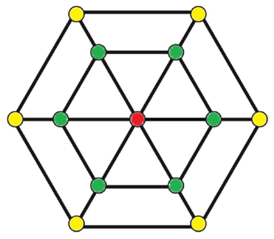
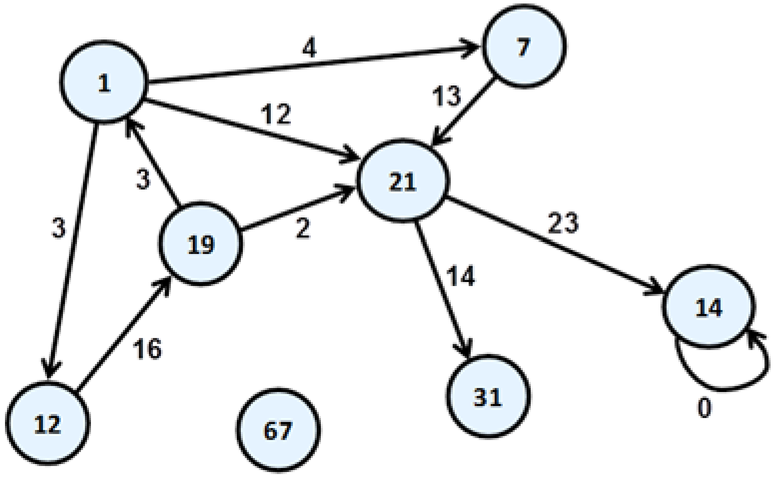
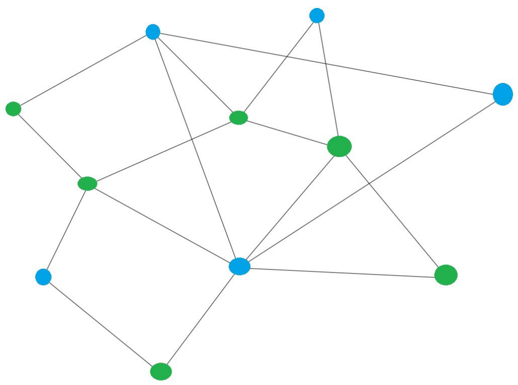
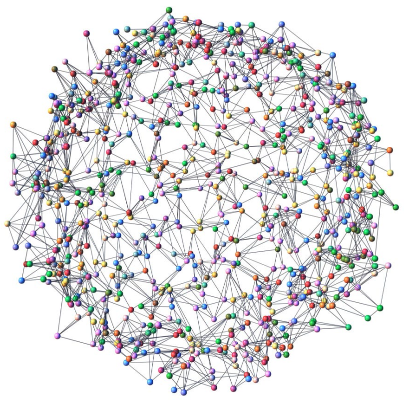


Graphs



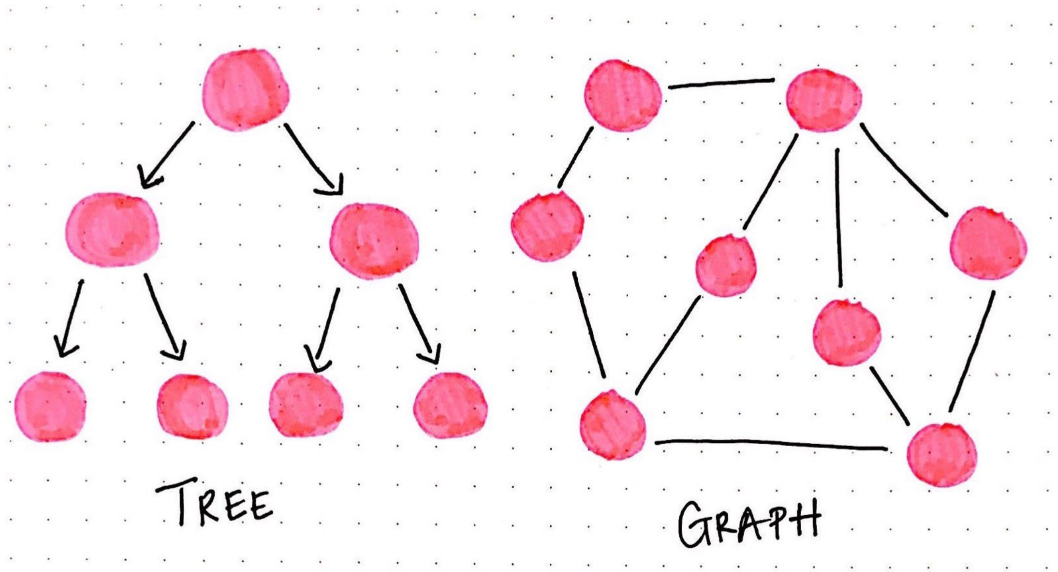




Trees vs Graphs

- Both are composed of vertices/nodes and edges/links.

- Tree: A tree is an undirected graph that is **connected** and has **no cycles**. A tree is a type of graph.
- Graph: Can have cycles (but doesn't have to). Does not have to be connected.



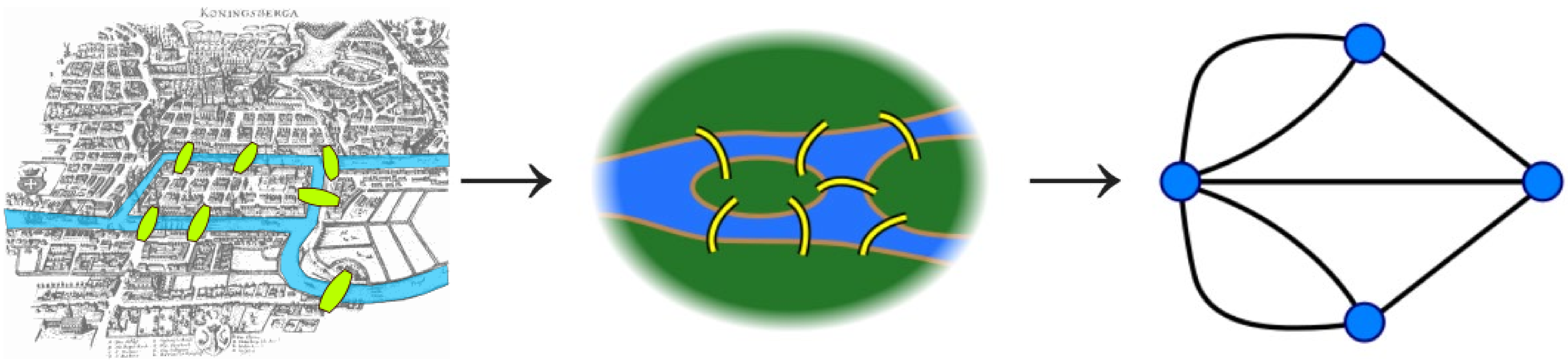
Graphic from: https://medium.com/swlh/making-data-trees-in-python-3a3ceb050cfd

