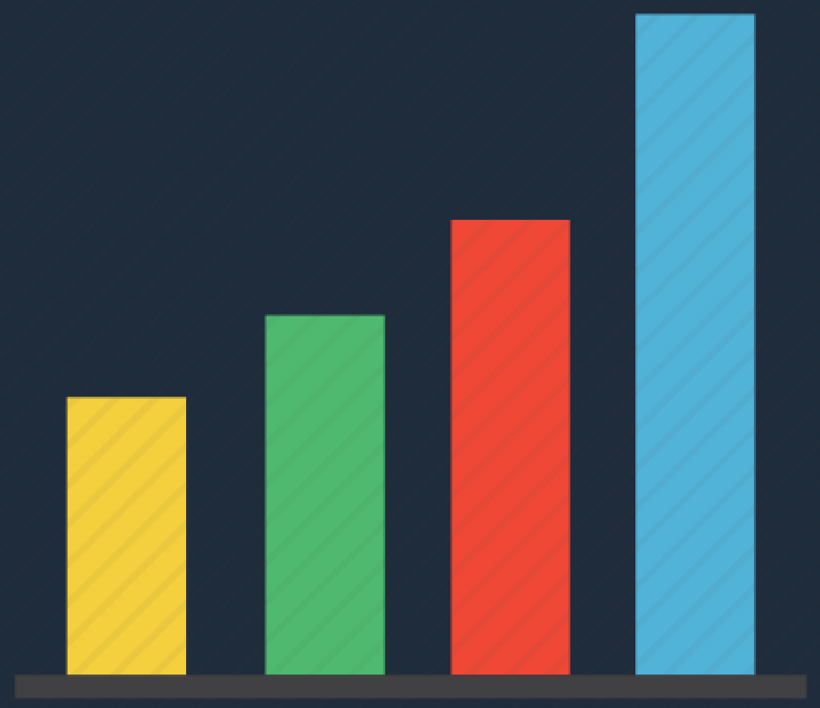
GRAPH THEORY

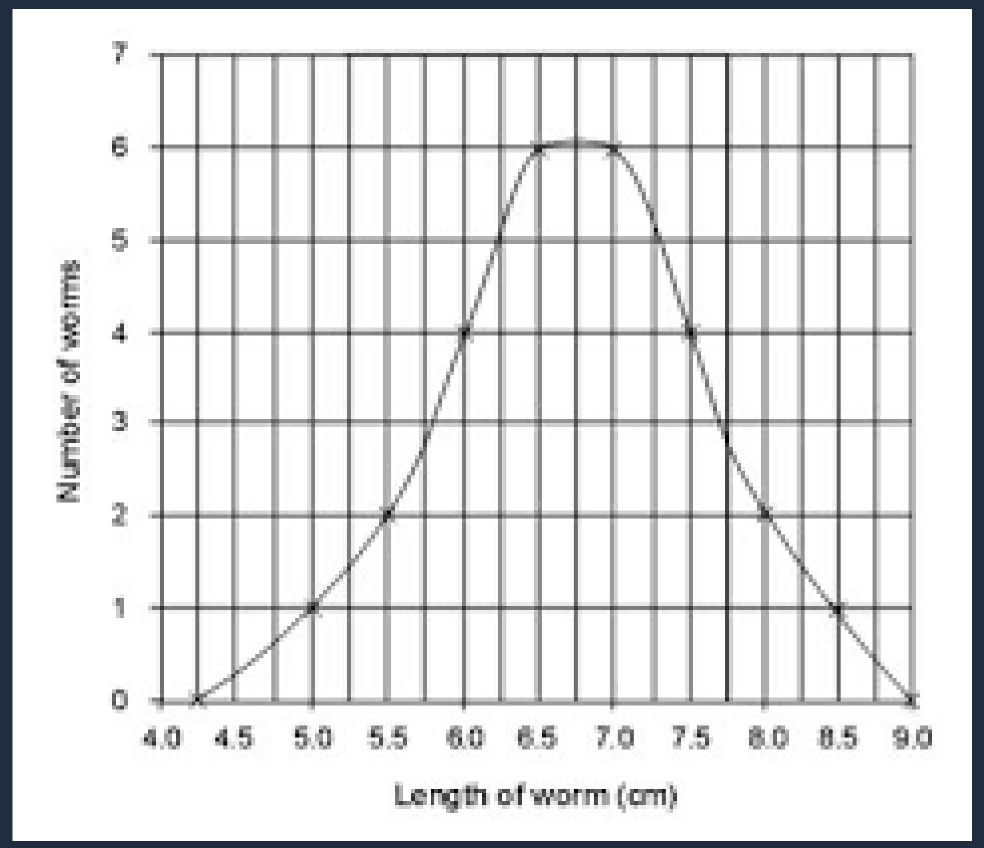
A Cursory Introduction to Graph Theory in Computer Science

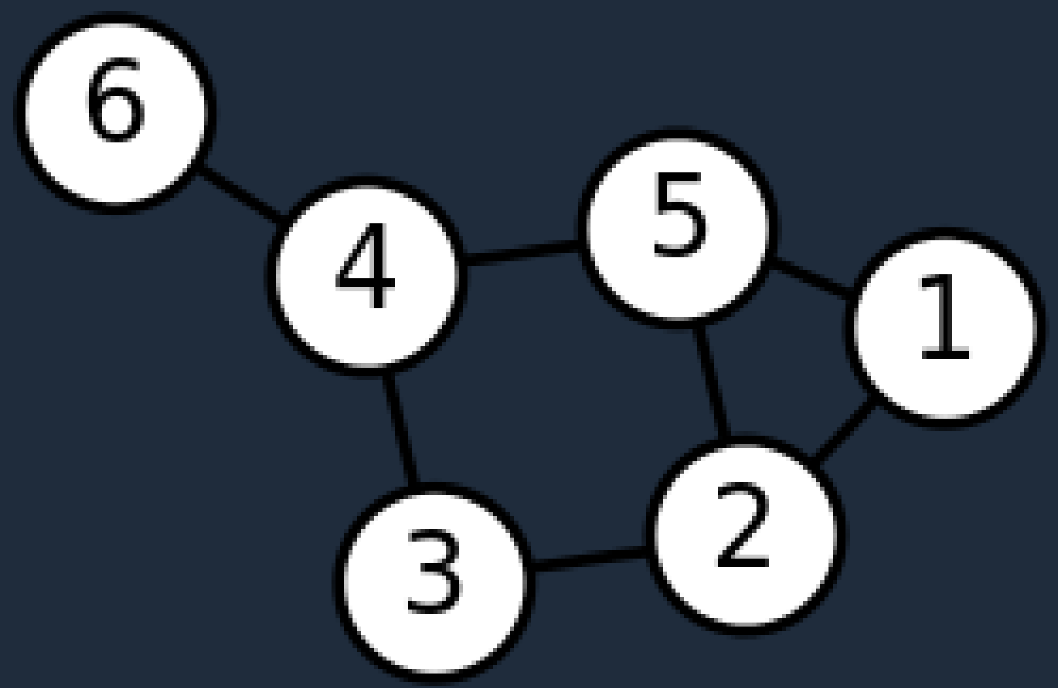
PRESENTED BY: **Brian Chen**

CSEC

What is a Graph?

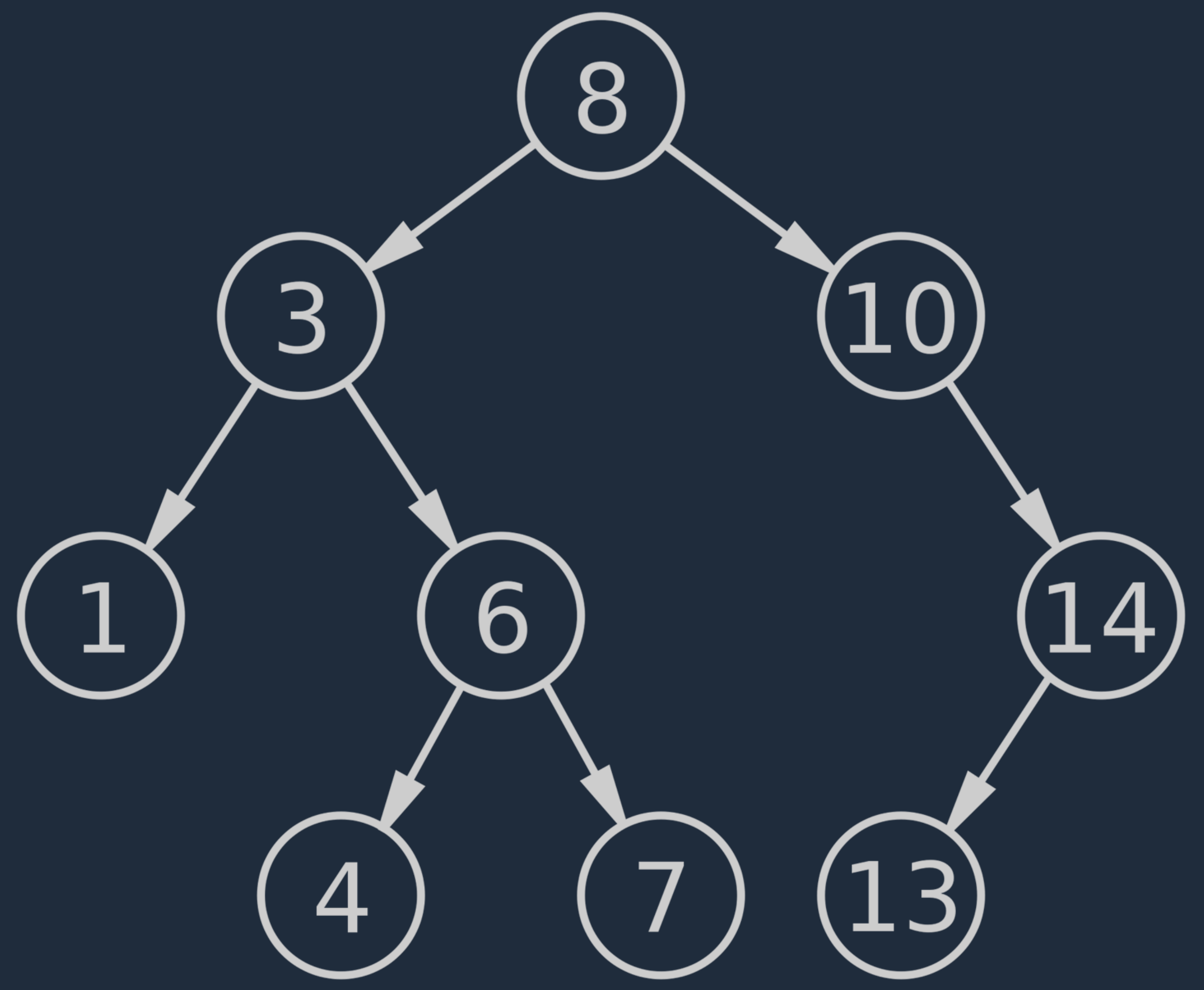


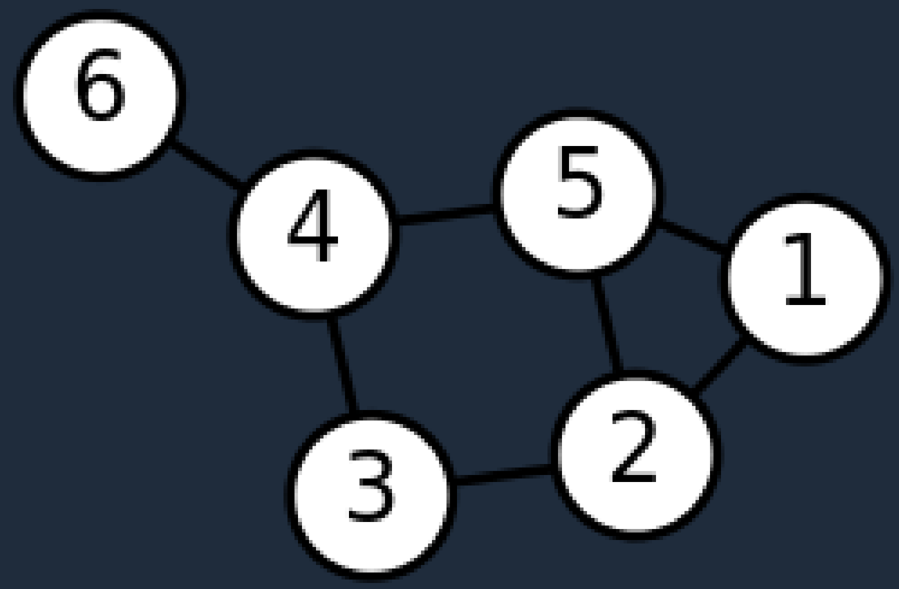




What does this remind us of?

