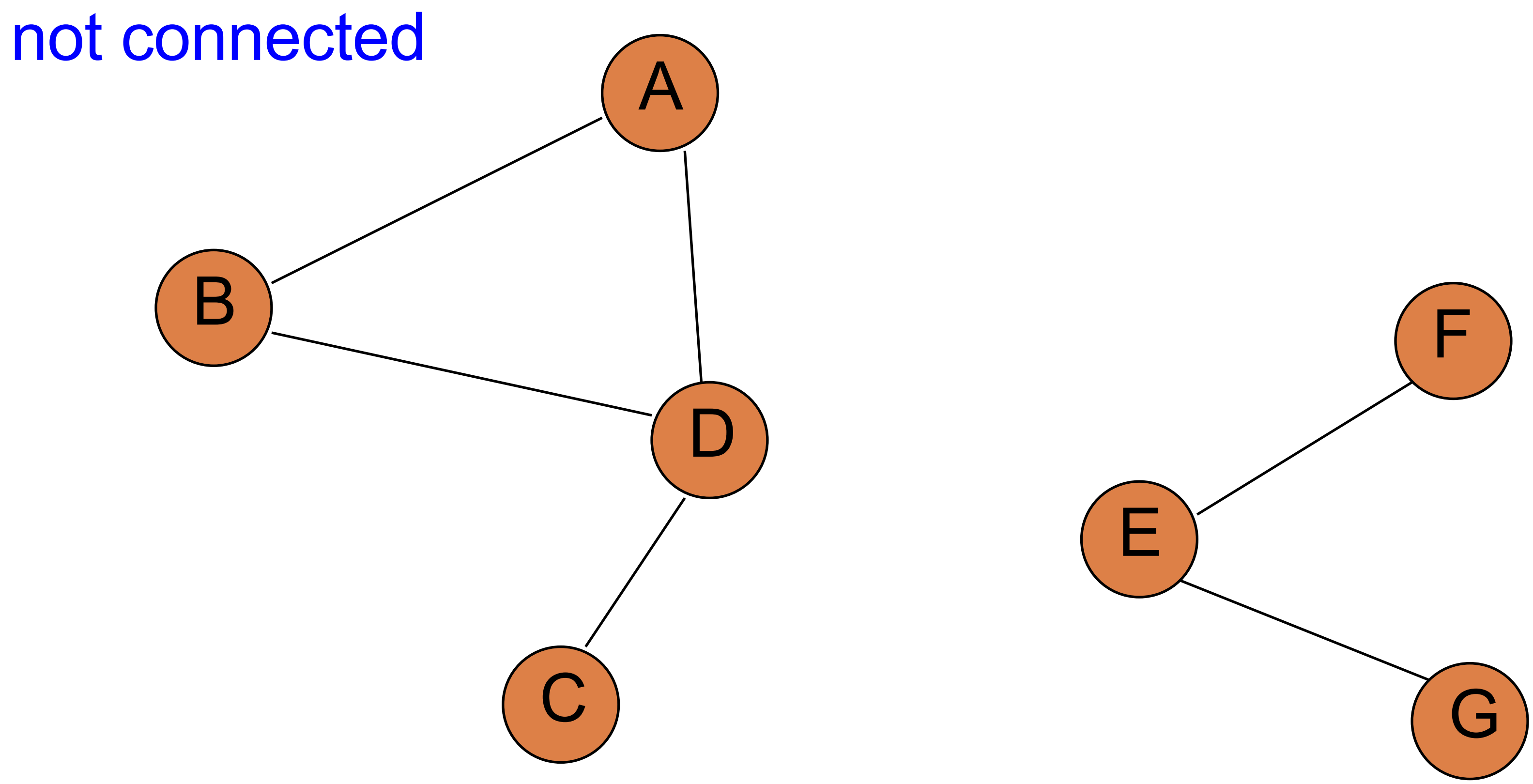
Connected – every pair of vertices is connected by a path

Is this graph connected?



Terminology

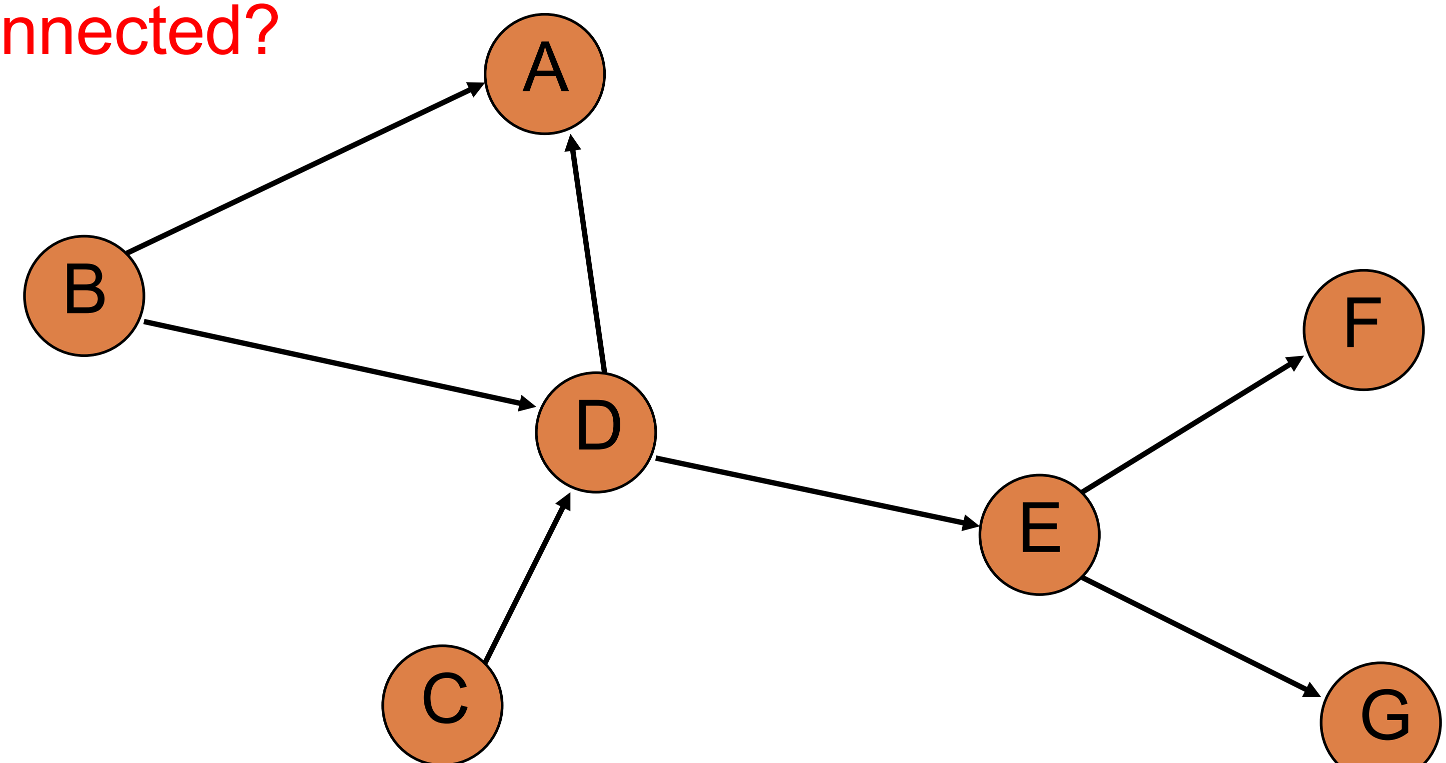
Connected – every pair of vertices is connected by a path



Terminology

Strongly connected (directed graphs) –
Every two vertices are reachable by a path