Applications of Graphs

- Used to model relationships
  - Can represent any *relation* (recall reflexive, symmetric, antisymmetric, transitive)

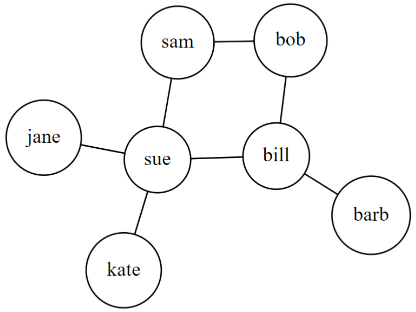
- Solve problems in many disciplines

- Graph theory started when Euler solved the Seven Bridges of Konigsberg problem

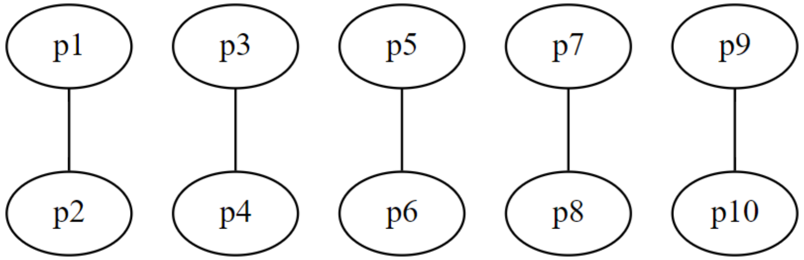


Applications of Graphs

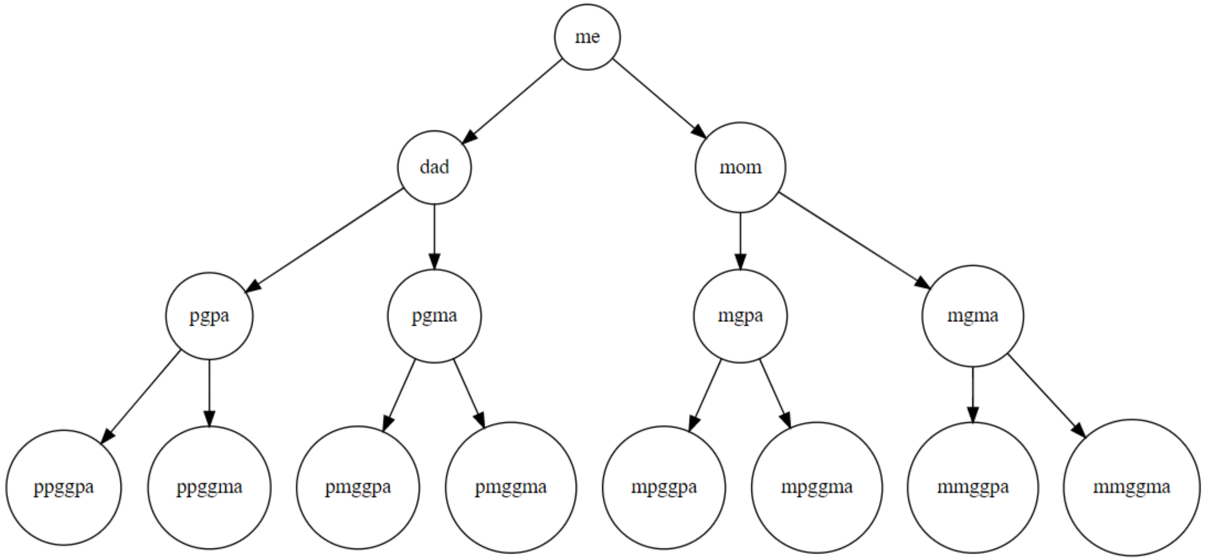
- Relationships between people
  - Friends in a social networking site



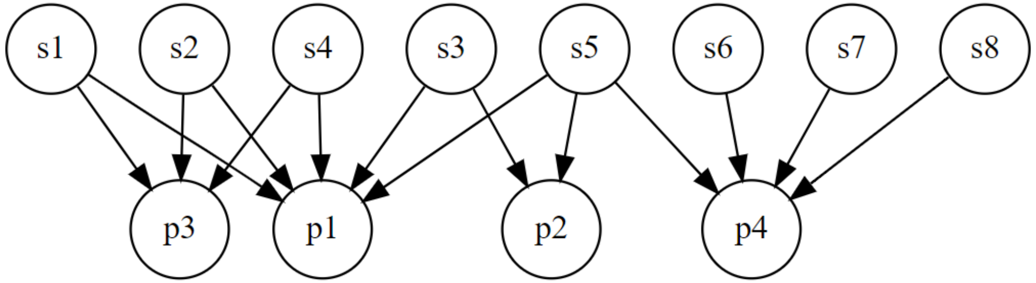
  - Marriages



  - Genealogical relationships



  - Students/Professors



Applications of Graphs

- Scheduling problems (example: How to fit the greatest number of classes into a given schedule)
- Countries that share a border
- What is the fewest number of distinct frequencies needed among 10 radio stations so there is no interference?
- Track winners in a tournament
- Roadmaps
- Can a given electrical circuit be implemented on a planar circuit board?
- Model computer networks
- And on and on…

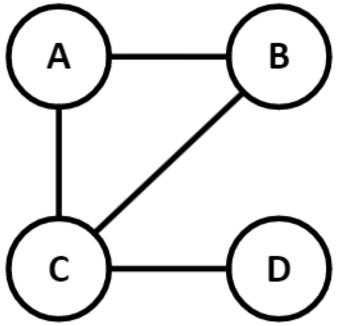
Simple Graph definition:

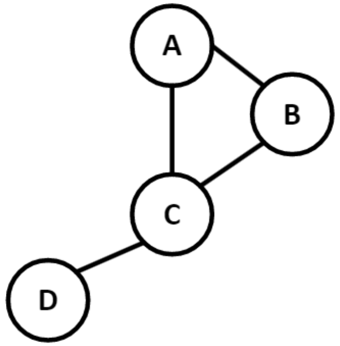
$G = (V, E)$ where V is a nonempty set of vertices and E (the *edges)* is a set of two-element sets (i.e., unordered pairs) of elements of V

It can be helpful to draw it, but this is not necessary to be considered a graph

Example:

$G = \left(\{A, B, C, D\}, \left\{\{A, B\}, \{A, C\}, \{B, C\}, \{C, D\}\right\}\right)$





A graph G is an ordered pair consisting of a nonempty set of *Vertices (Nodes)* and a set of *Edges (Links)*

These both represent the same graph

Simple Graphs:

- No self-loops. Only one edge between any two vertices.
- Model symmetric relations
- Edges are undirected (i.e., edges represent a symmetric relation)
- Any pair of objects can be related or not related
- We can use a simple graph to model anything with symmetric relations

Recall symmetric relations:

$\forall a \forall b\ (aRb \rightarrow bRa)$

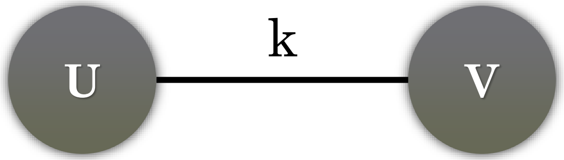
Examples:

Countries that share a border or do not share a border
Land masses connected by a bridge
Cities connected by a road
Family relationships (marriage, sibling, cousin, etc)
People that speak the same language or don't speak the same language
Classes that are taught at the same time or not taught at the same time
...