Similar to a Tree..?



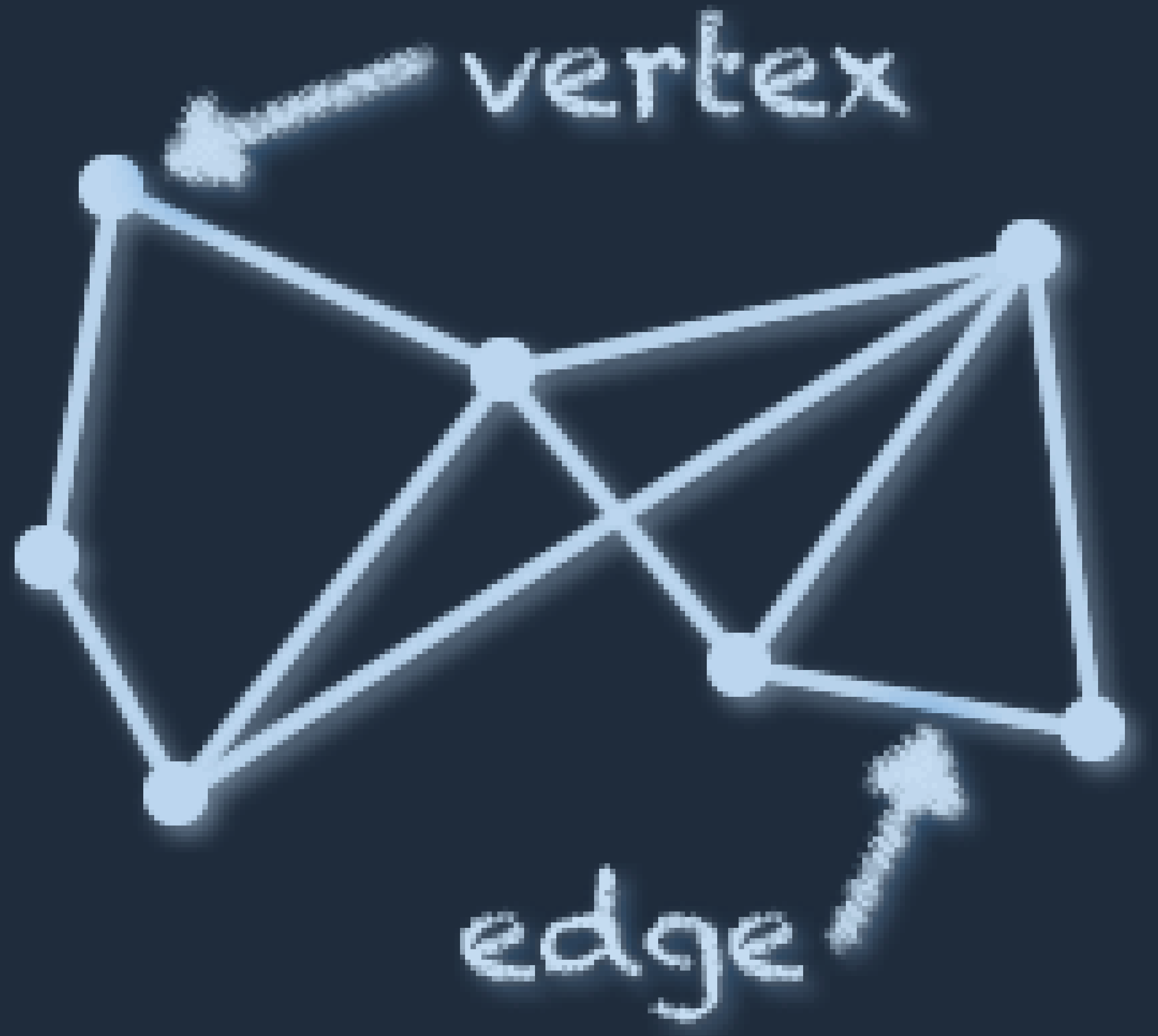


Yep, Trees are a **special type of Graph.**

In particular, **Trees** are **Graphs** that:

- Have no loops
- Have no circuits
- There are no self-loops
- Only one path between two vertices

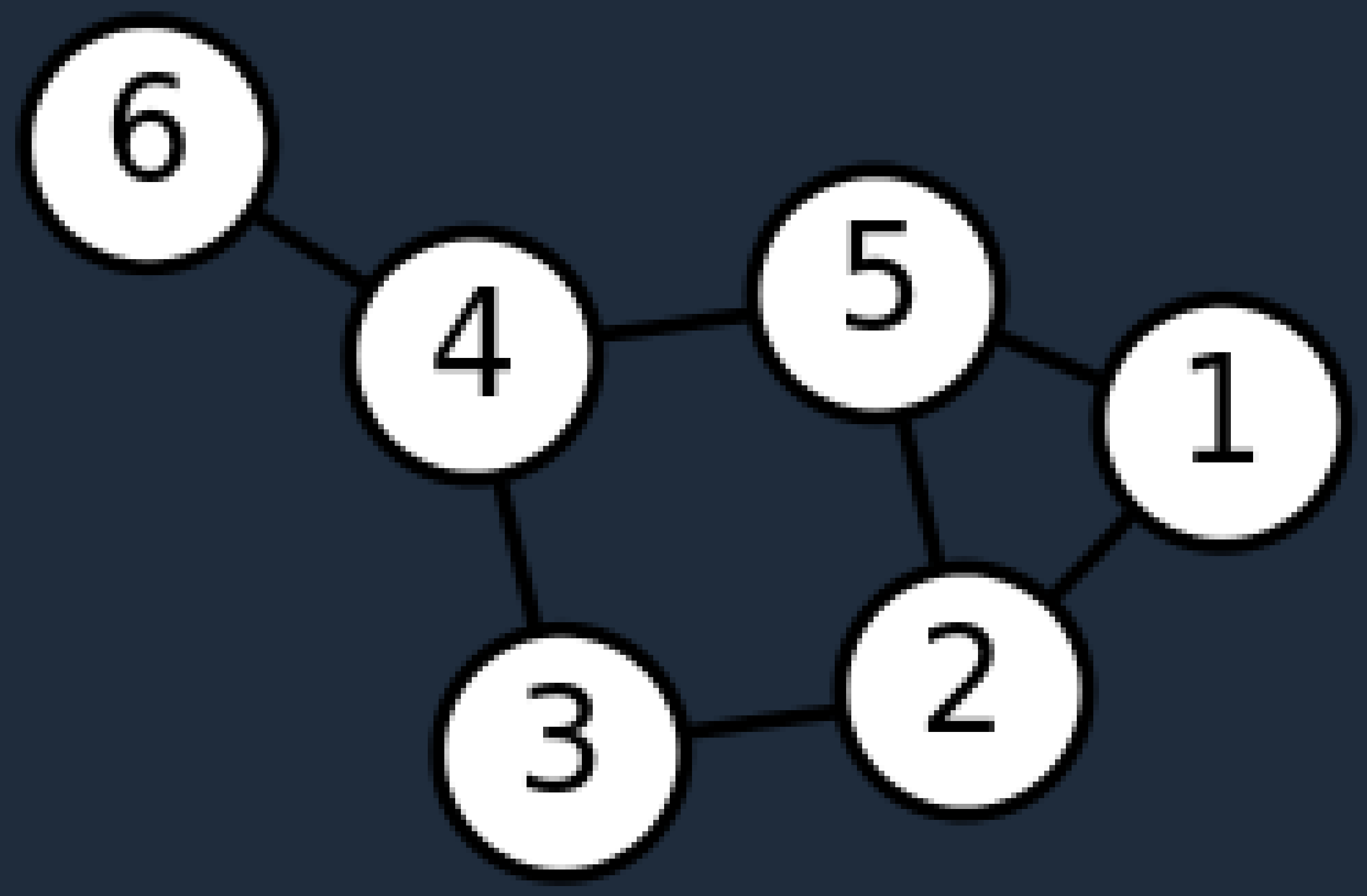
The Definition of a Graph



A graph is usually made out of:

- Nodes called **Vertices**
  - They can contain values
- **Lines called Edges**
  - They can be assigned distance values

How do we Define a Graph?



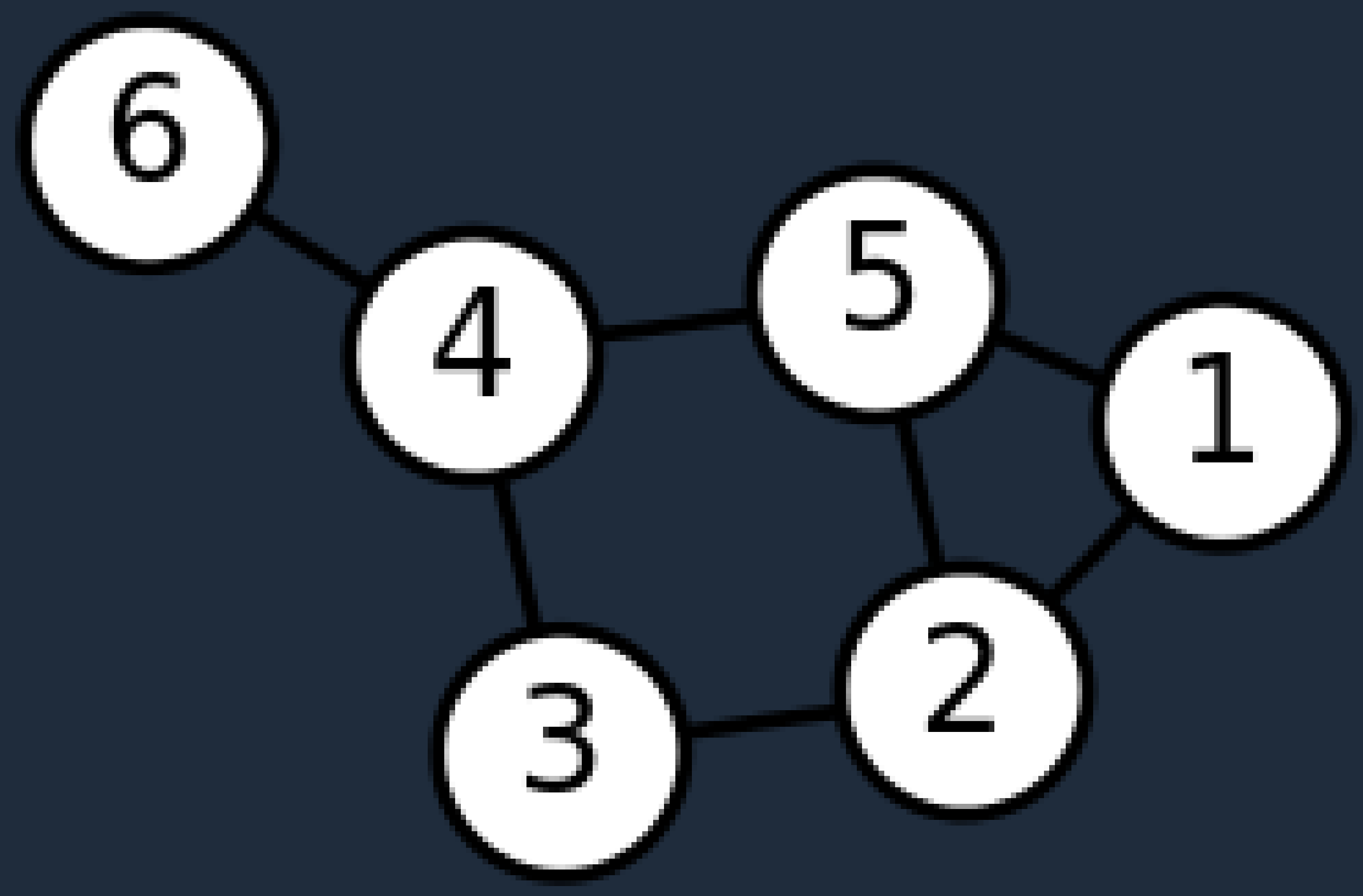
To define a graph, we give a set of the **Vertices** and the **Edges**