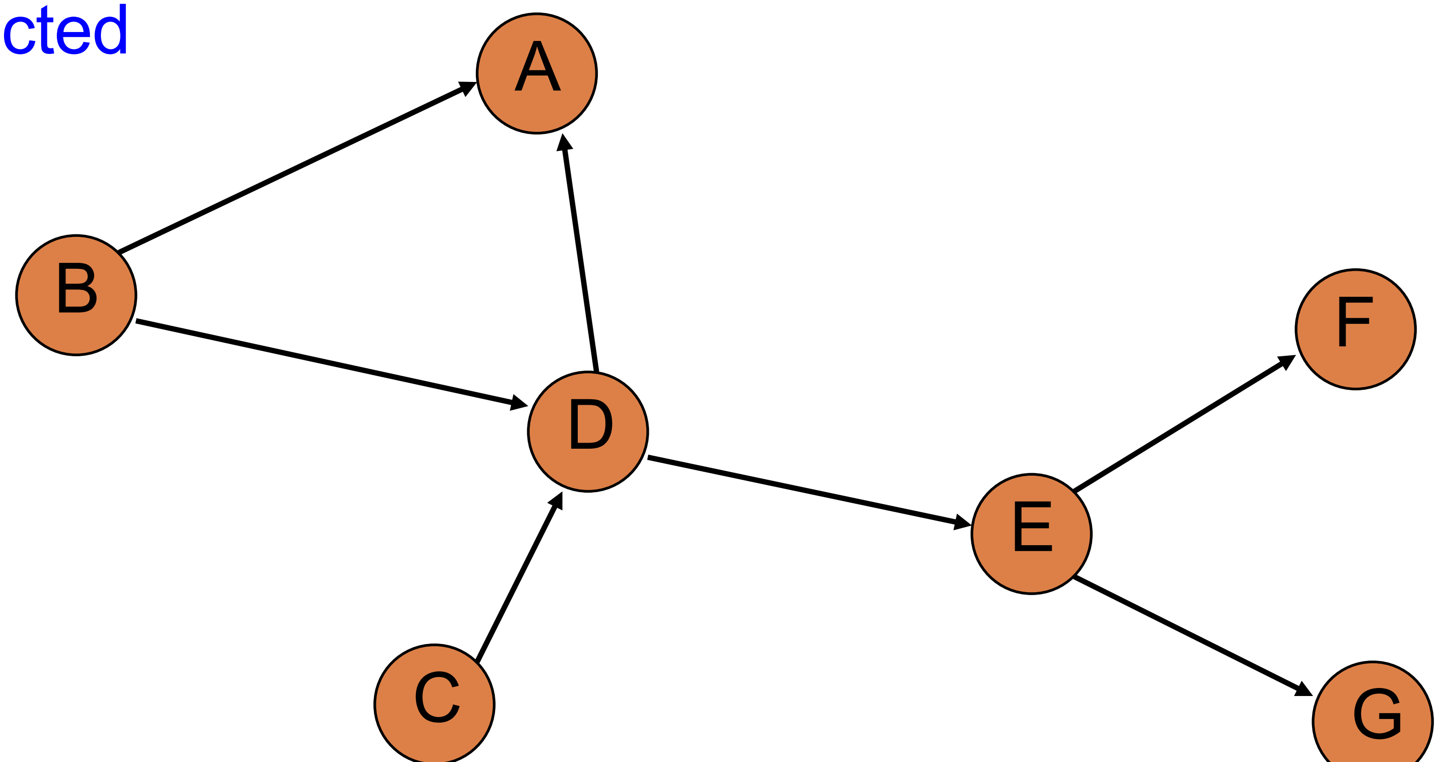
Is this graph
strongly connected?



Terminology

Strongly connected (directed graphs) –
Every two vertices are reachable by a path

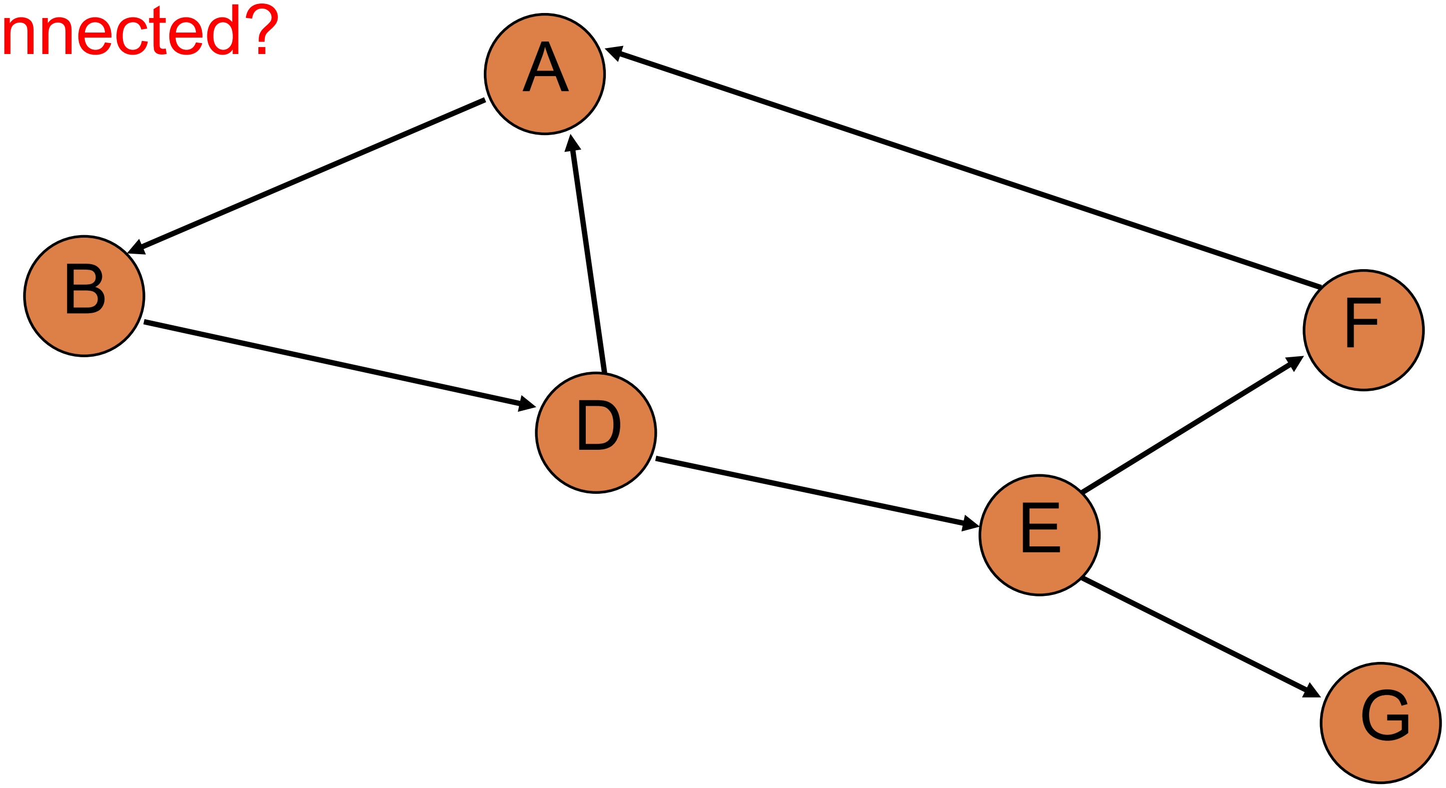
not strongly
connected



Terminology

Strongly connected (directed graphs) –
Every two vertices are reachable by a path

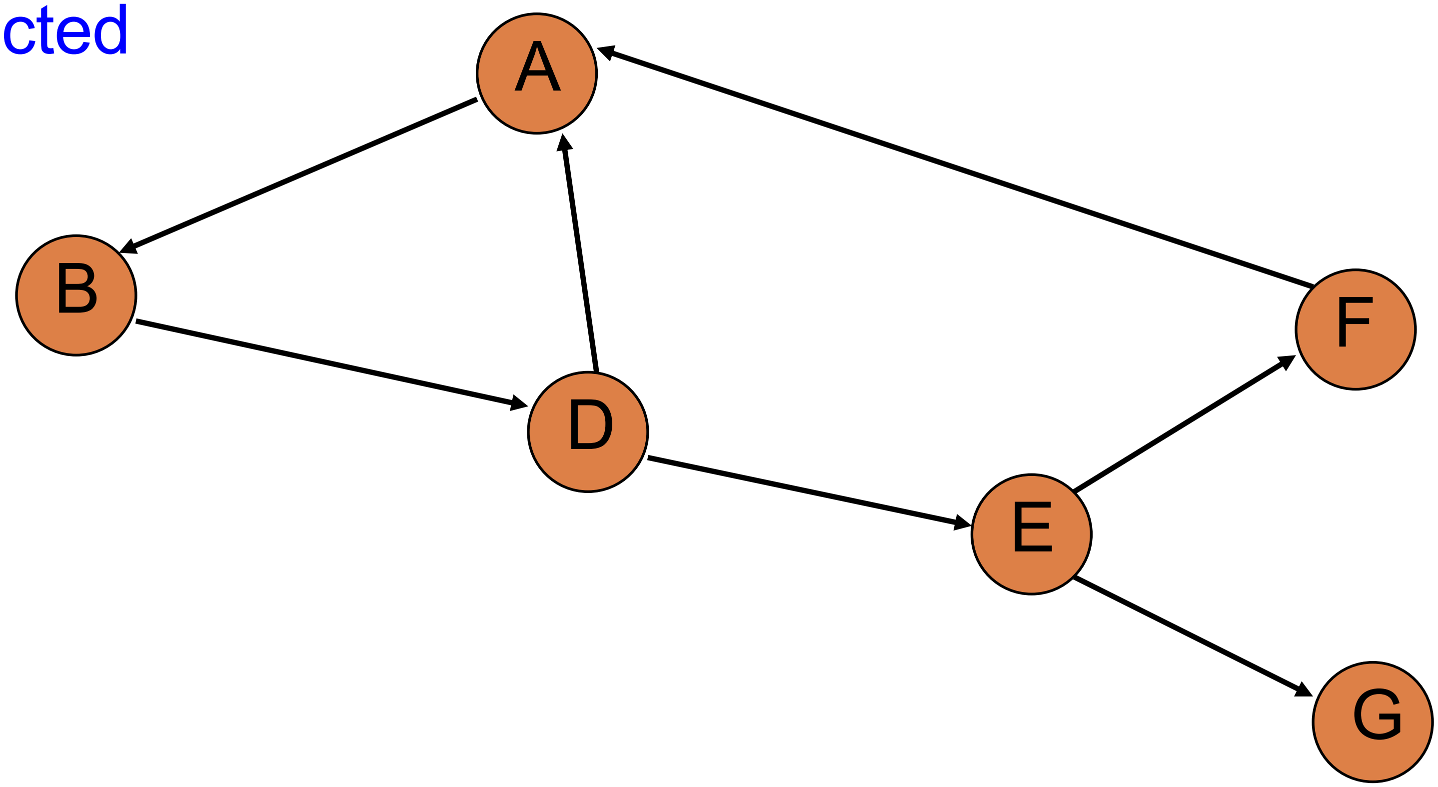
Is this graph
strongly connected?