Some Terms

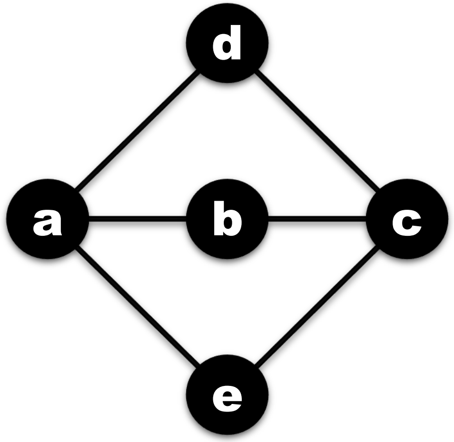
Let G be an undirected graph and let k be an edge of G that is {u, v}.

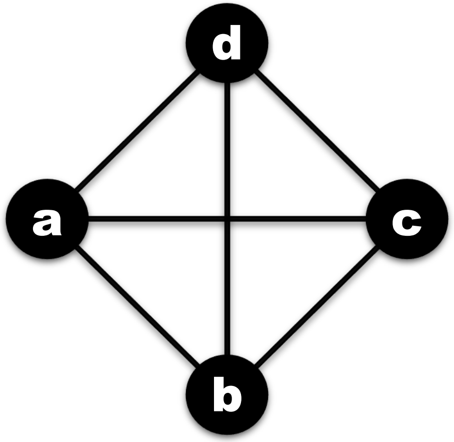
u and v are **adjacent**
u and v are **neighbors**
u and v are **connected**
Edge k is **incident** with nodes u and v
Edge k **connects** u and v
Nodes u and v are the **endpoints** of edge k

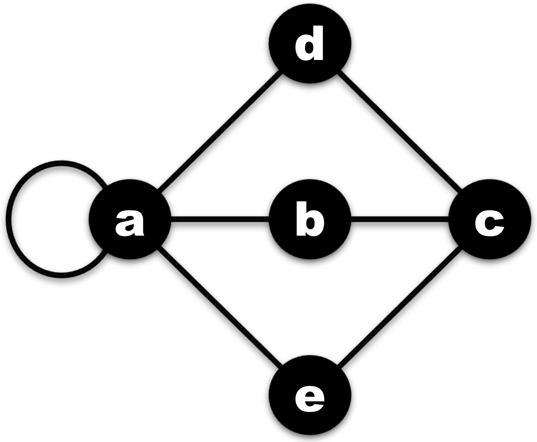


More terms

- Directed vs undirected
- Node/vertex
- Link/edge
- Degree of a vertex
- Total degree
- Loop
- Regular graph
- D-regular graph





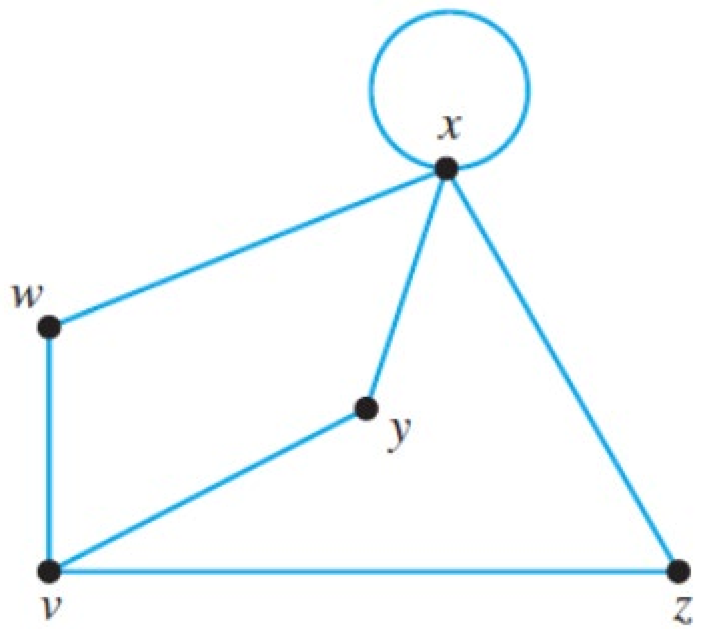


Degree Sequence

A sequence of the degrees of every node in a graph, listed in non-increasing order (highest to lowest)

What is the degree sequence of this graph?

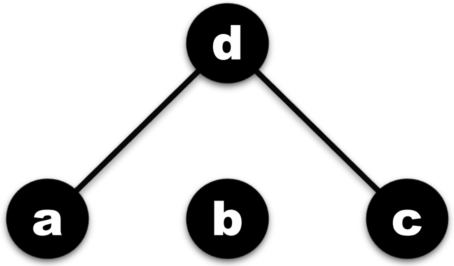
[5,3,2,2,2]



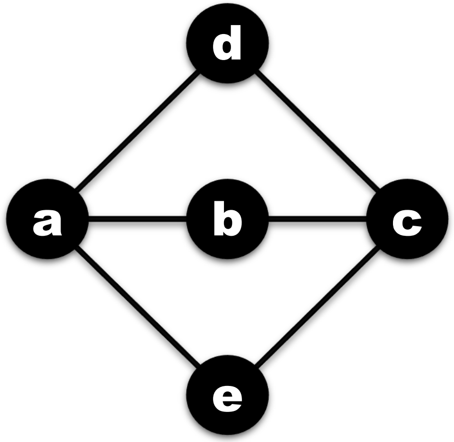
Subgraphs

A graph $H = (V_H, E_H)$ is a subgraph of graph $G = (V_G, E_G)$ if $V_H \subseteq V_G$ and $E_H \subseteq E_G$.

Is H a subset of G?