For example, for the graph on the left:

V = {1, 2, 3, 4, 5, 6}

E = {(6, 4), (4, 5), (4, 3), (3, 2), (5, 2), (5, 1), (2, 1)}

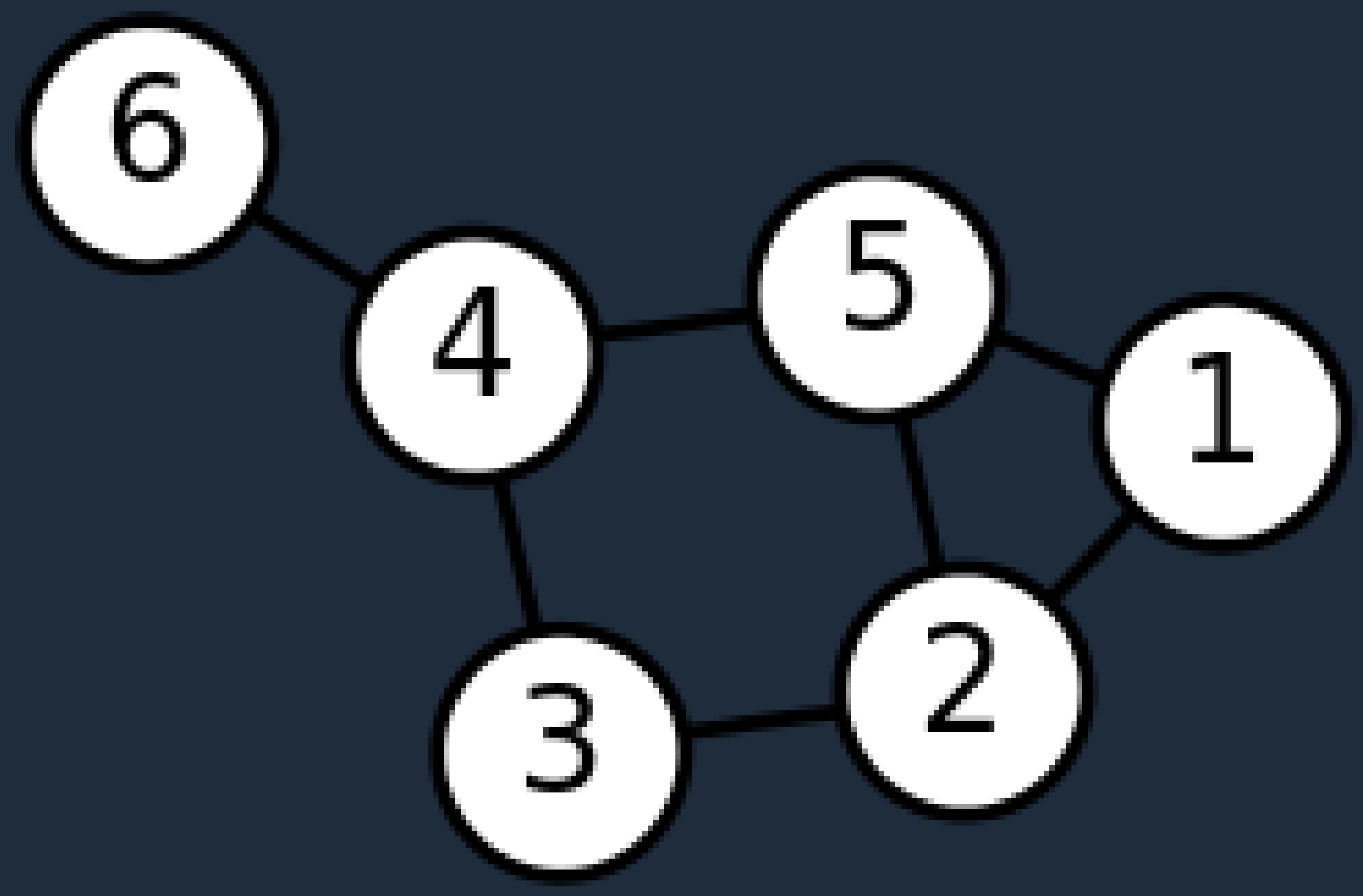
What can we do with graphs?



Let's think of some practical usages for Graphs.

Let's say you're stuck in **City 6** and you want to get to **City 1**. You look at a bus map and notice that City 6 doesn't connect to City 1 directly, but you have to transfer busses at different cities. You can represent this using a Graph.