Terminology

Strongly connected (directed graphs) –
Every two vertices are reachable by a path

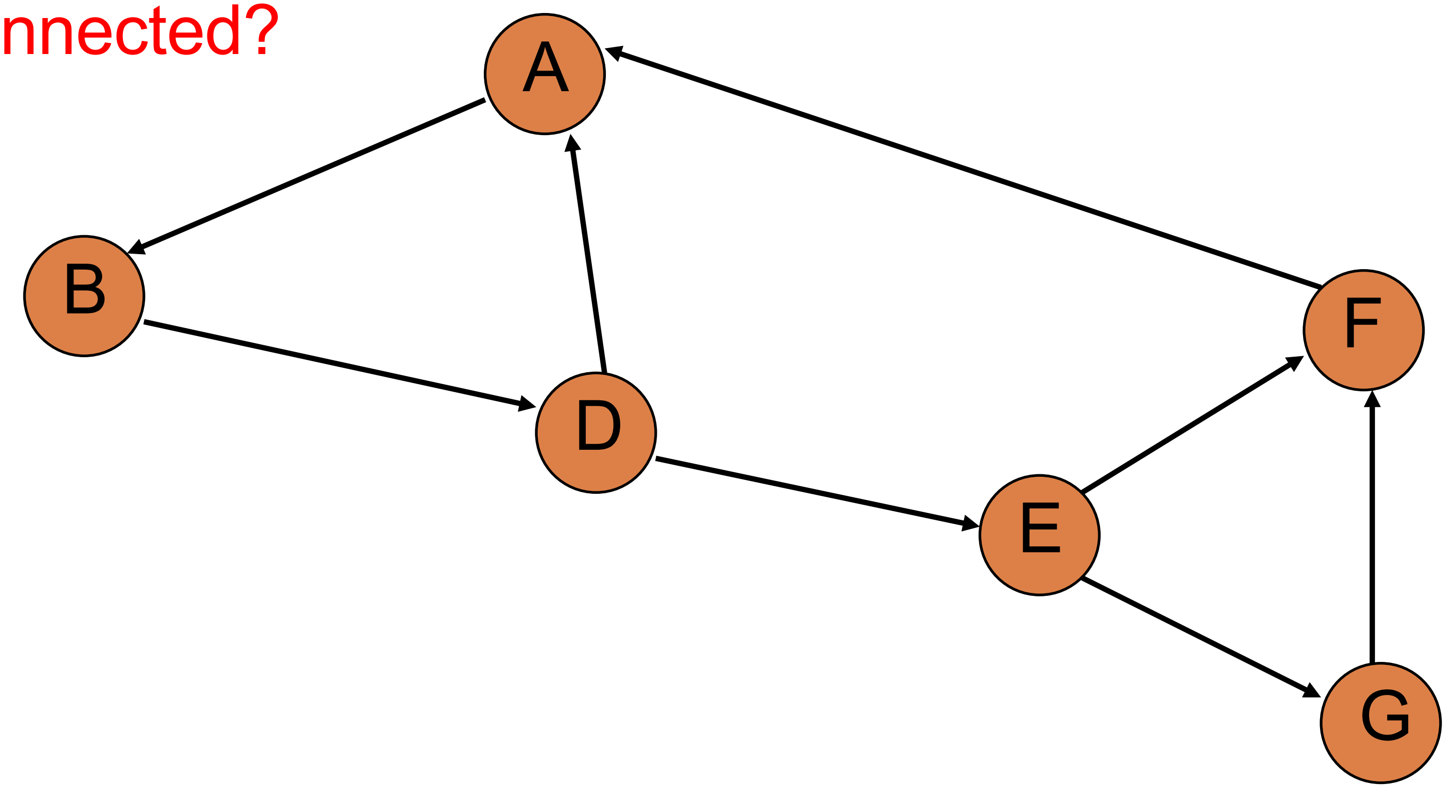
not strongly
connected



Terminology

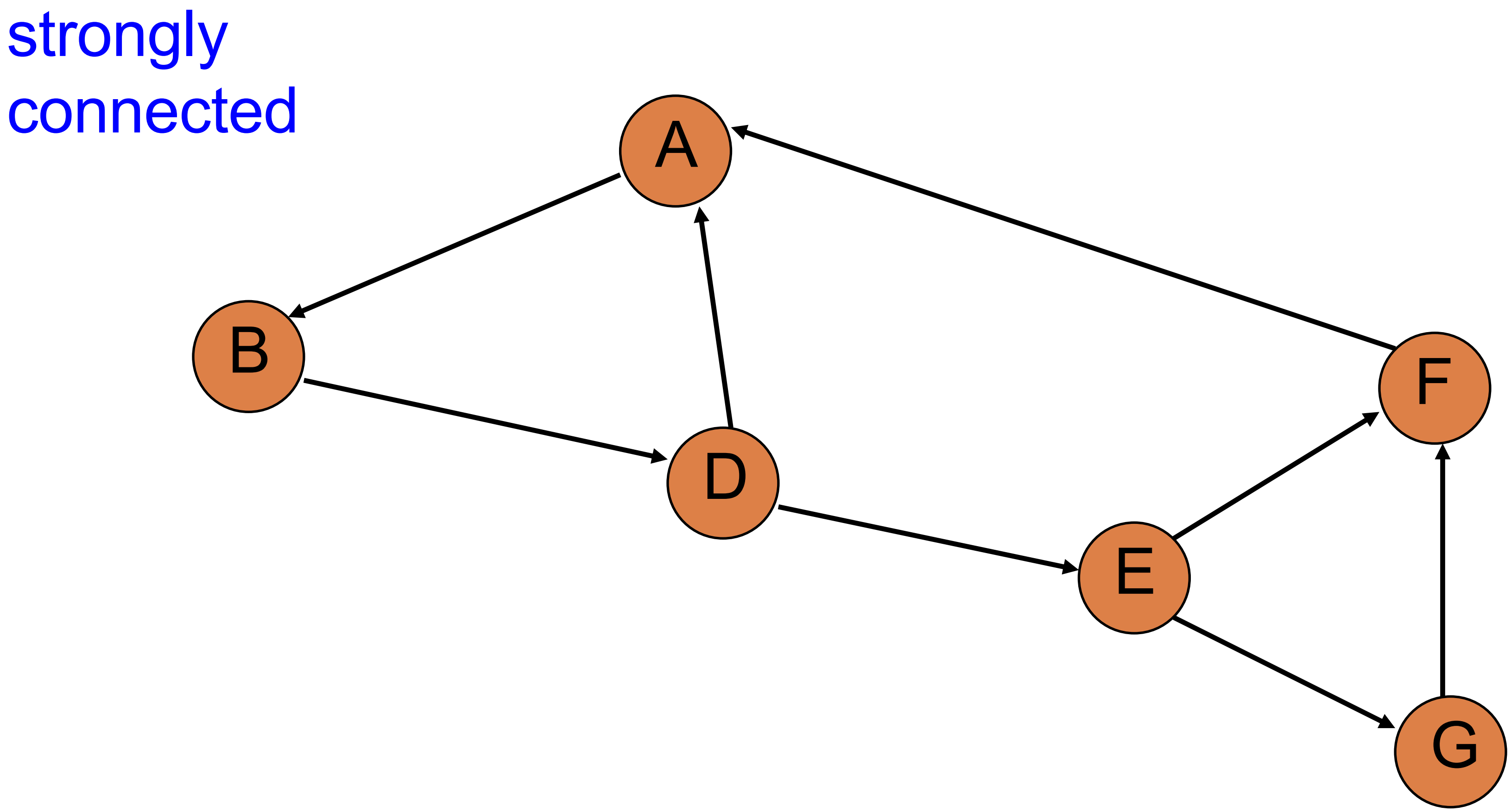
Strongly connected (directed graphs) –
Every two vertices are reachable by a path

Is this graph
strongly connected?