H

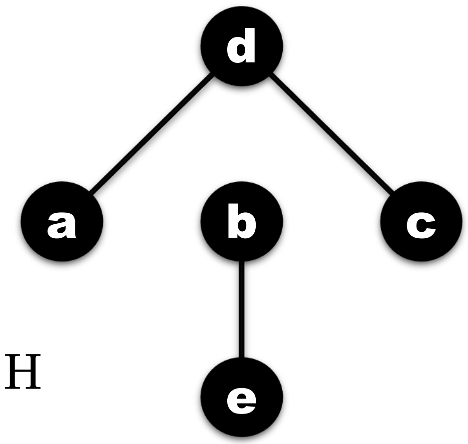


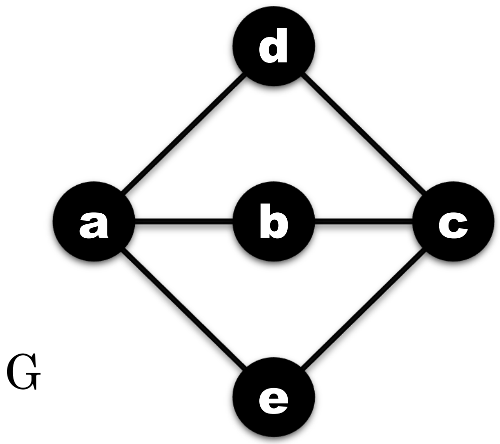
G

Subgraphs

A graph $H = (V_H, E_H)$ is a subgraph of graph $G = (V_G, E_G)$ if $V_H \subseteq V_G$ and $E_H \subseteq E_G$.

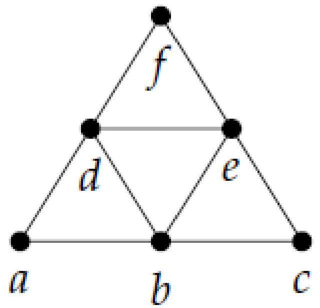
What about now?



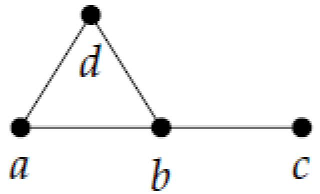


Subgraphs

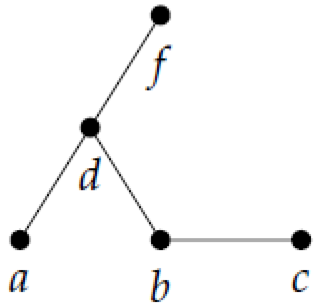
Given a graph $G = (V, E)$, a subgraph of G is any graph $H = (O, P)$ where $O \subseteq V$ and $\mathrm{P} \subseteq E$.



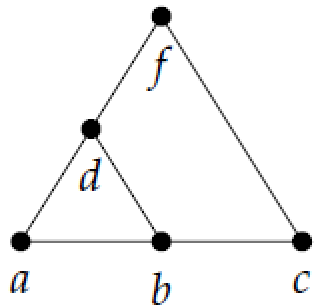
G_1



G_2



G_3



G_4

Which of these are subgraphs of G_1?

G_2 and G_3 are both subgraphs of G_1.
G_4 is NOT a subgraph of G_1 because the link $\{c, f\}$ is not a link in G_1

Levin, Oscar. *Discrete Mathematics, An Open Introduction, 3rd Edition*. 2021

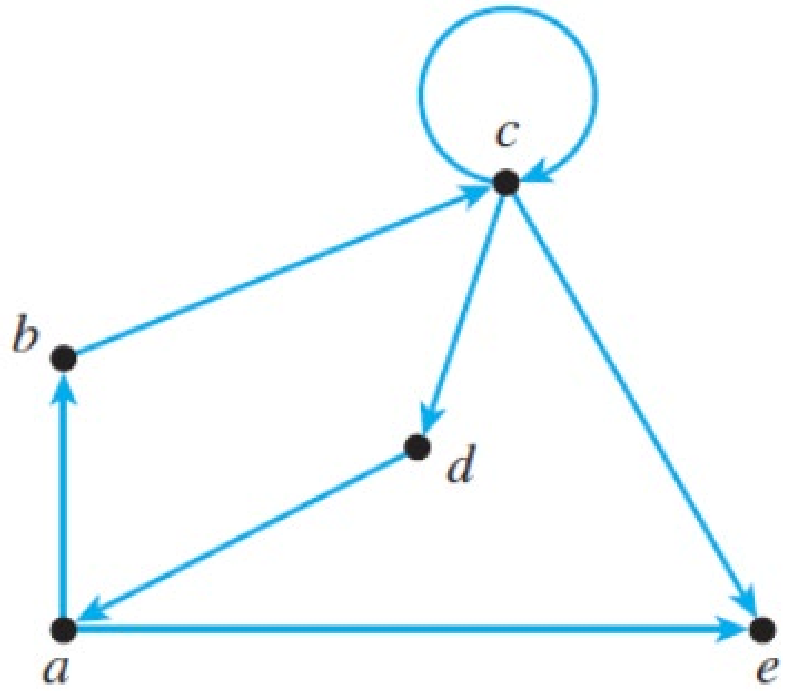
Directed Graphs (Digraphs)

- Arbitrary binary relations (not necessarily symmetric)
- Edges have direction. In other words, edges are **ordered pairs**.

$G = (V, E)$

$V = \{a, b, c, d, e\}$

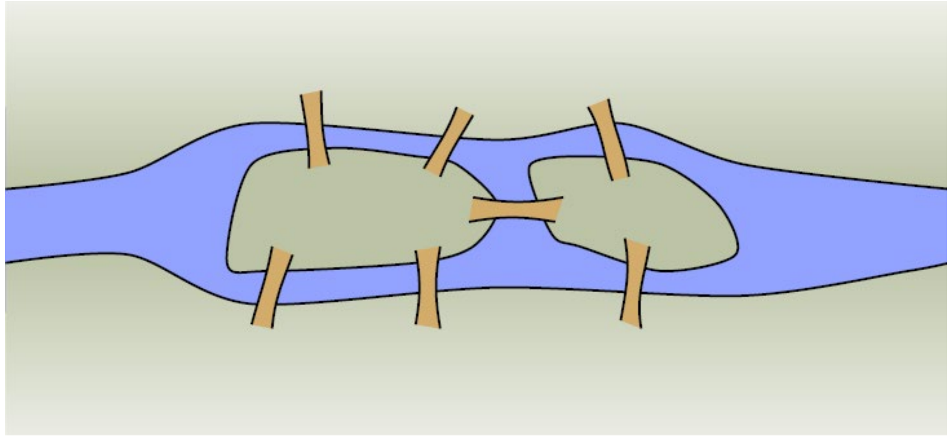
$E = \{(a, b), (a, e), (b, c), (c, c), (c, d), (c, e), (d, a)\}$

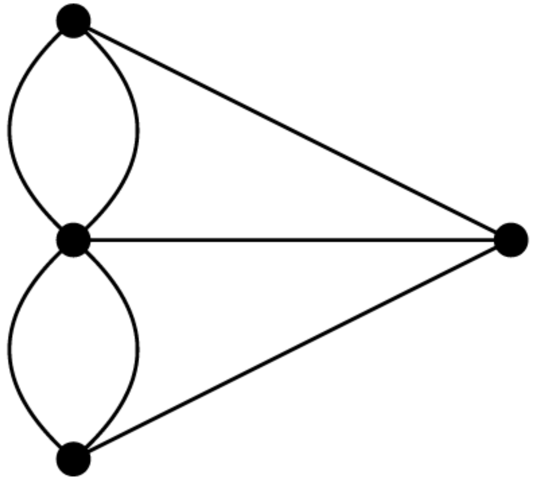


Hunter, David J. *Essentials of Discrete Mathematics, 3rd Edition*. Jones & Bartlett Learning 2017