Directed and Undirected Graphs

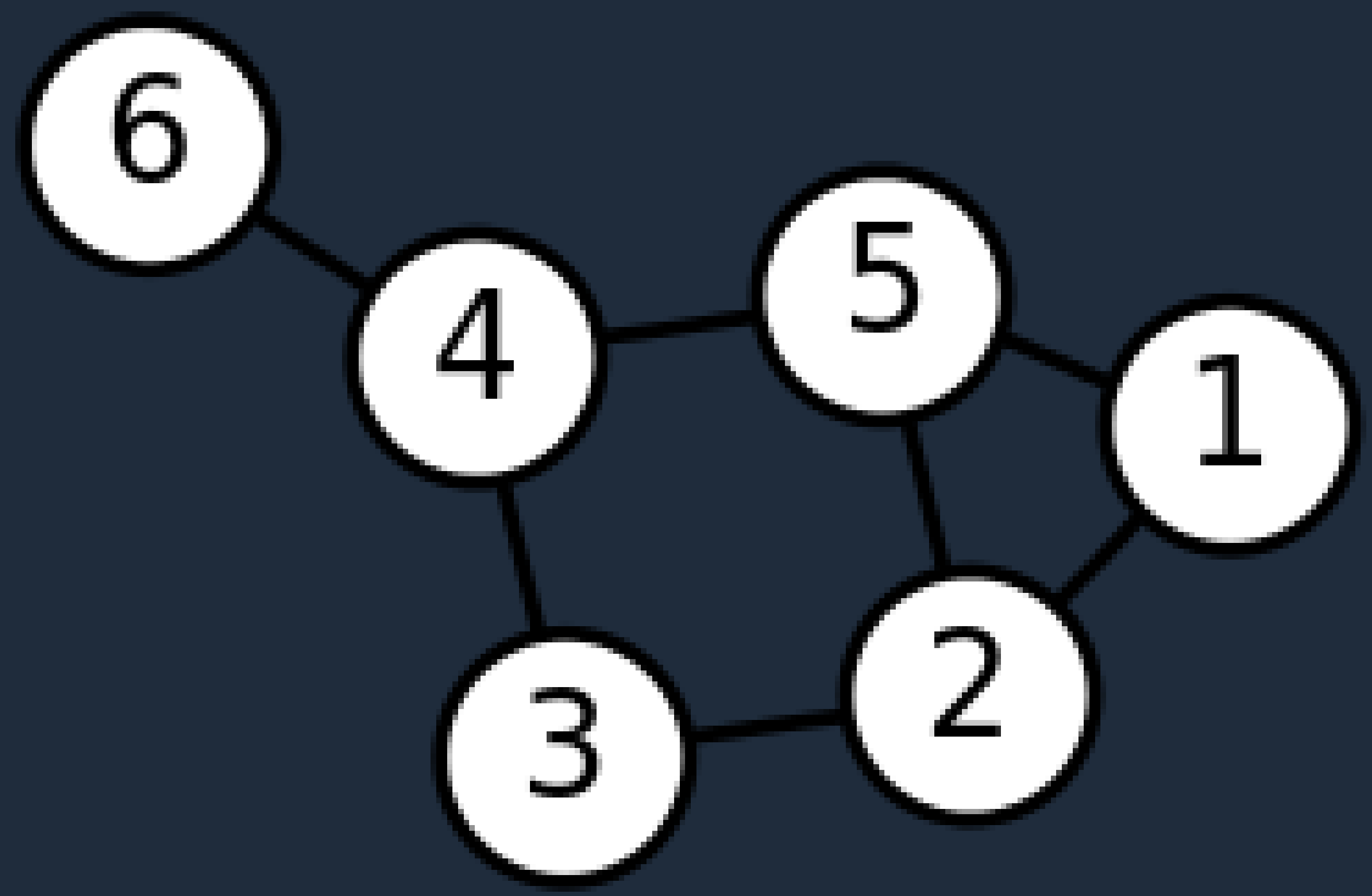


Directed Graphs – Edges can be bidirectional or unidirectional

Undirected Graphs – Edges are always assumed to be bidirectional

Logically, using our bus example, we should have a undirected graph

Node Weighted and Unweighted Graphs

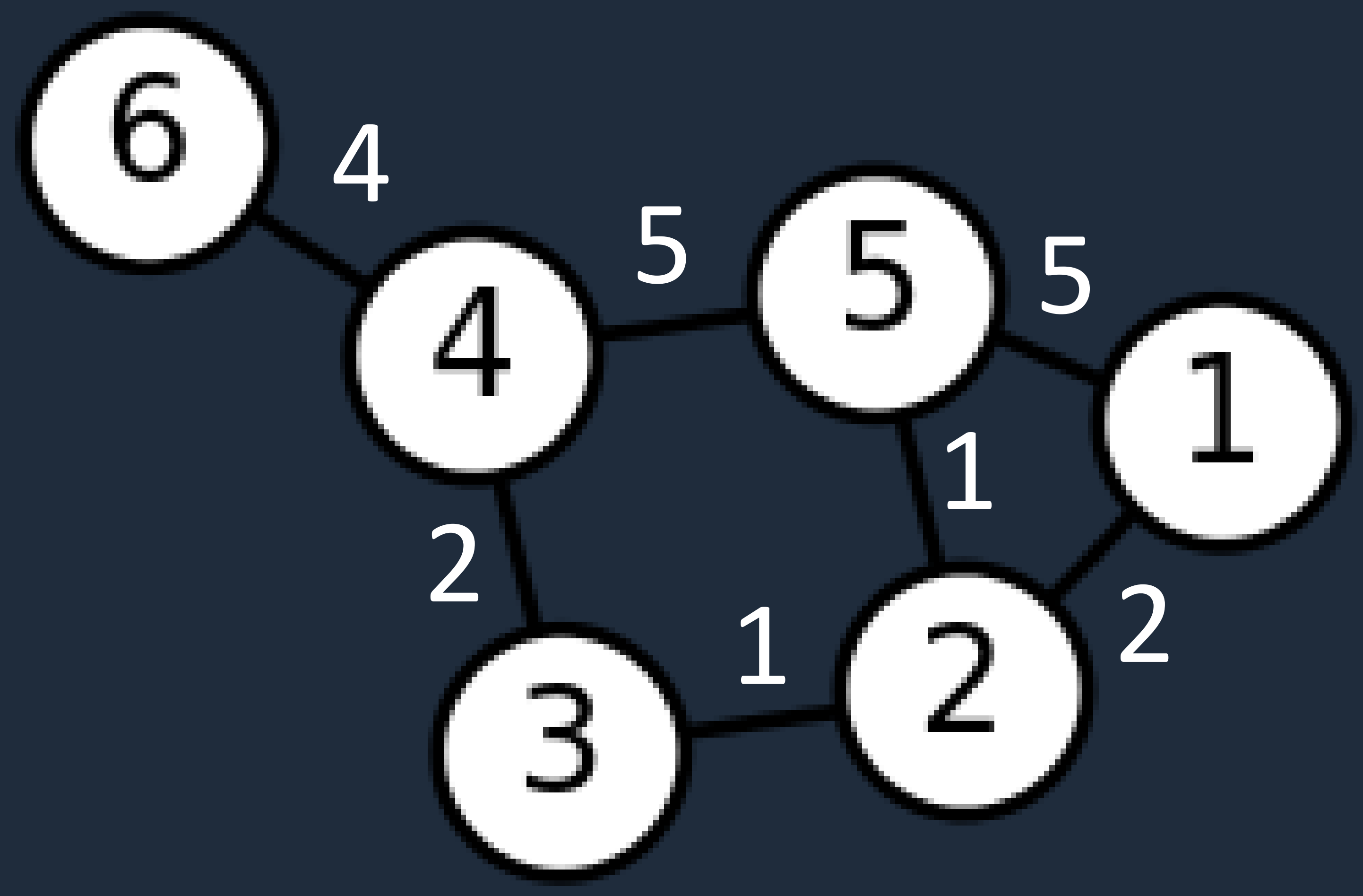


Weighted Graphs – Nodes are assigned a 'weight' or value

Unweighted Graphs – Nodes are not assigned any values

Logically, using our bus example, we should have a node-unweighted graph

Edge Weighted and Unweighted Graphs

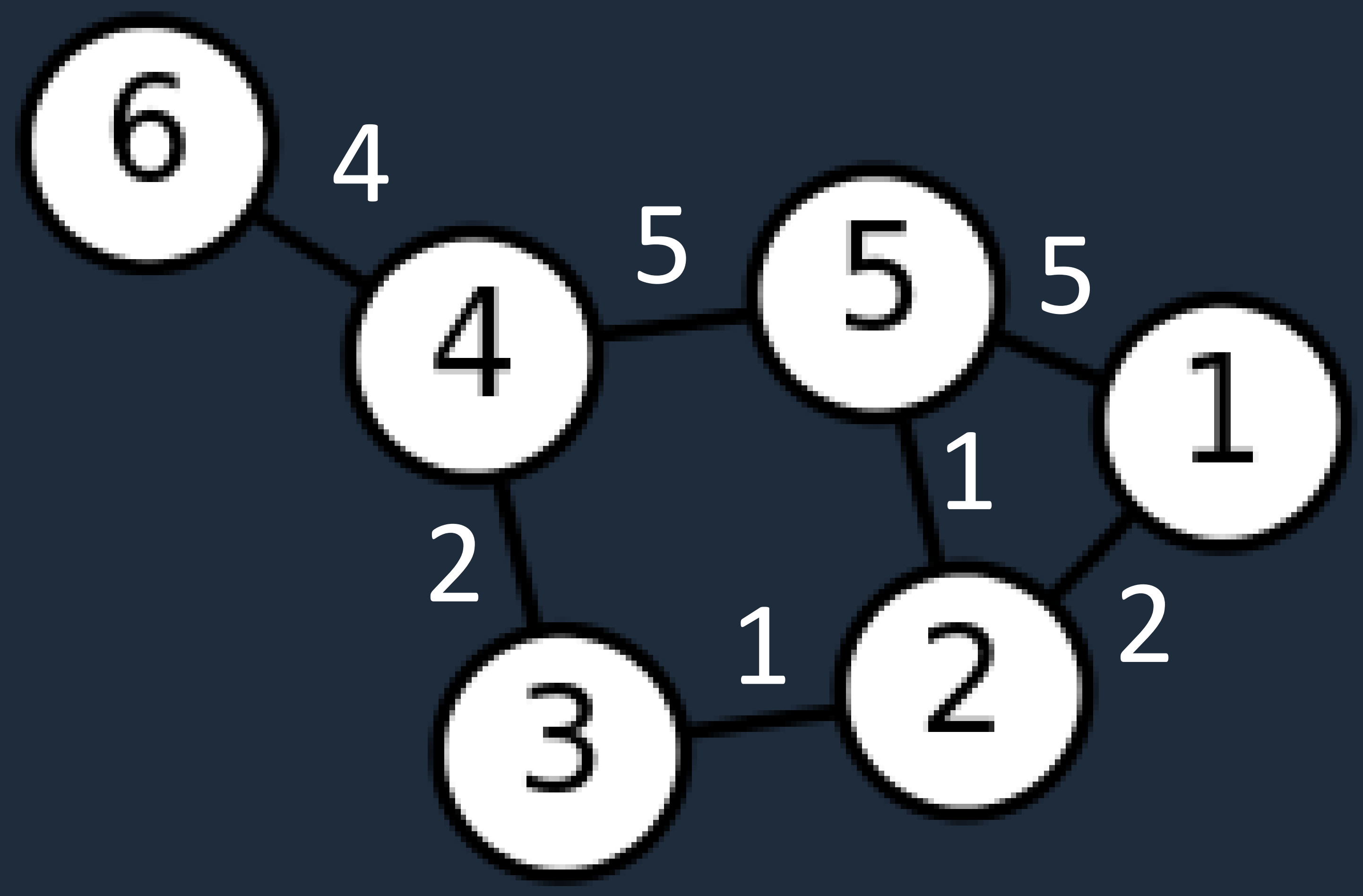


Weighted Graphs – Edges are assigned a ‘weight’ or distance value

Unweighted Graphs – Edges are assigned no distance values – assumed to be one

Logically, using our bus example, we should have a edge-weighted graph

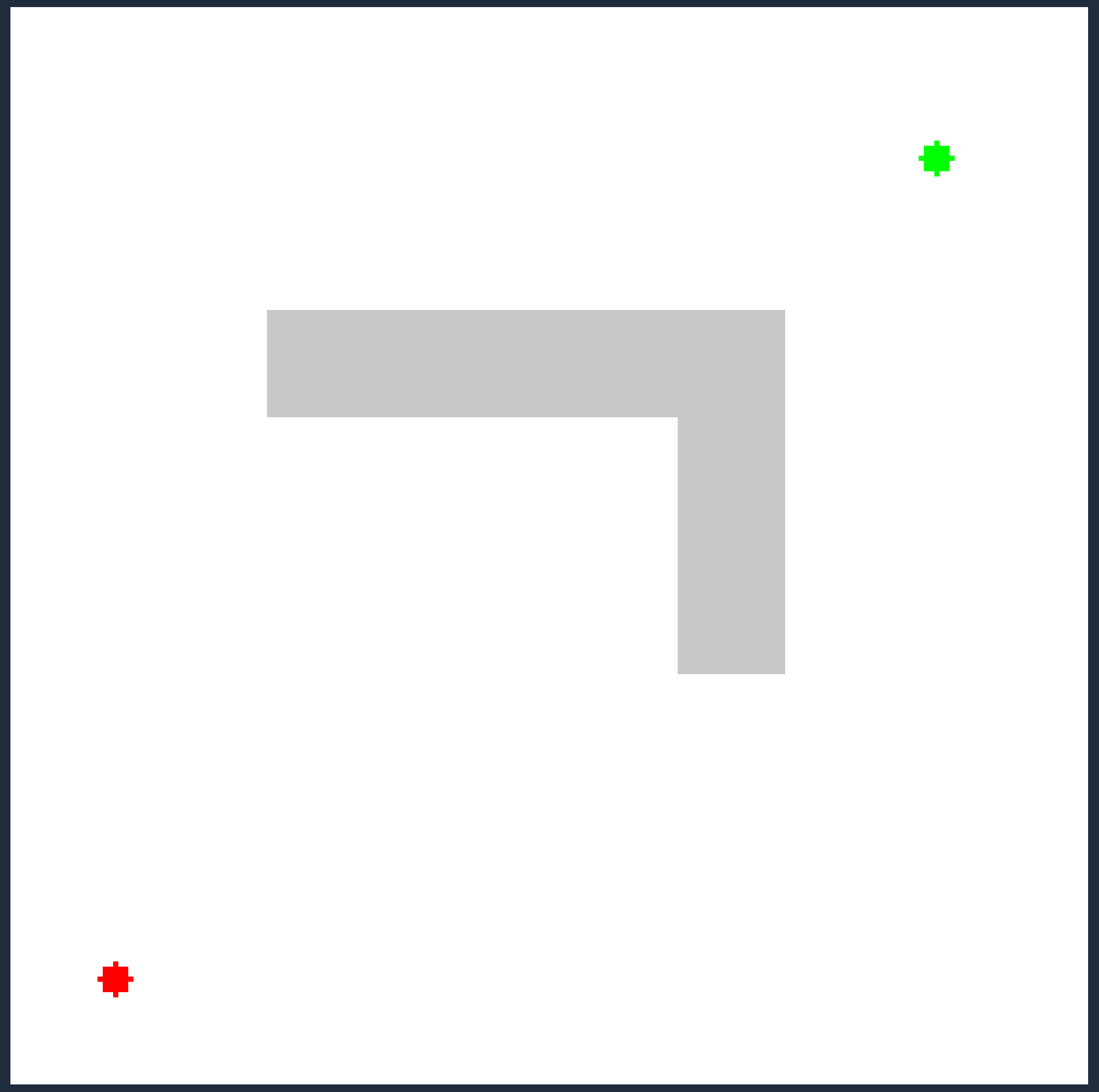
Applications of Edge-Weighted Graphs



Let’s say we want to drive to City 1. We want to find the **Shortest Path** between City 6 and City 1 to save on gas.

What is the Shortest Path in this Graph from City 6 to City 1?