Terminology

Strongly connected (directed graphs) –
Every two vertices are reachable by a path



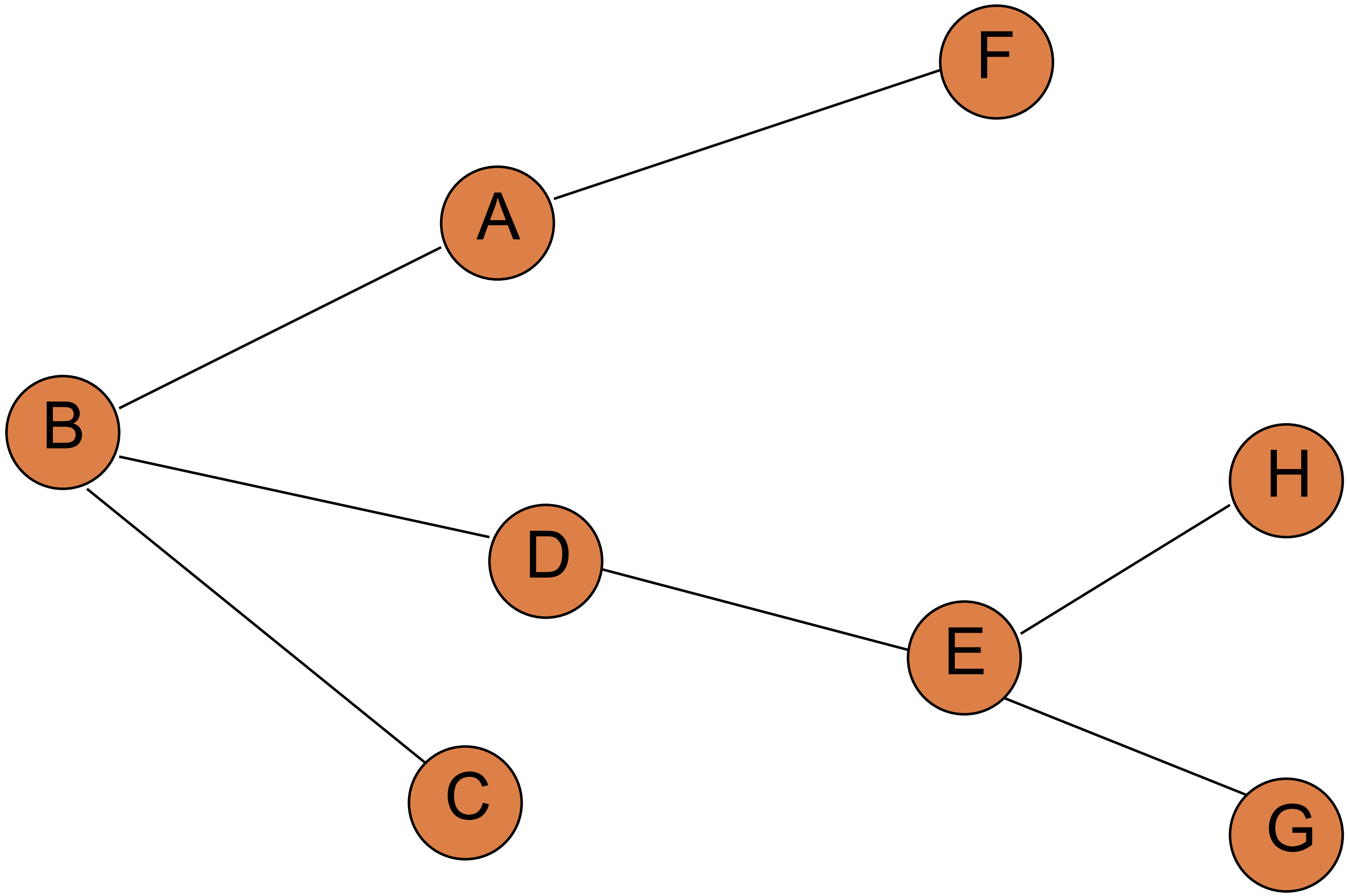
Graphs aren't new…

Have we seen graphs in this class already?

Trees!

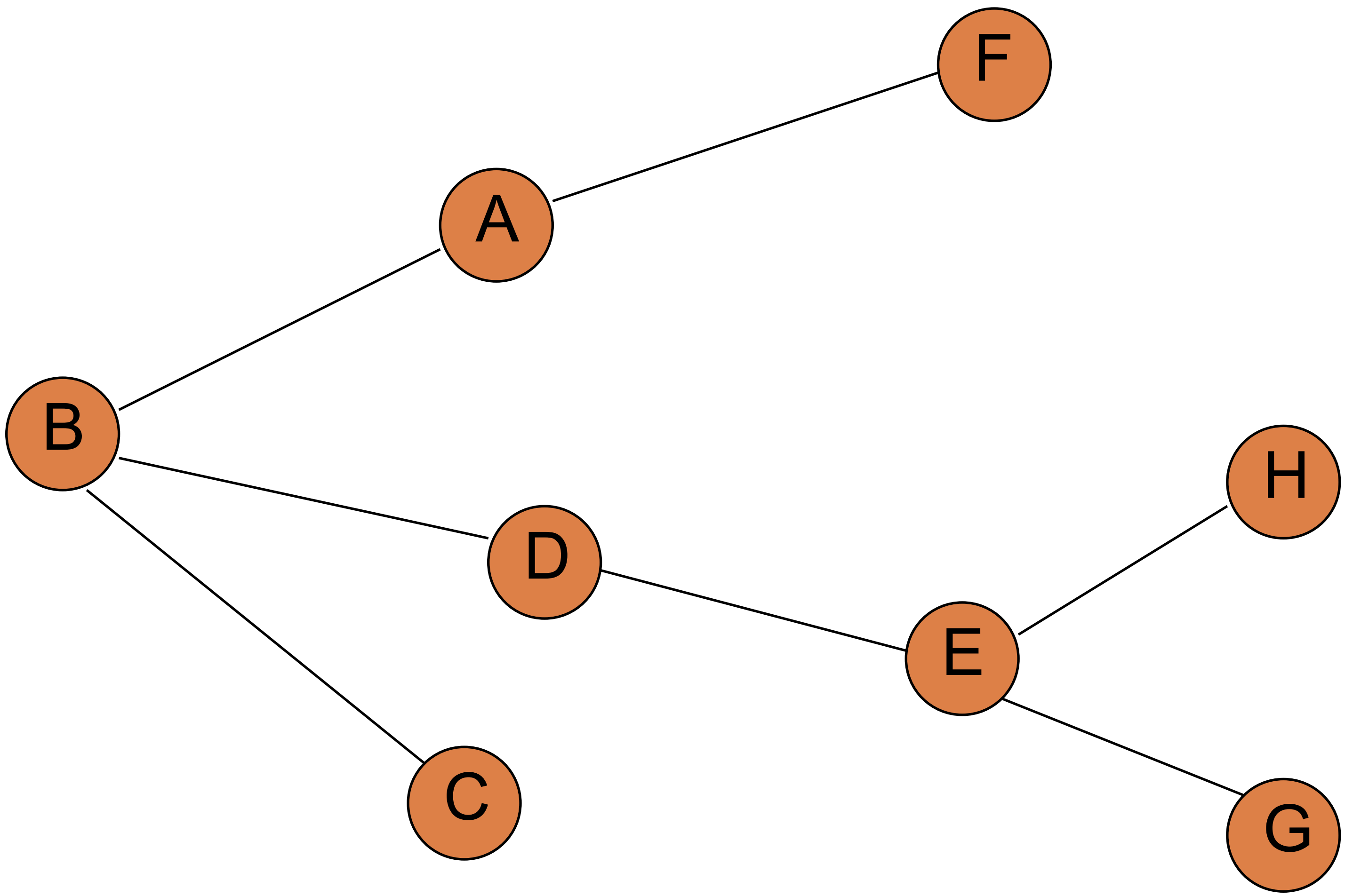
Different types of graphs

What is a tree (in our terminology)?