Multigraph

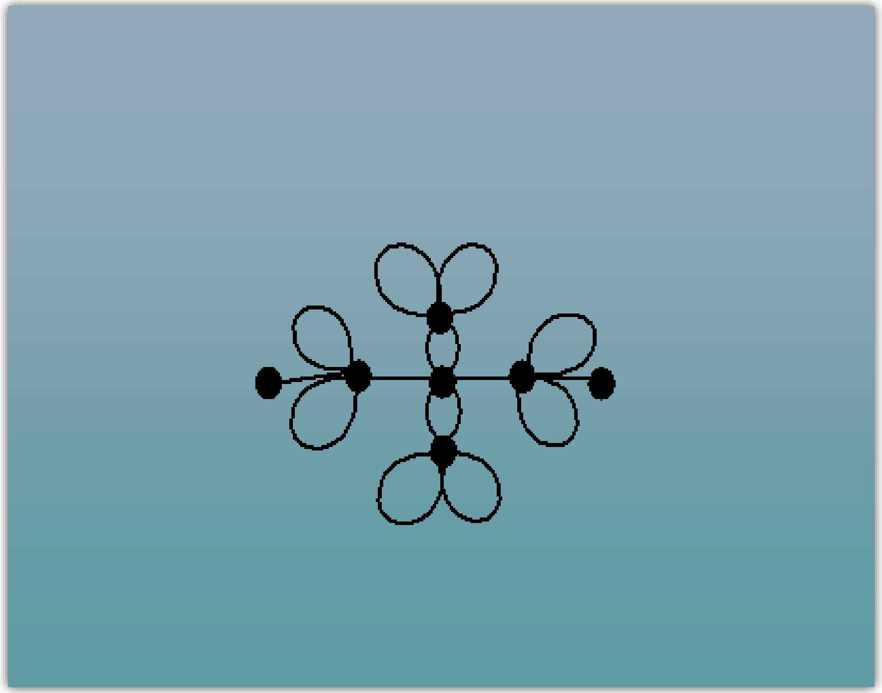
- There may be more than one link between a pair of adjacent nodes

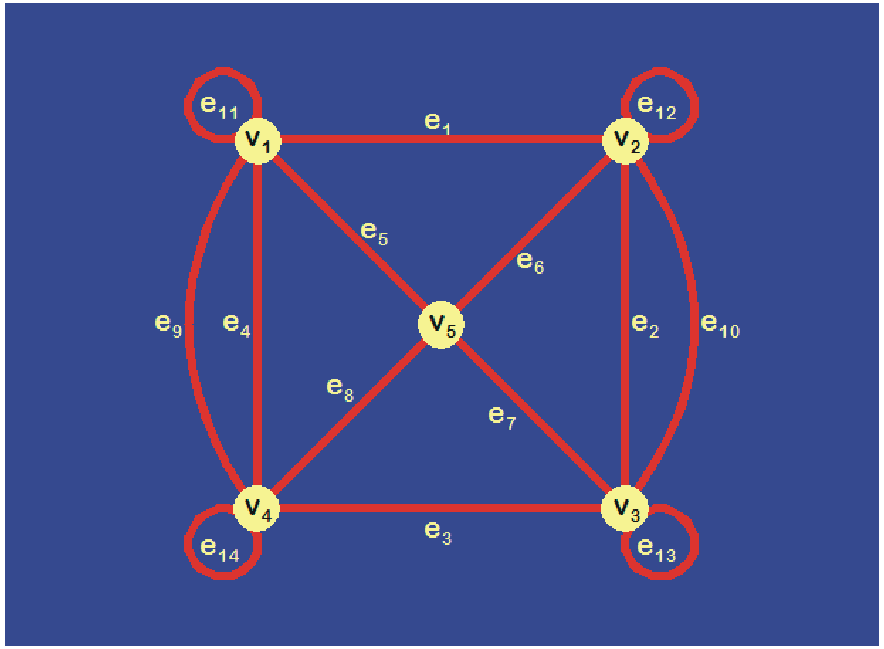




Pseudograph

- One or more nodes is adjacent to itself (loops)





Complete Graph

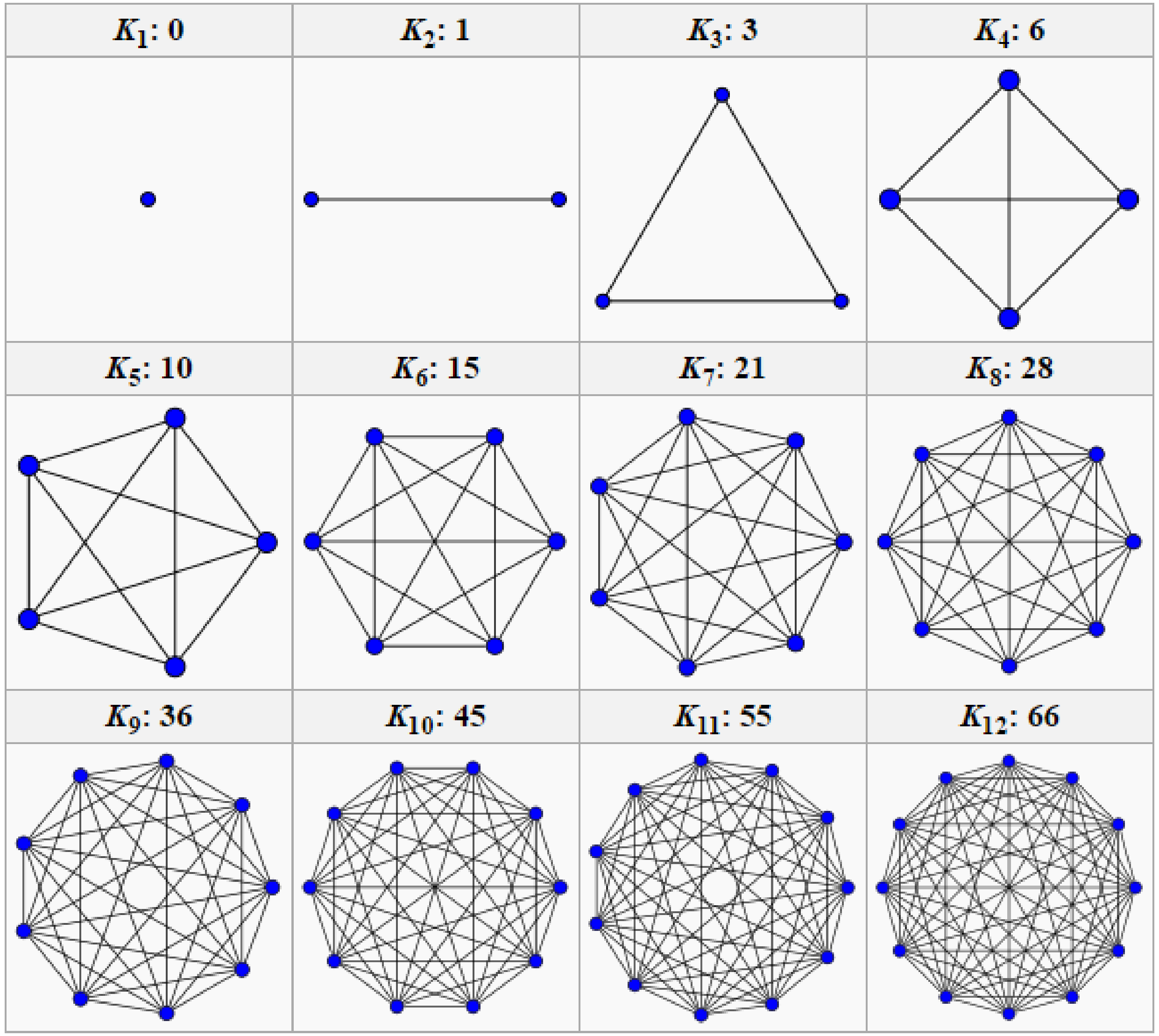
Every pair of vertices is connected by an edge.