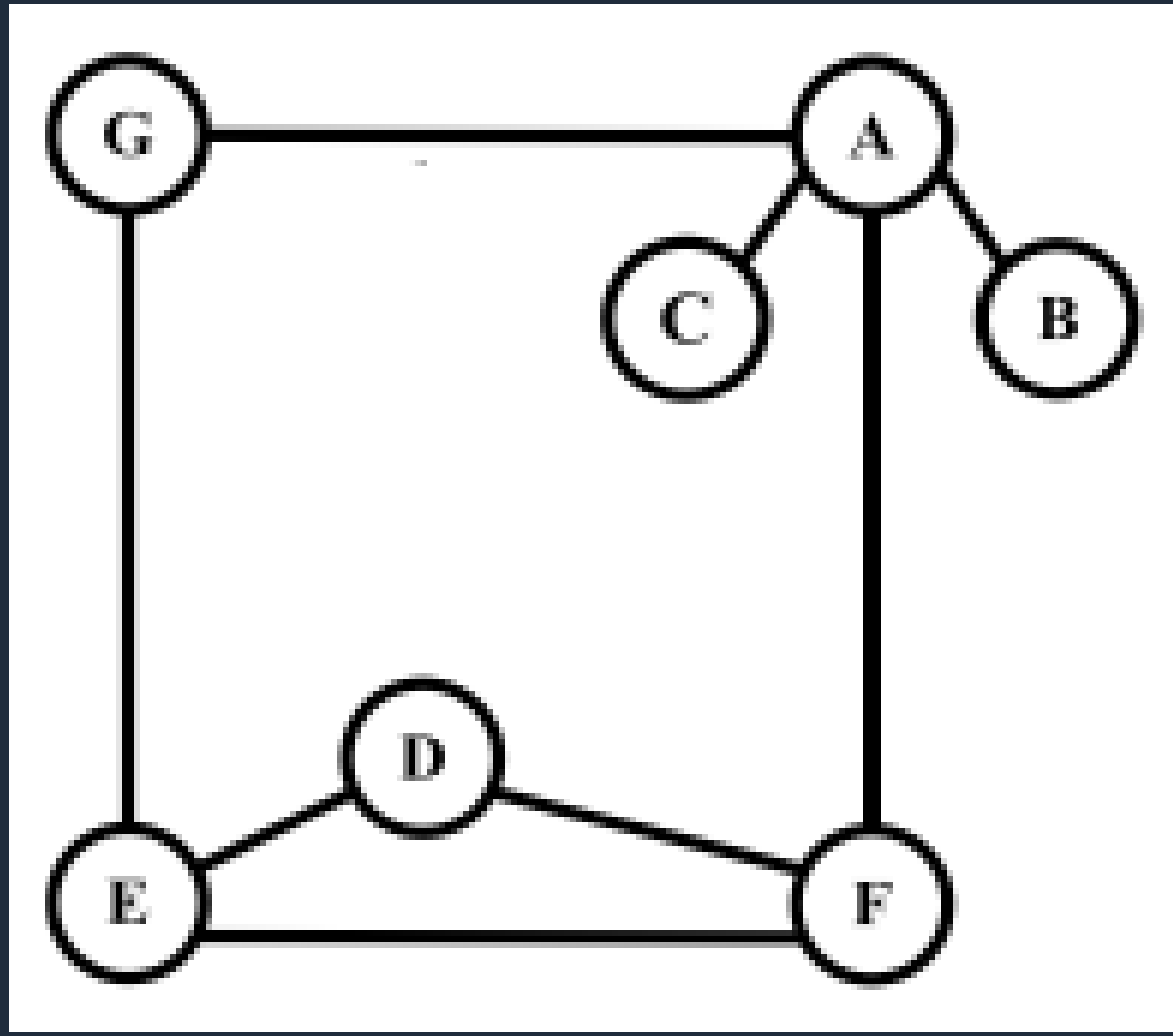
Shortest Path Algorithms



We'll cover more on this later. For now, some examples of shortest path algorithms are:

- Dijkstra's Algorithm
- Bellman-Ford Algorithm
- Floyd-Warshall Algorithm

Graph Cycles

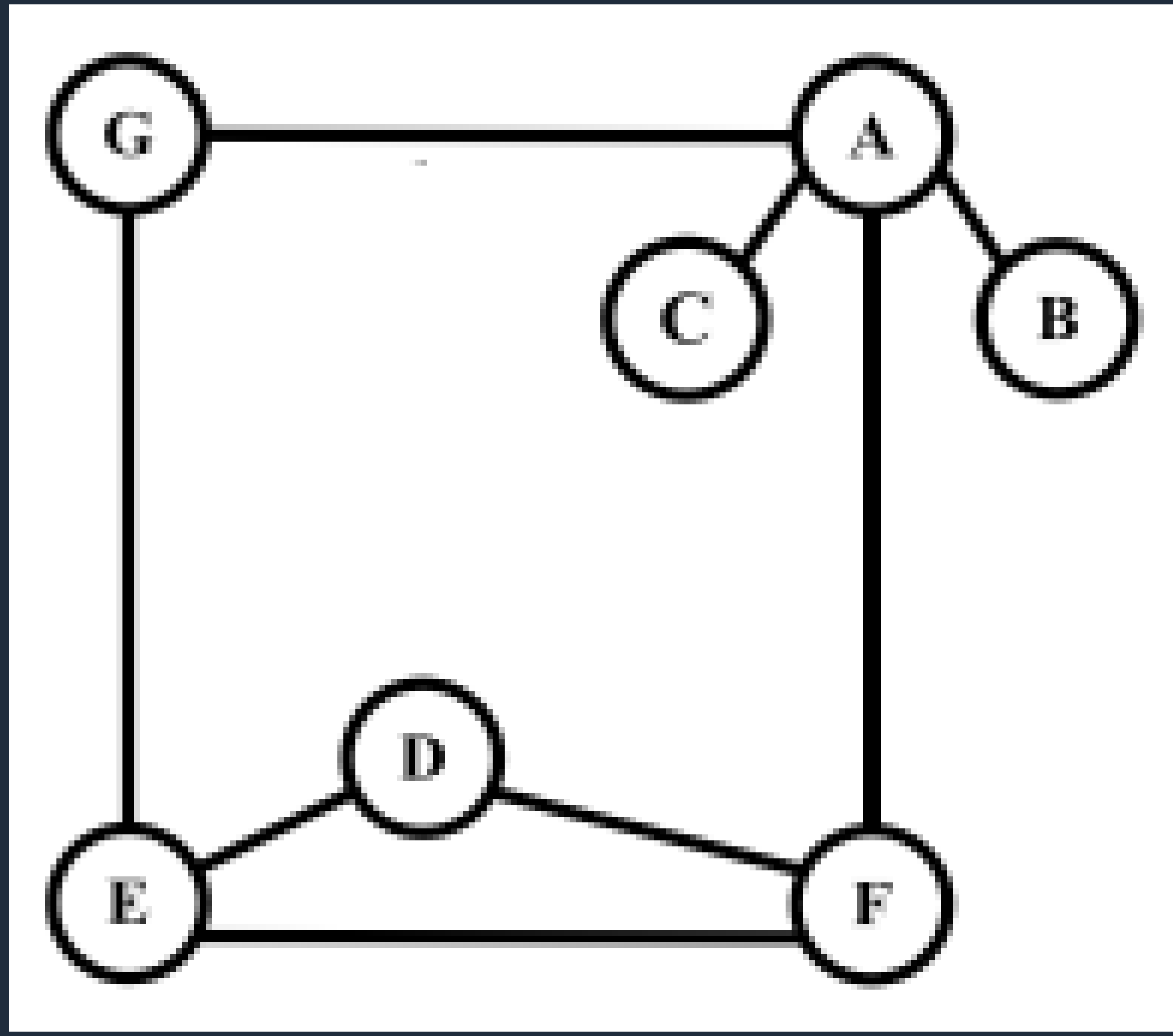


What does it mean for a graph to have a **Cycle**?

We define a Cycle as a path whose first and last vertex is the same. For example, **AFEGA** is a cycle in the graph to the left.

We call a graph with cycles **Cyclic**. Otherwise, it is an **Acyclic** graph.

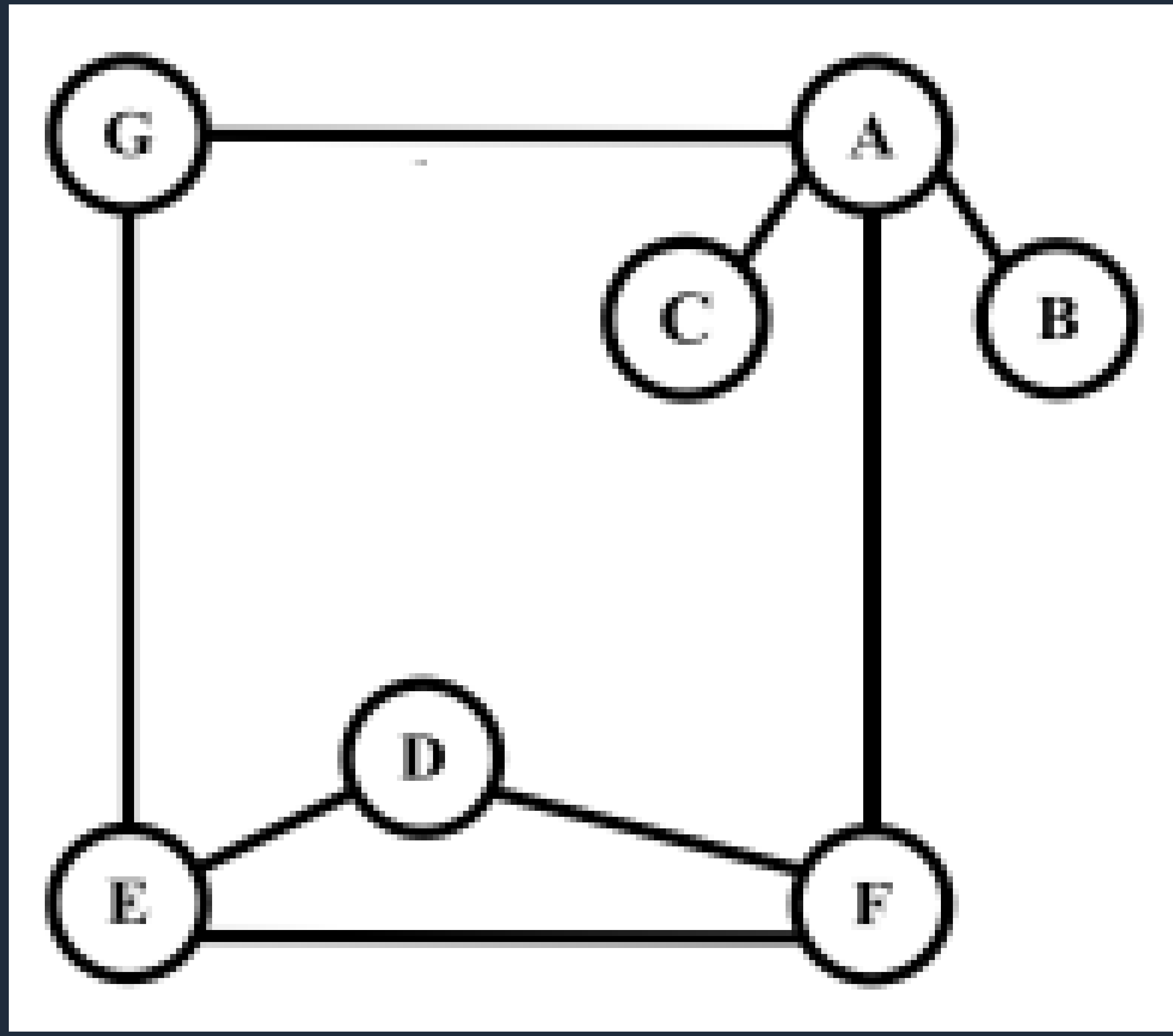
Cycle Equivalency



Are the cycles **AFEGA** and **FEGAF** equivalent?

Are the cycles **AFEGA** and **AGEFA** equivalent?