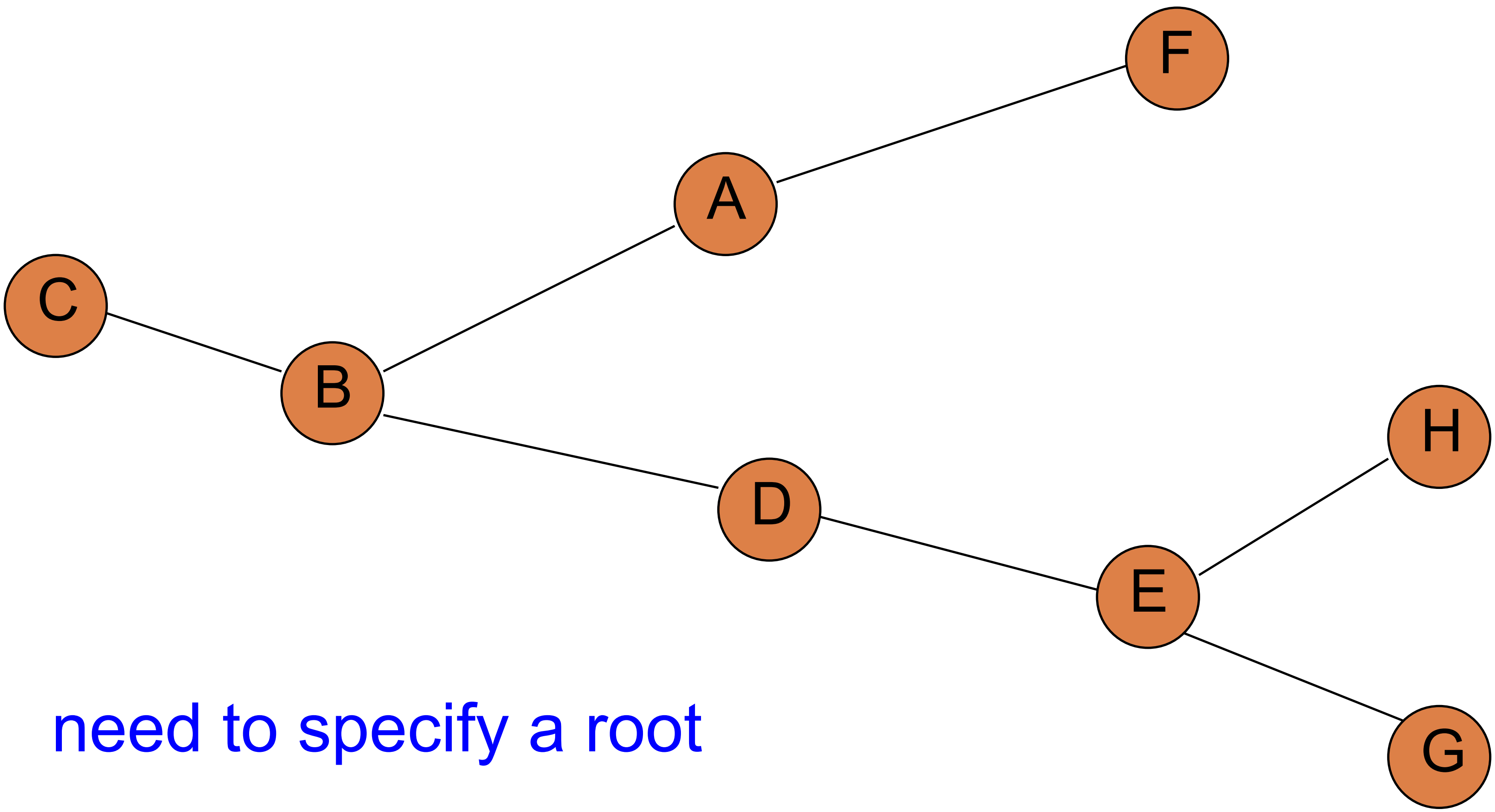
Different types of graphs

Tree – connected, undirected graph without any cycles



Different types of graphs

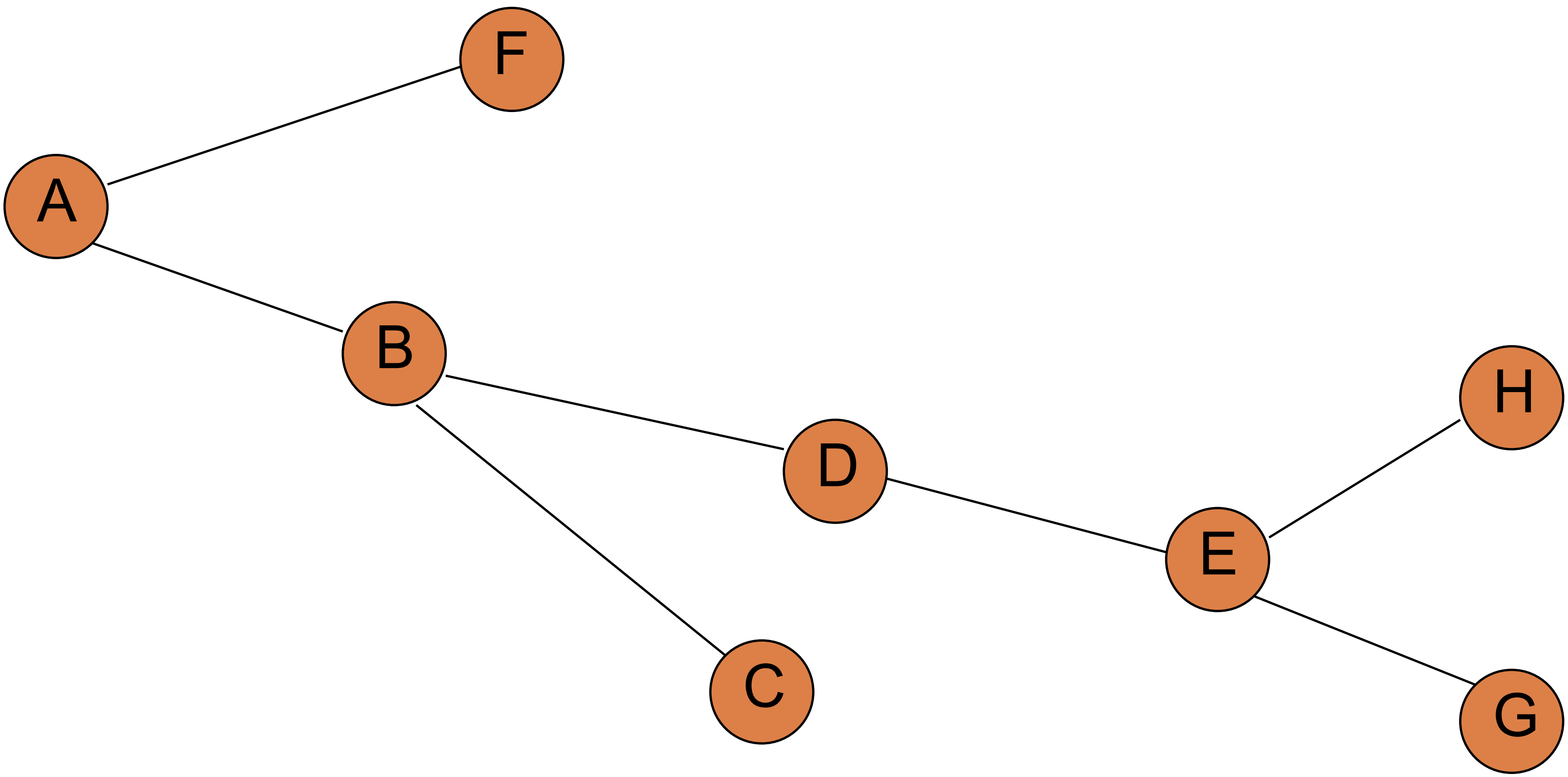
Tree – connected, undirected graph without any cycles