All possible edges are in the graph.

Every node in the graph is adjacent to every other node in the graph.

Denoted by K_n

Also called a **clique** of size n.



Handshaking Theorem

If we know the degrees of all the vertices in an undirected graph, we can find the number of edges.

Let $G = (V, E)$ be an undirected graph. Then,

$$\sum_{v \in V} \deg(v) = 2 \cdot |E|$$

The sum of the degrees of each vertex is twice the number of edges

Think about it this way:

Each edge must have two vertices. Therefore, the sum of the degrees of the vertices is twice the number of edges.

Handshaking Theorem

$$\sum_{v \in V} \deg(v) = 2 \cdot |E|$$

Example:

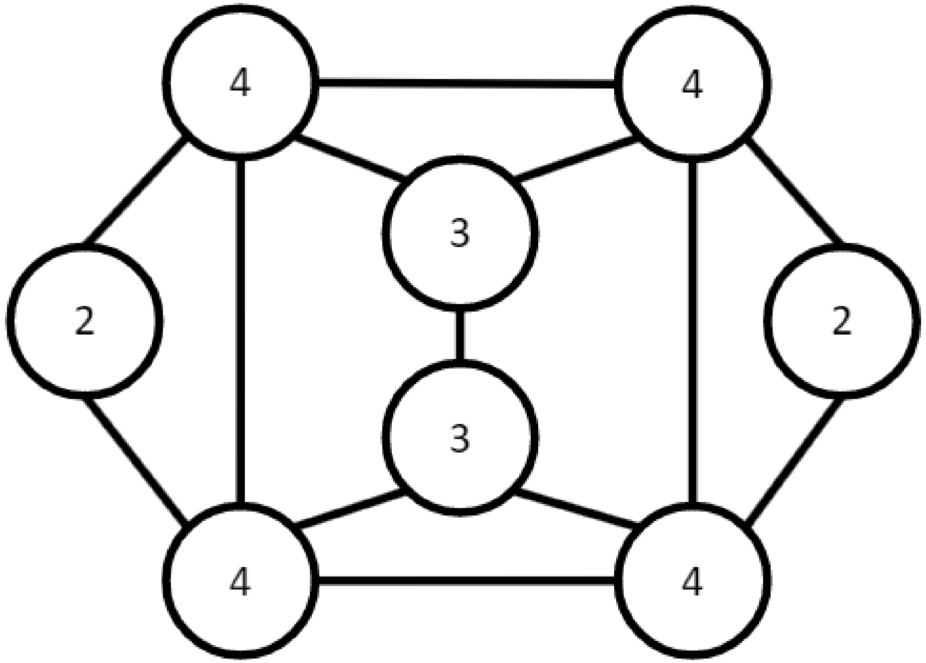
How many edges are in the graph with the following degree sequence?

[4,4,4,4,3,3,2,2]

$4 + 4 + 4 + 4 + 3 + 3 + 2 + 2 = 26$

$\frac{26}{2} = \mathbf{13}$ edges

Count them. Is it true?



Handshaking Theorem

$$\sum_{v \in V} \deg(v) = 2 \cdot |E|$$

Example:

Is it possible for 7 people in a room to each shake hands with exactly 5 other people?

Turn it into a graph problem.

There are 7 vertices representing the 7 people. An edge between vertices represents a handshake. If each person shakes hands with exactly 5 other people, then each vertex has a degree of 5.

The sum of the degrees is

$5 + 5 + 5 + 5 + 5 + 5 + 5 = 7 * 5 = \ 35$

That would mean there are $\frac{35}{2} = 17.5$ edges. That is not possible.

Handshaking Theorem

$$\sum_{v \in V} \deg(v) = 2 \cdot |E|$$

Try it:

Among a group of 5 people, is it possible for everyone to be friends with exactly 2 of the people in the group?

Yes. Using a graph of 5 vertices, there are 2 edges per vertex.
There are $\frac{2*5}{2} = 5$ edges. It is possible.

What about with exactly 3 of the people in the group?

No. There are $\frac{5*3}{2} = 7.5$ edges. This is not possible.