Cycle Equivalency



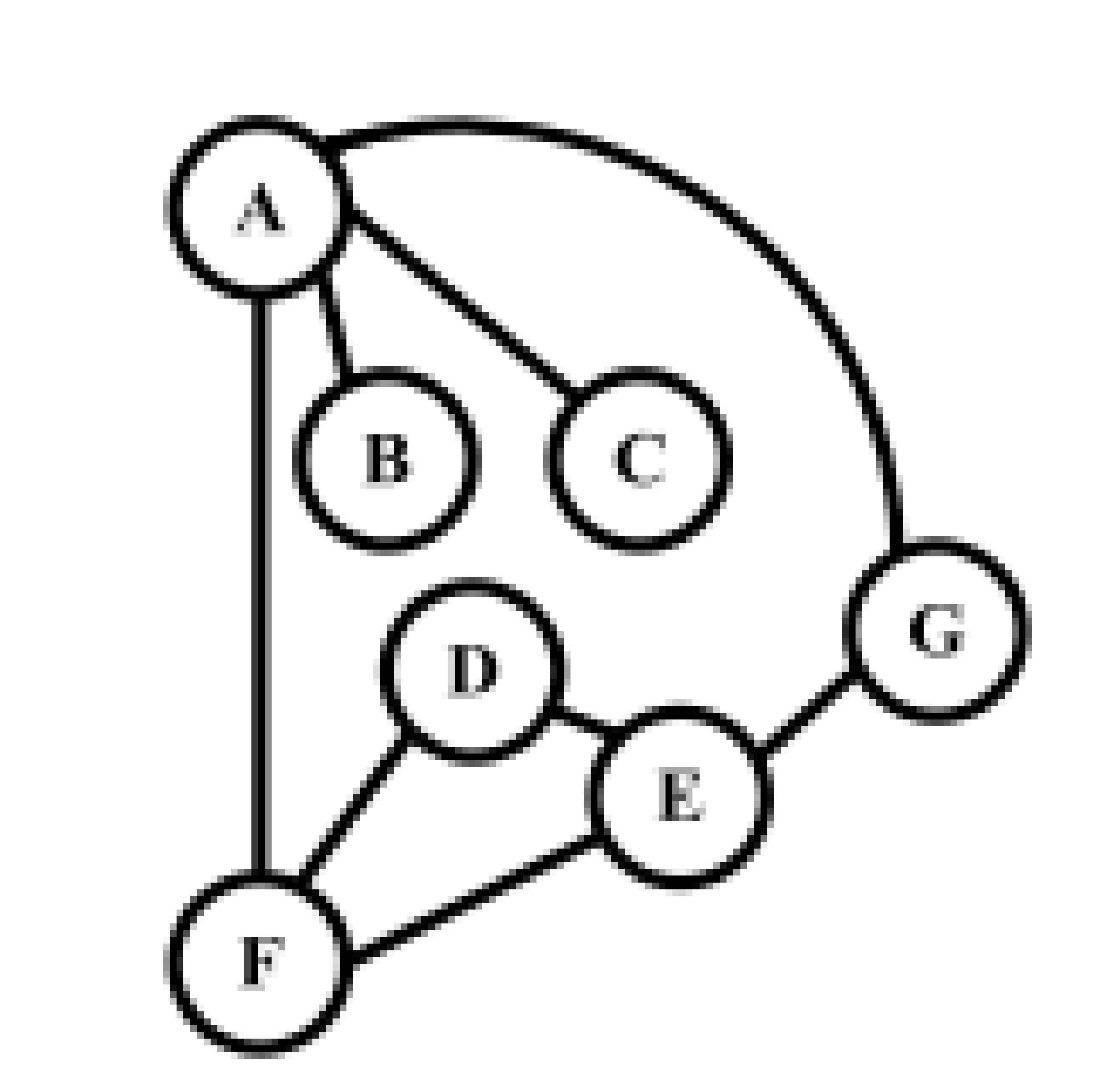
Are the cycles **AFEGA** and **FEGAF** equivalent?

Yes – F must go to E must go to G and so on

Are the cycles **AFEGA** and **AGEFA** equivalent?

No – The order is completely reversed

Simple Cycles and Closed Walks



There are different types of cycles:

A **Simple Cycle** is a cycle where no vertices are repeated except the first and the last

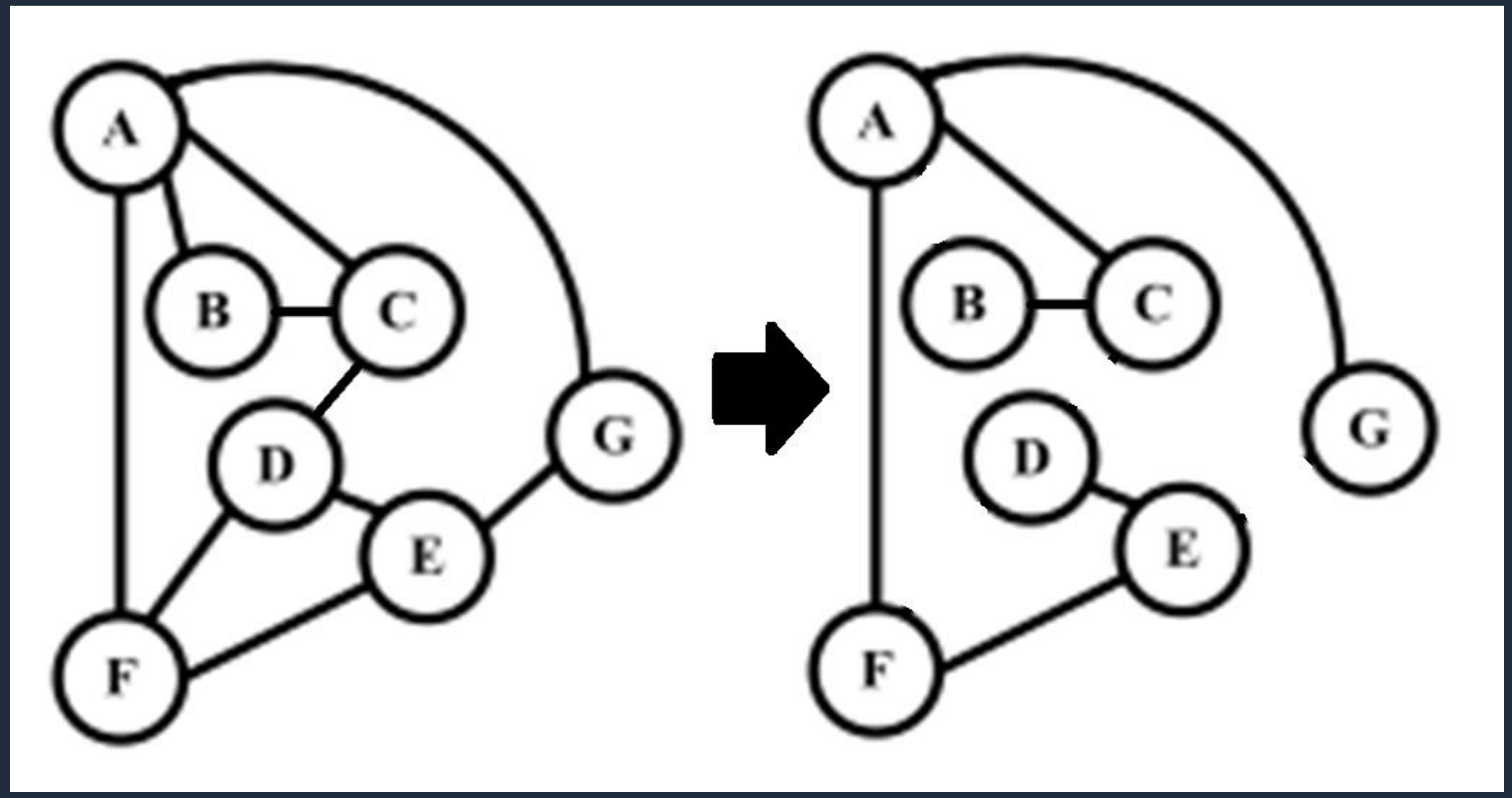
An example would be **AGEFA**

A **Closed Walk** is any sequence starting and ending on the same vertex

An example would be **ACAGEDFA**

TAKE A BREAK

Spanning Trees

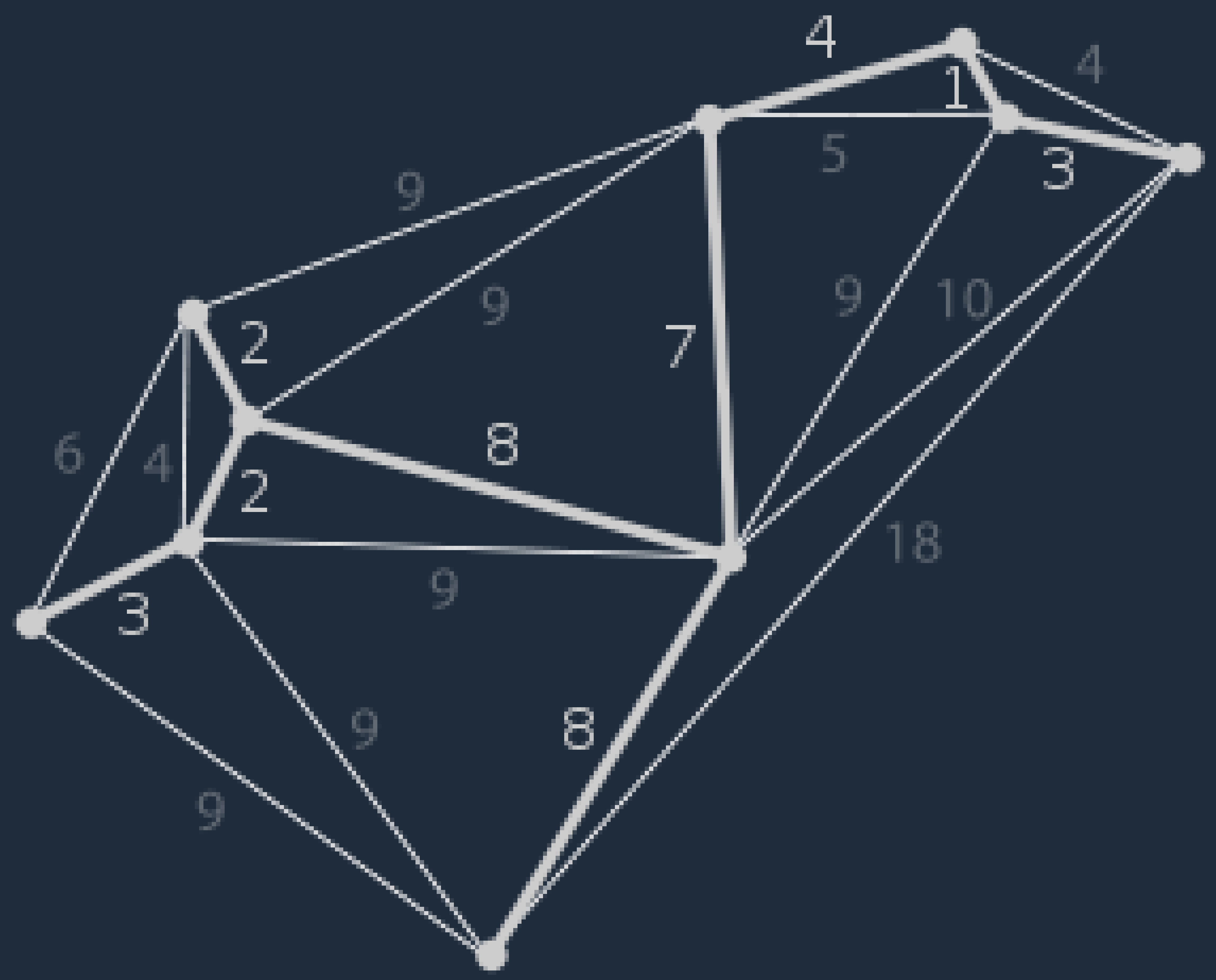


Some graphs look redundant. Why do we need $A \rightarrow B$ and $A \rightarrow C$ when we have already $A \rightarrow B \rightarrow C$?

A **Spanning Tree** is

A graph which contains all of the vertices and a subset of the edges of the original graph and forms a tree, but contains no simple cycles.