need to specify a root

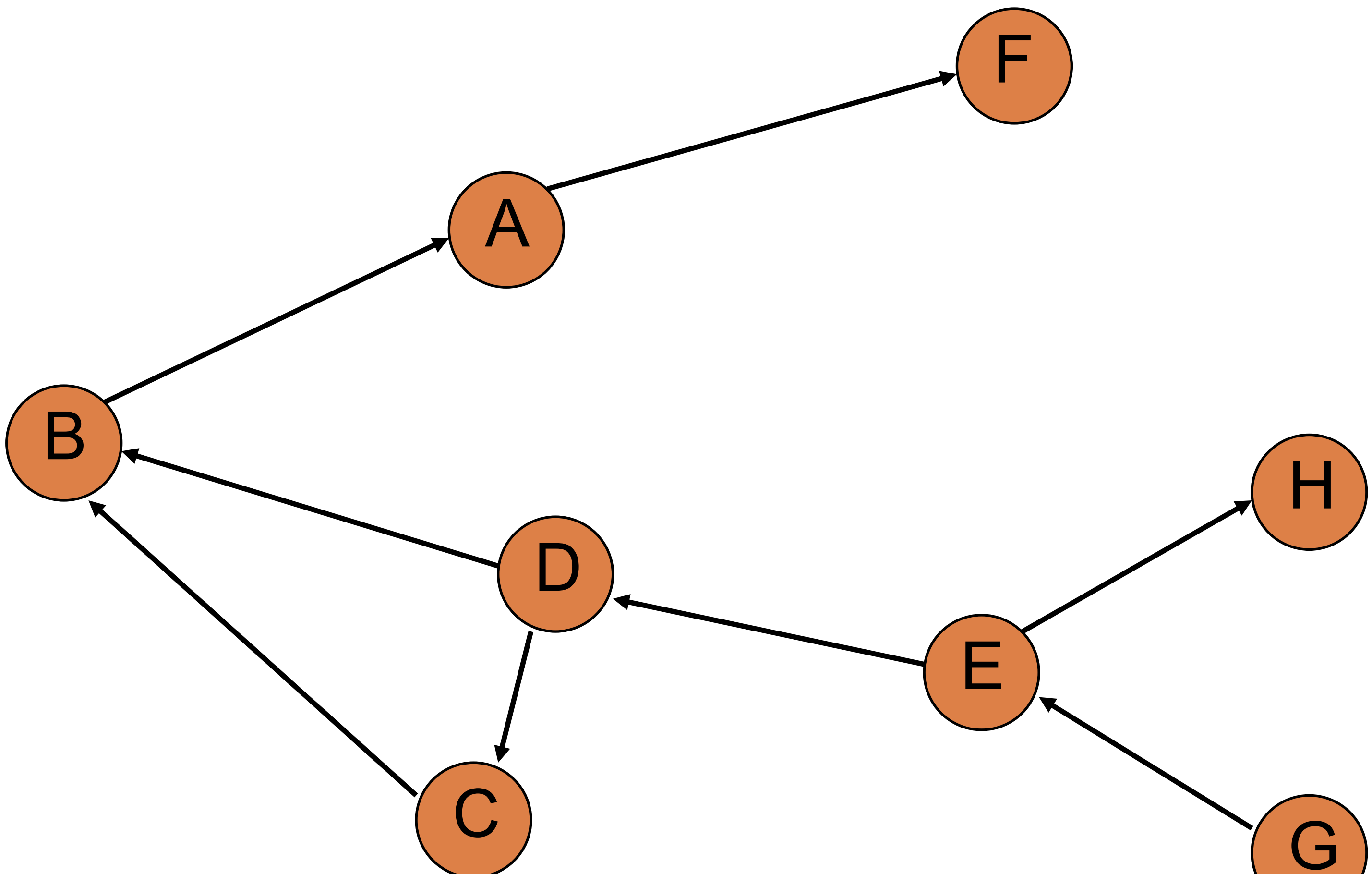
Different types of graphs

Tree – connected, undirected graph without any cycles



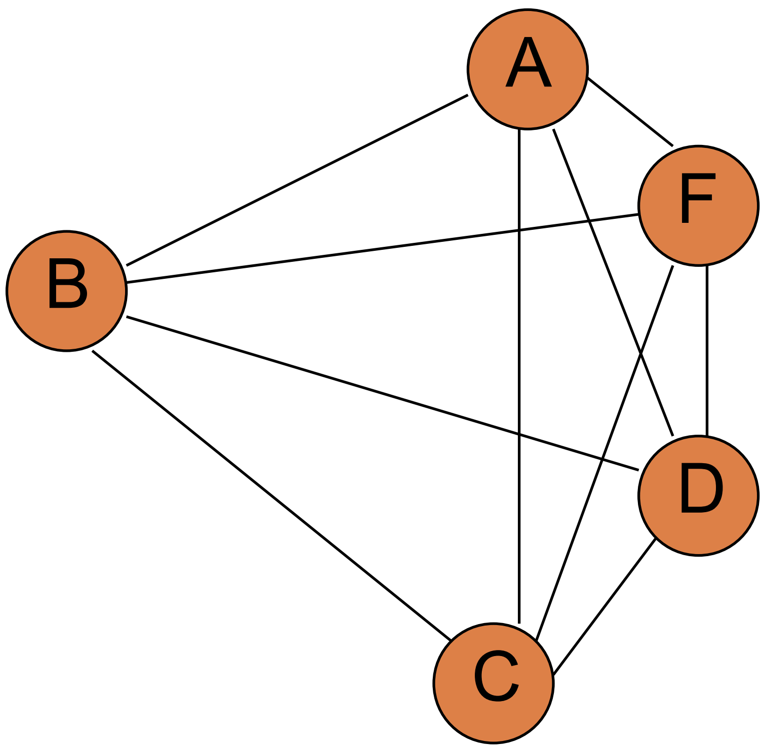
Different types of graphs

DAG – directed, acyclic graph



Different types of graphs

Complete graph – an edge exists between every node



Graph questions?

Does it have a cycle?

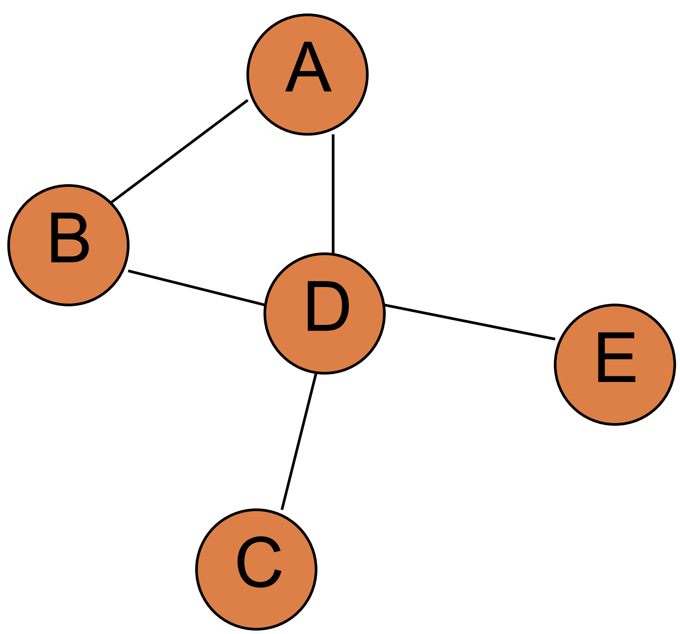
Is it connected? Strongly connected?

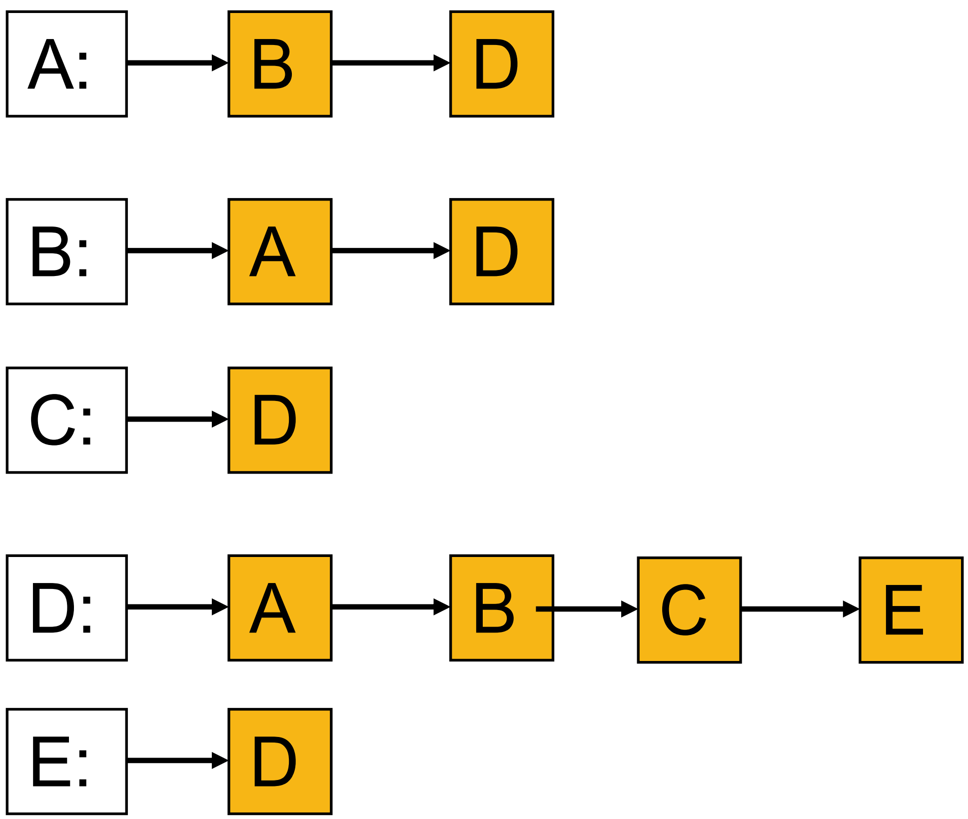
Is there a path from a to b?