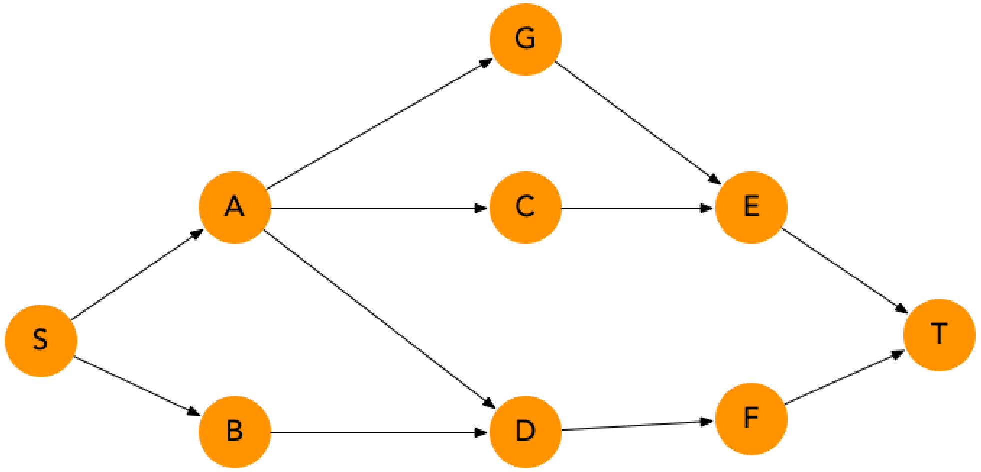
Additional Exercises

13.1.1

13.1.3

Adjacency List

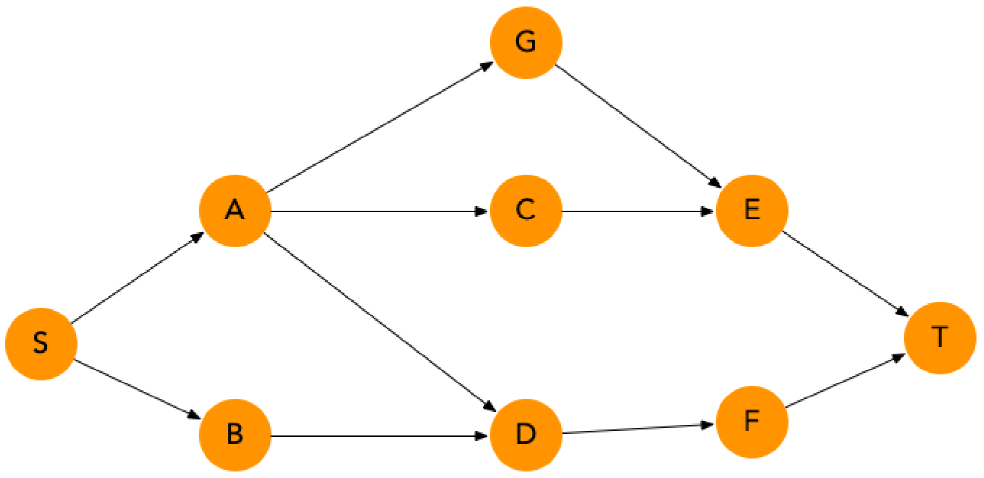
Represents the relationship between vertices



s	a, b
a	g,c,d
b	d
g	e
c	e
d	f
e	t
f	t
t	

Adjacency Matrix

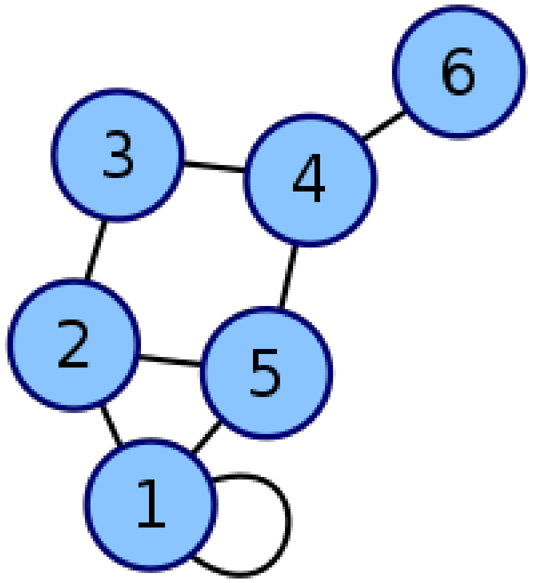
Represent the relationship between nodes



	S	A	B	C	D	E	F	G	T
S		1	1						
A				1	1			1	
B					1				
C						1			
D							1		
E									1
F									1
G						1			
T									

Try It

Create an adjacency matrix for this graph



	1	2	3	4	5	6
1	**1**	**1**			**1**	
2	**1**		**1**		**1**	
3		**1**		**1**		
4			**1**		**1**	**1**
5	**1**	**1**		**1**		
6				**1**		

What do you notice about the symmetry of this undirected graph?

Additional Exercises

13.2.1

13.2.2

13.2.3 (if time)

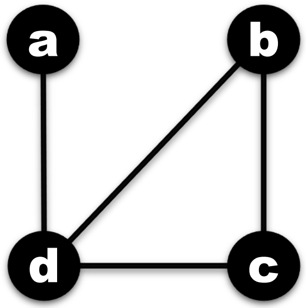
Graph Isomorphisms

Two graphs G_1 and G_2 are isomorphic if we can find a way to map the vertex sets of each graph such that there is an edge between two vertices in G_1 **if and only if** there is an edge between the corresponding two vertices in G_2.

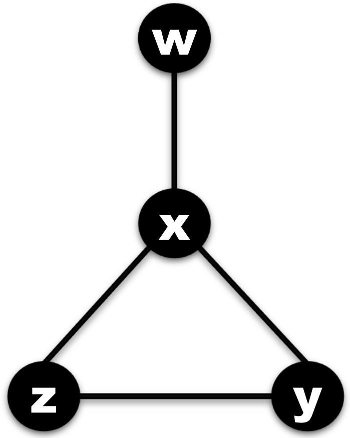
In other words, we need to find a bijection function $f: \mathrm{V}_1 \to \mathrm{V}_2$ such that for every $x, y \in V_1$, then $\{x, y\} \in E_1$ if and only if $\{f(x), f(y)\} \in E_2$.

$f: V_1 \to V_2 = \{(a, w), (d, x), (c, z), (b, y)\}$

G_1



G_2



f is an **isomorphism** from G_1 to G_2.

Another way to think about it: Two graphs are isomorphic if the vertices can be relabeled so that the graphs are identical.

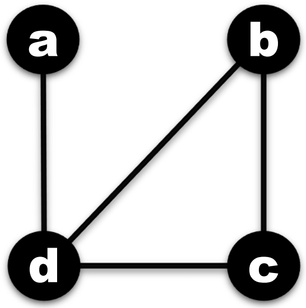
Graph Isomorphisms

Some graph properties are preserved under isomorphism.

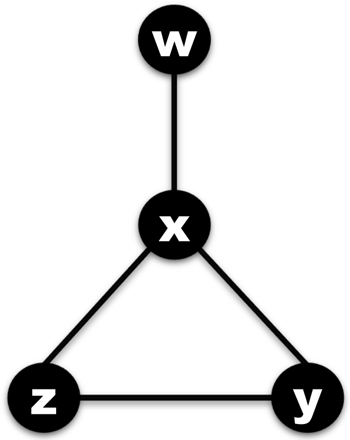
If two graphs are isomorphic, then both have the same:

- Vertex degree
- Degree sequence
- Anything related to the structure of the graph rather than the labels on the vertices/edges.

G_1



G_2



Additional Exercises:

13.3.1

13.3.2

Walks

Graphs are made up of:

Vertices: ID, MT, WY, etc.

Edges: (ID,MT), (NV,ID), etc.

We represent a **walk, trail,** or **path** as a sequence of edges or as a sequence of vertices.

A **trail** is a sequence in which no **edge** occurs more than once.

A **circuit** is a closed sequence in which no edge occurs more than once.

A **path** is a trail that does not repeat **vertices** (which also means it does not repeat **edges**).