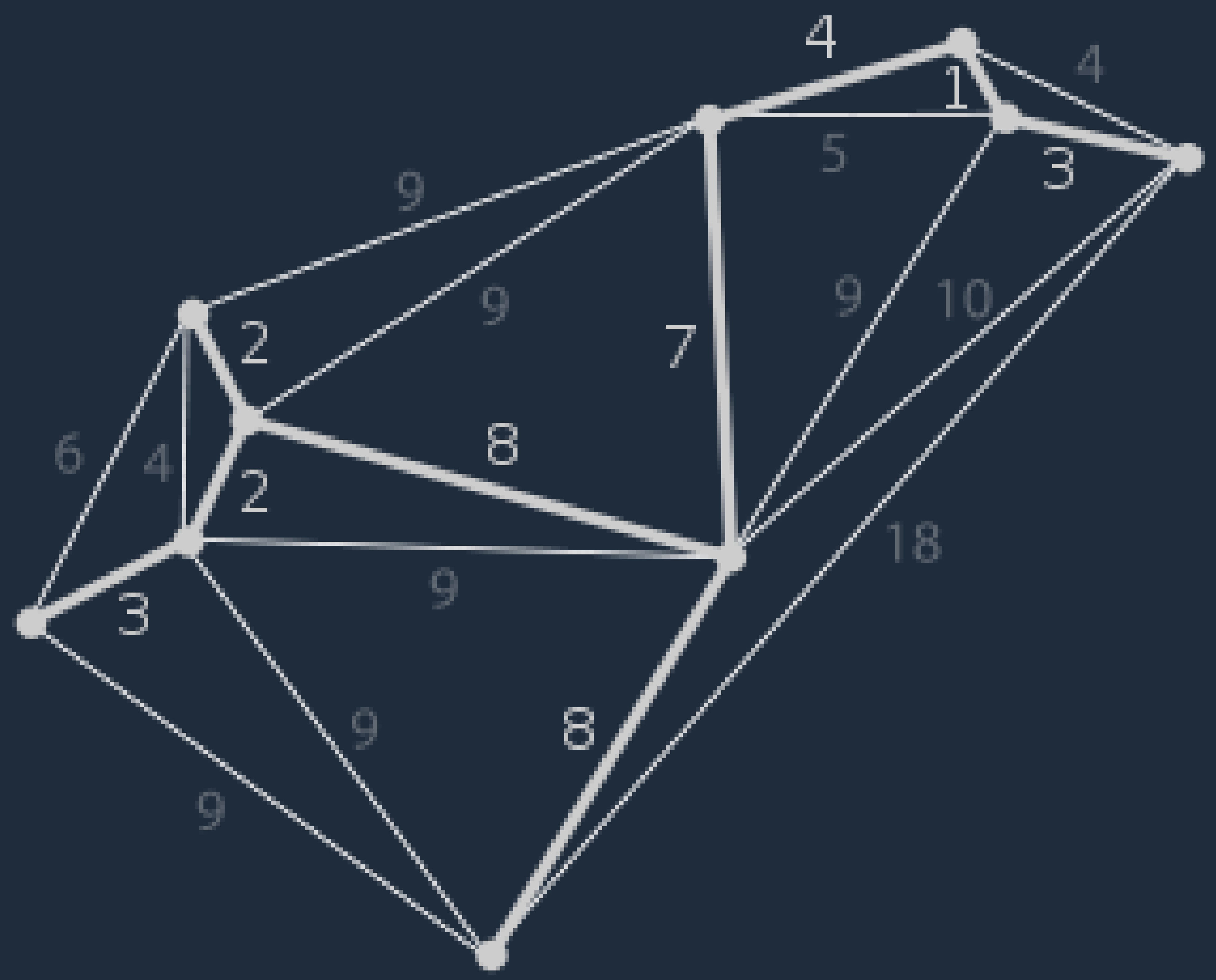
Minimum Spanning Trees



If A → B has weight 6, but A → C → B has weight 4, would path A → B be redundant?

We introduce the concept of a **Minimum Spanning Tree** – minimizing the total edge weight of our new tree while still connecting all the vertices

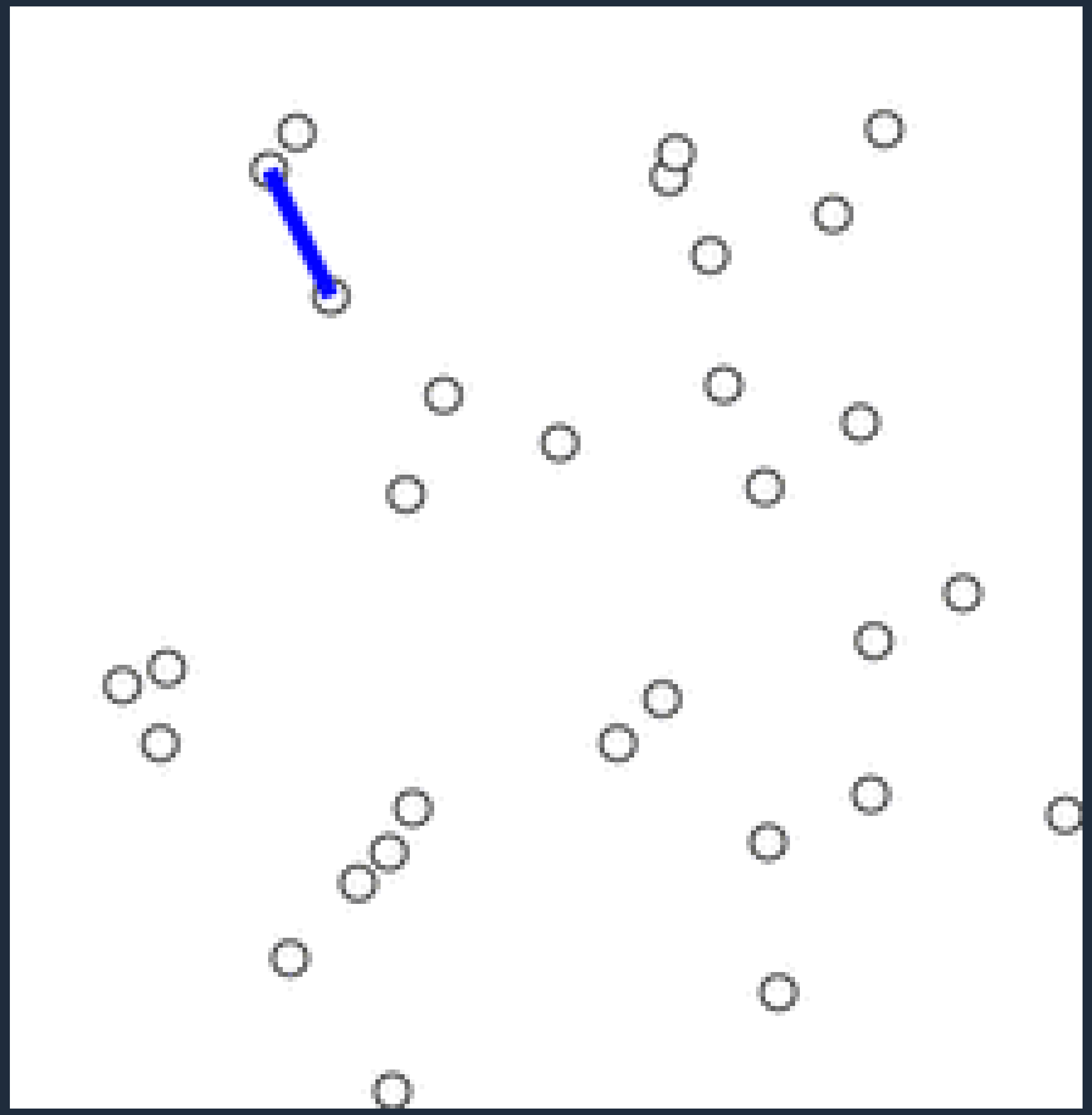
Applications of Minimum Spanning Trees



You have a business with several offices; you want to lease phone lines to connect them up with each other; and the phone company charges different amounts of money to connect different pairs of cities.

We want a set of lines that connects all our offices with minimum cost. We would model the solution using a **Minimum Spanning Tree**

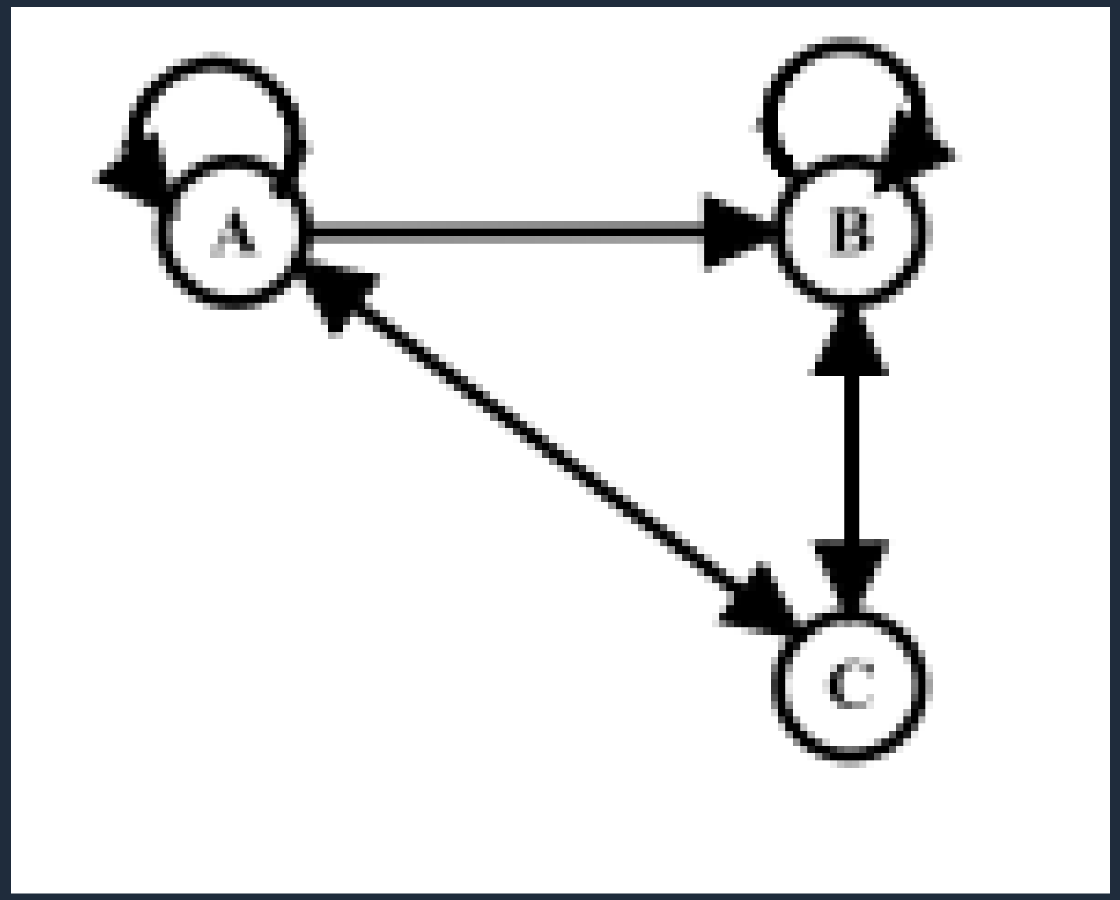
Minimum Spanning Tree Algorithms



We'll cover more on this later. For now, some examples of shortest path algorithms are:

- Prim's Algorithm
- Kruskal's Algorithm

Adjacency Matrices



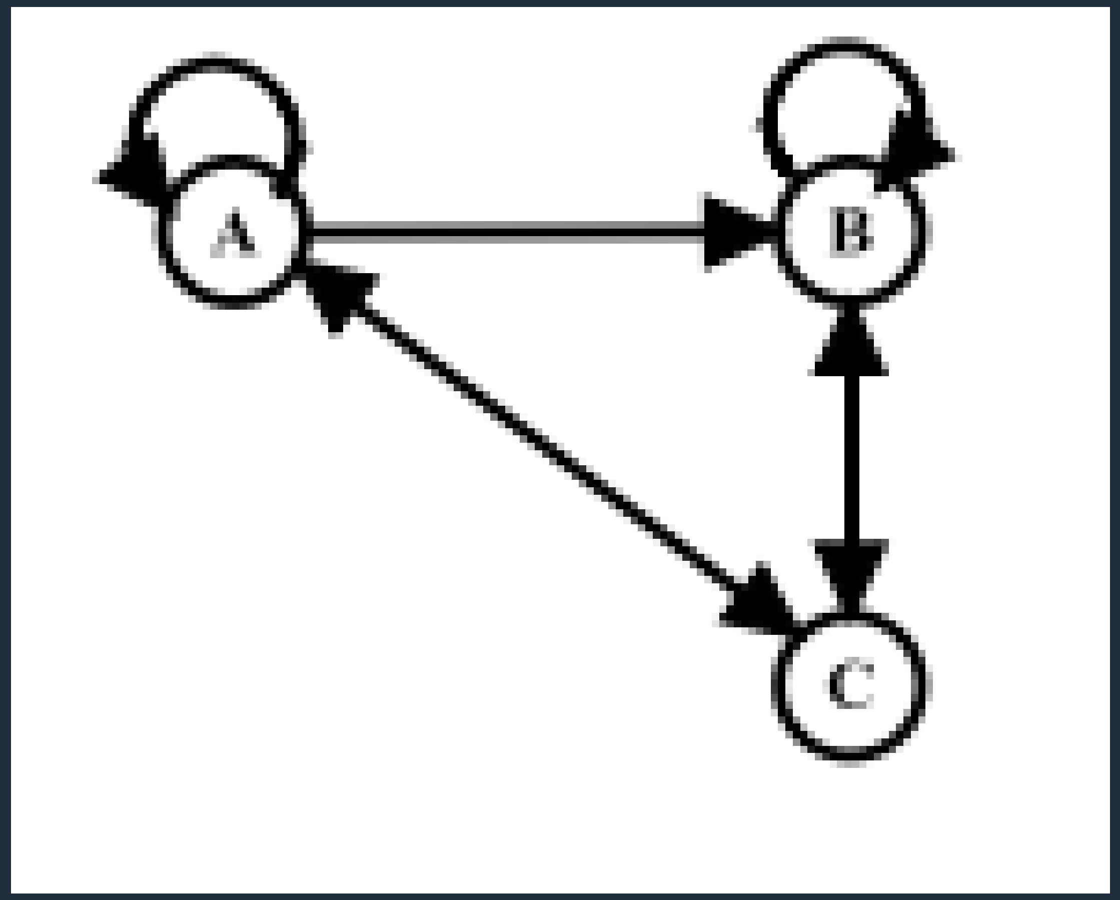
$$M = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \end{bmatrix}$$

Let’s say we want to find all paths of length 4 in our graph. We can arbitrarily count all of the paths (and accidentally repeat a few).

We should instead use an **Adjacency Matrix**

Each row represents the nodes A, B, C …
Each column represents the nodes A, B, C ...

Adjacency Matrices



$$M = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \end{bmatrix}$$

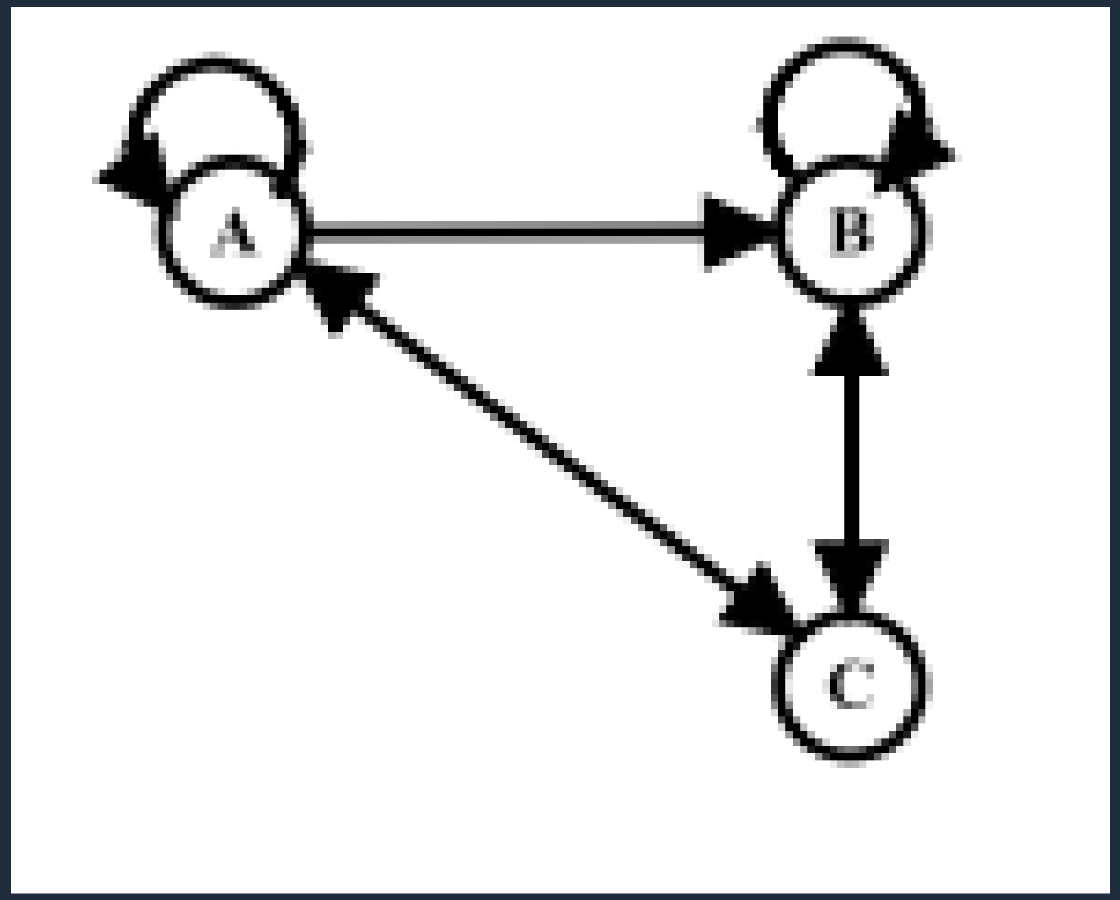
How do we write an initial Adjacency Matrix?

We set up a grid, then populate it with how many length-one paths there are from each row item to each column item (e.g. B cannot go to A, thus $M_{2,1}$ is zero)

	A	B	C
A			
B			
C			

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

Adjacency Matrix Multiplication



$$M = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \end{bmatrix}$$

When we raise the Adjacency Matrix to a power, we get the number of paths of length *n* where *n* is the power we raise it to.

For example, the number of path length two is M^2 which would be the matrix at the right →

$$\begin{pmatrix} 2 & 3 & 2 \\ 1 & 2 & 1 \\ 1 & 2 & 2 \end{pmatrix}$$

We can read this as:

- A → B has 3 paths of length 2
- C → A has 1 path of length 2

APPLYING THE CONCEPTS

Question 1

Given the following, draw out this undirected graph

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

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

Is this graph cyclic? If so, how many cycles from A exist?

Draw this graph but assume it is directed instead. Is this graph cyclic?

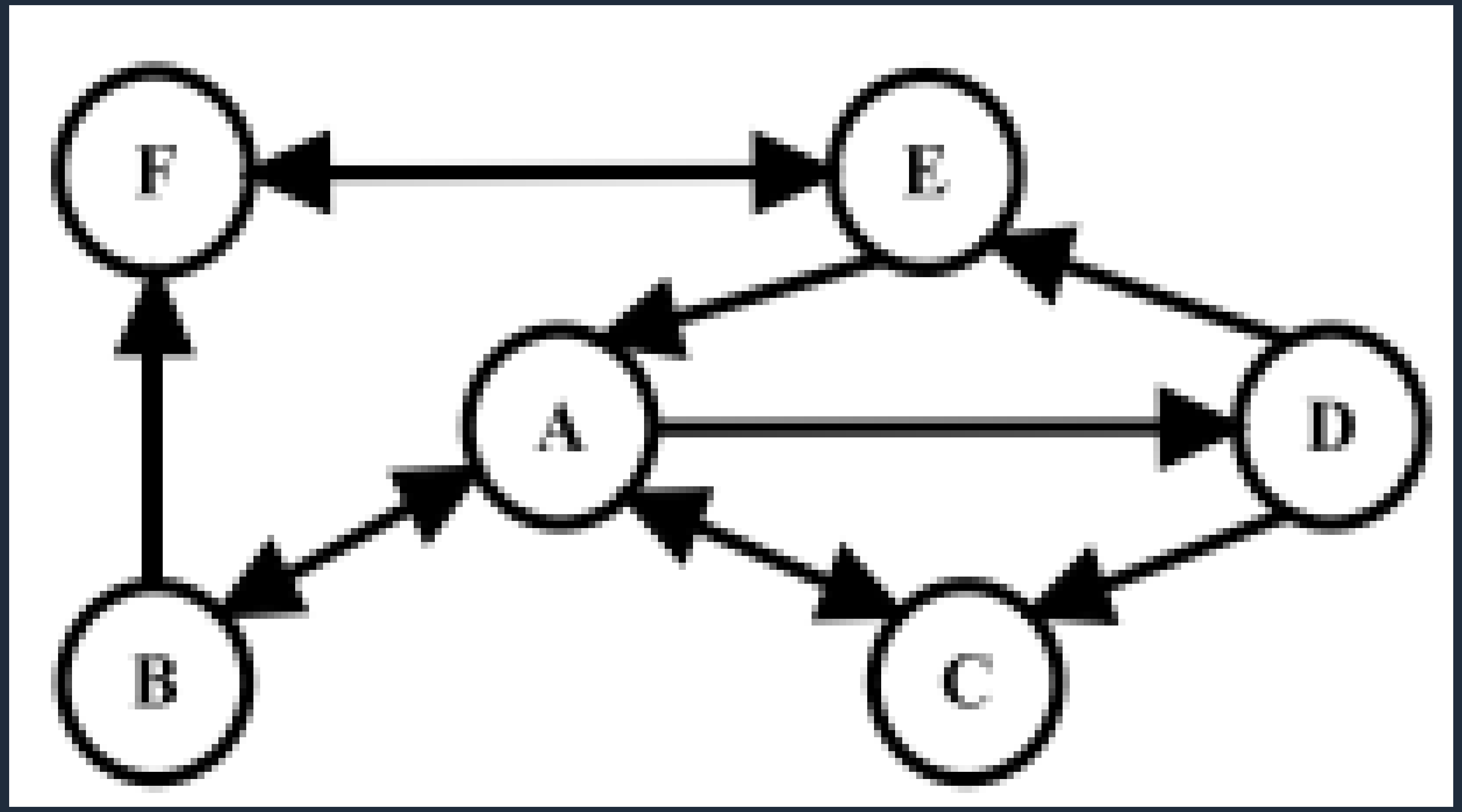
Question 2

Draw out the graph given by this Adjacency Matrix

$$M = \begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix}$$

Question 3

Draw out the Adjacency Matrix given by this graph.



Question 4

Determine the number of paths of length 3 ending at C