What is the shortest path from a to b? In number of edges? In sum of the edge weights?

Representing graphs

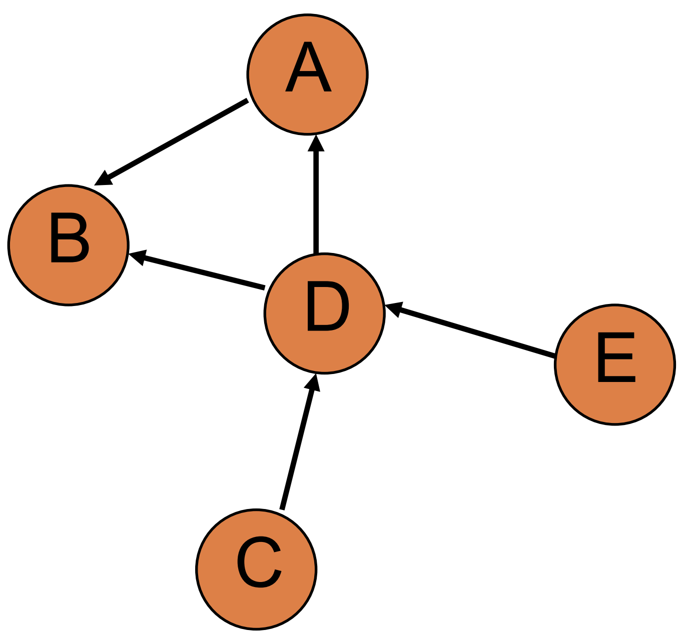
Adjacency list – Each vertex u ∈ V contains an adjacency list of the set of vertices v such that there exists an edge (u,v) ∈ E

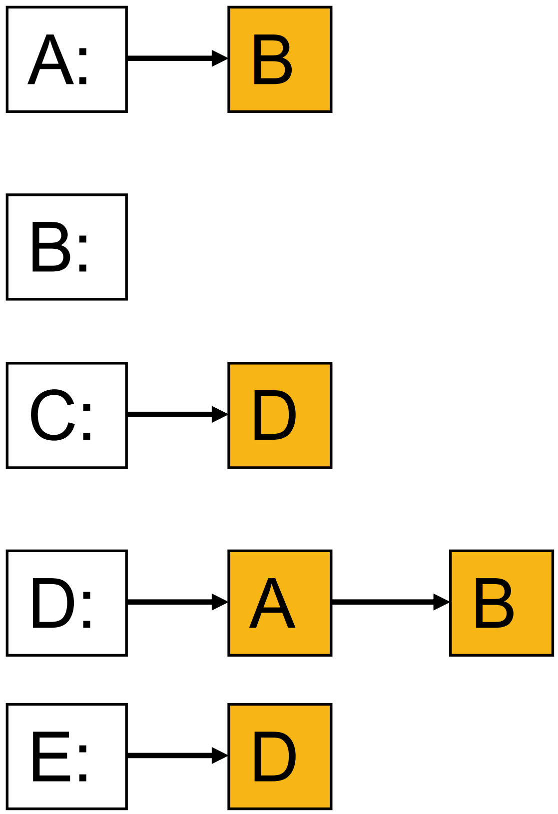




Representing graphs

Adjacency list – Each vertex u ∈ V contains an adjacency list of the set of vertices v such that there exists an edge (u,v) ∈ E

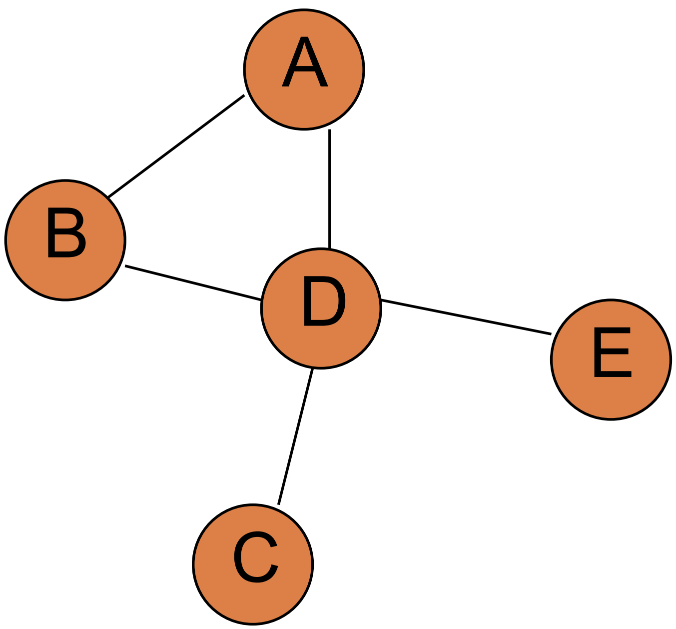




Representing graphs

Adjacency matrix – A |V|x|V| matrix A such that:

$$a_{ij} = \begin{cases} 1 & \text{if } (i,j) \in E \\ 0 & \text{otherwise} \end{cases}$$

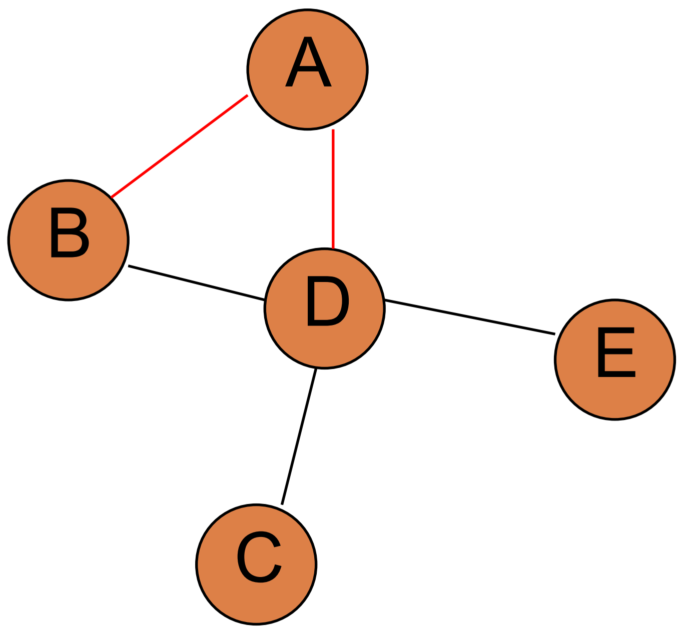


	A	B	C	D	E
A	0	1	0	1	0
B	1	0	0	1	0
C	0	0	0	1	0
D	1	1	1	0	1
E	0	0	0	1	0

Representing graphs

Adjacency matrix – A |V| x |V| matrix A such that:

$$a_{ij} = \begin{cases} 1 & \text{if } (i,j) \in E \\ 0 & \text{otherwise} \end{cases}$$

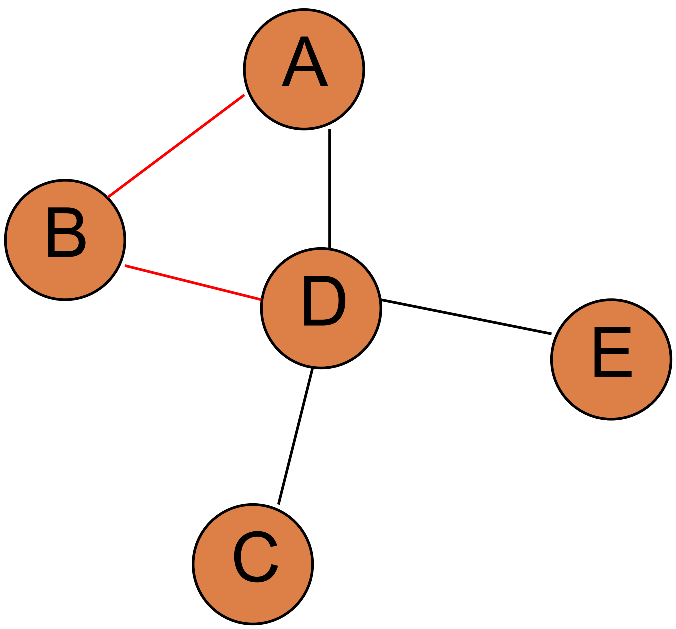


	A	B	C	D	E
A	0	**1**	0	**1**	0
B	1	0	0	1	0
C	0	0	0	1	0
D	1	1	1	0	1
E	0	0	0	1	0

Representing graphs

Adjacency matrix – A |V|x|V| matrix A such that:

$$a_{ij} = \begin{cases} 1 & \text{if } (i,j) \in E \\ 0 & \text{otherwise} \end{cases}$$

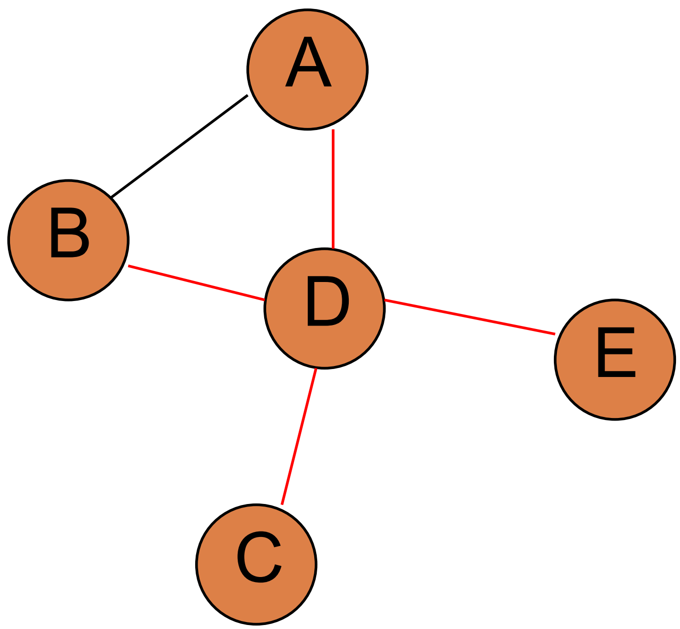


	A	B	C	D	E
A	0	1	0	1	0
B	1	0	0	1	0
C	0	0	0	1	0
D	1	1	1	0	1
E	0	0	0	1	0

Representing graphs

Adjacency matrix – A |V| x |V| matrix A such that:

$$a_{ij} = \begin{cases} 1 & \text{if } (i,j) \in E \\ 0 & \text{otherwise} \end{cases}$$

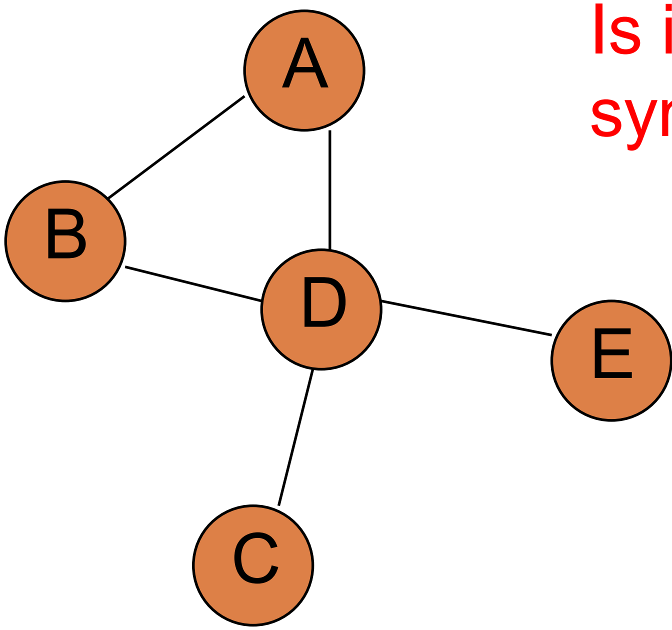


	A	B	C	D	E
A	0	1	0	1	0
B	1	0	0	1	0
C	0	0	0	1	0
D	1	1	1	0	1
E	0	0	0	1	0

Representing graphs

Adjacency matrix – A |V| x |V| matrix A such that:

$$a_{ij} = \begin{cases} 1 & \text{if } (i, j) \in E \\ 0 & \text{otherwise} \end{cases}$$



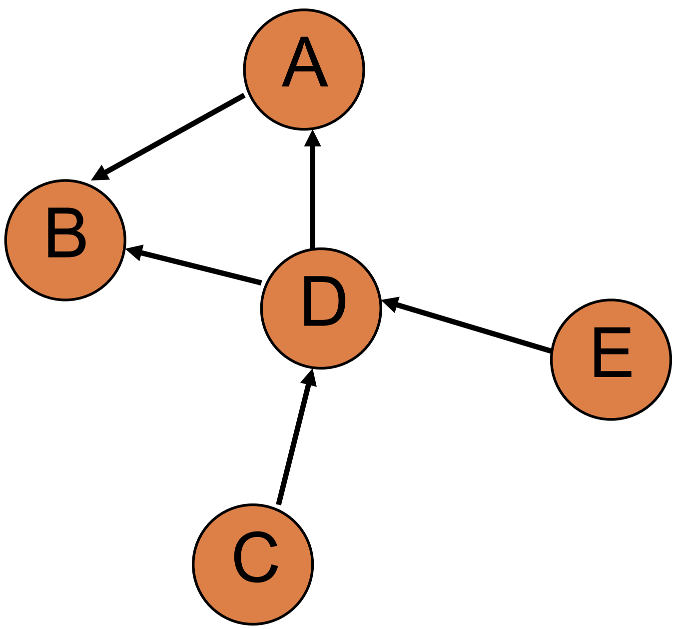
Is it always symmetric?

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

Representing graphs

Adjacency matrix – A |V| x |V| matrix A such that:

$$a_{ij} = \begin{cases} 1 & \text{if } (i, j) \in E \\ 0 & \text{otherwise} \end{cases}$$



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

Adjacency list vs. adjacency matrix

Adjacency list	Adjacency matrix

Pros/Cons?

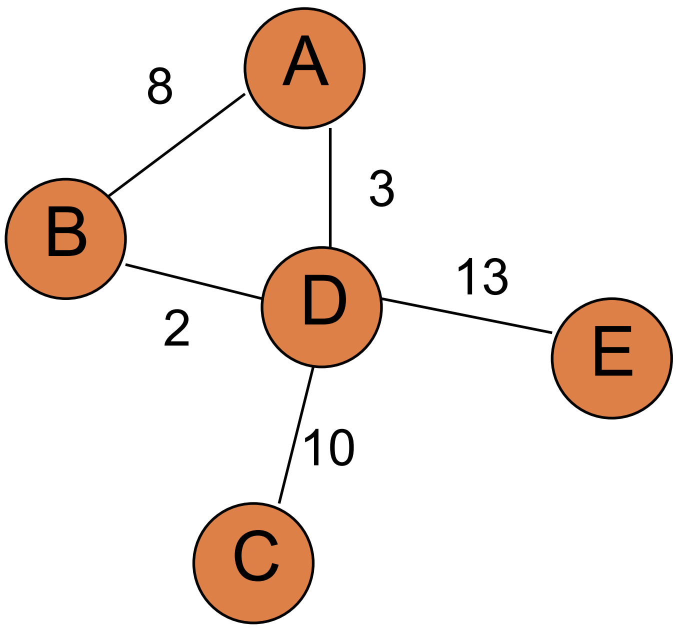
Adjacency list vs. adjacency matrix

Adjacency list	Adjacency matrix
Sparse graphs (e.g. web)	Dense graphs
Space efficient	Constant time lookup to discover if an edge exists
Must traverse the adjacency list to discover is an edge exists	Simple to implement
	For non-weighted graphs, only requires boolean matrix

Weighted graphs

Adjacency list

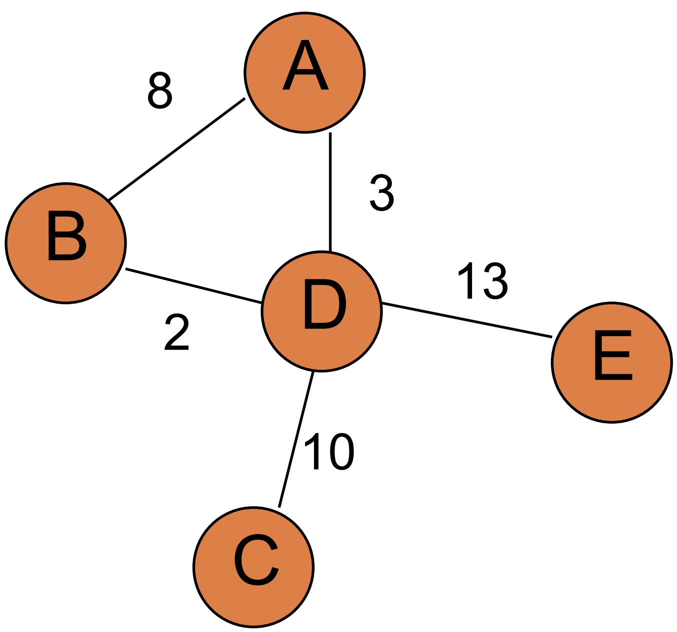
- store the weight as an additional field in the list



Weighted graphs

Adjacency matrix

$$a_{ij} = \begin{cases} weight & \text{if } (i,j) \in E \\ 0 & \text{otherwise} \end{cases}$$



	A	B	C	D	E
A	0	8	0	3	0
B	8	0	0	2	0
C	0	0	0	10	0
D	3	2	10	0	13
E	0	0	0	13	0