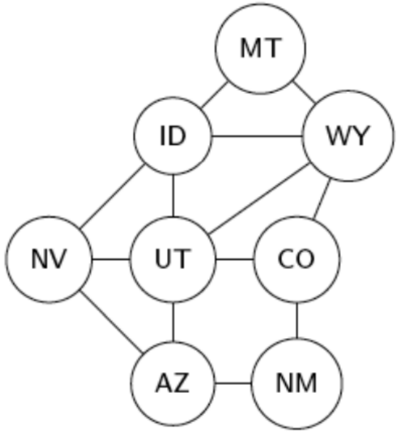
A **cycle** is a circuit in which no vertices are repeated, except the starting/ending vertex

Example:

A path from NV to WY:

As a sequence of Vertices: [NV, UT, WY]

As a sequence of Edges: [(NV,UT), (UT,WY)]

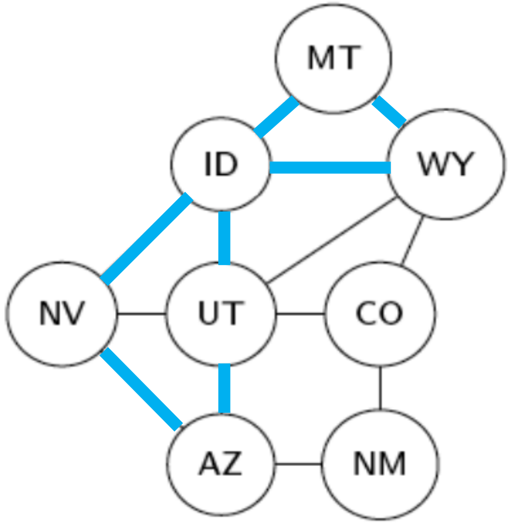


Circuit vs Cycle

A **circuit** is a **trail** from a vertex back to itself, with no repeated **edges.**

[(NV, ID), (ID, MT), (MT, WY), (WY, ID), (ID, UT), (UT, AZ), (AZ, NV)]

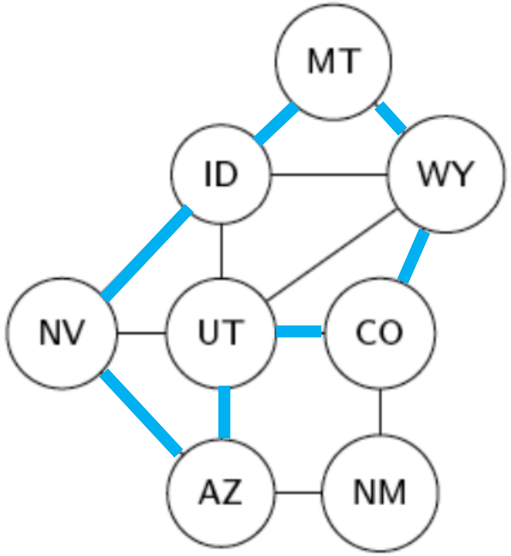
[NV, ID, MT, WY, ID, UT, AZ, NV]



A **cycle** is a **path** from a vertex back to itself, with no repeated **vertices**.

[(ID, MT), (MT, WY), (WY, CO), (CO, UT), (UT, AZ), (AZ, NV), (NV, ID)]

[ID, MT, WY, CO, UT, AZ, NV, ID]

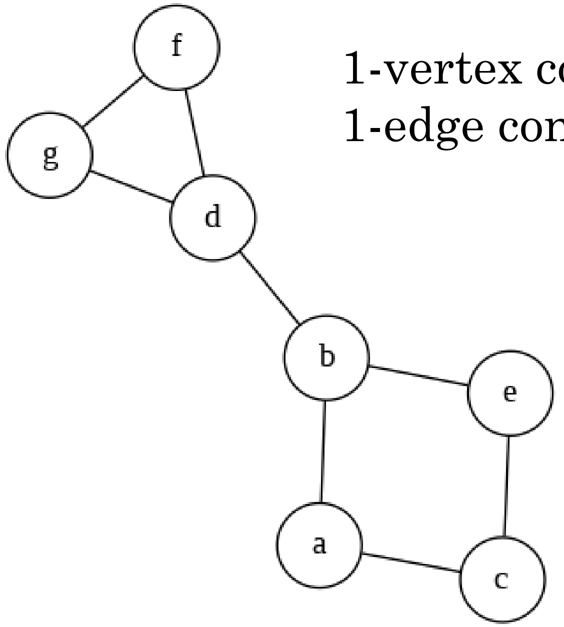


Graph Connectivity

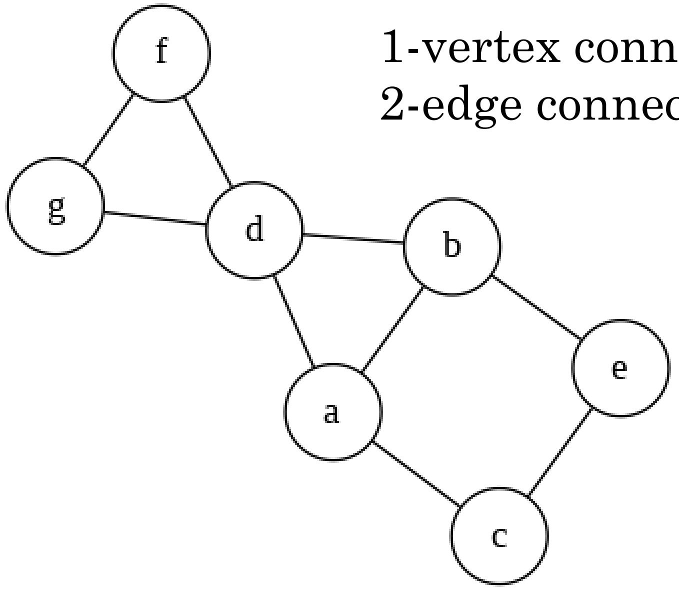
k-vertex connectivity: The minimum number of vertices to remove so the graph is no longer connected.

k-edge connectivity: The minimum number of edges to remove so the graph is no longer connected.

What is the vertex connectivity and edge connectivity of these graphs?



1-vertex connected
1-edge connected



1-vertex connected
2-edge connected

Graph Connectivity

Additional exercises:

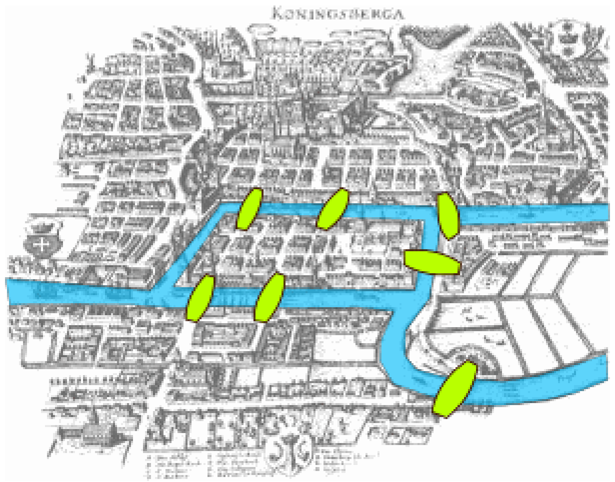
13.5.2

Euler Circuits

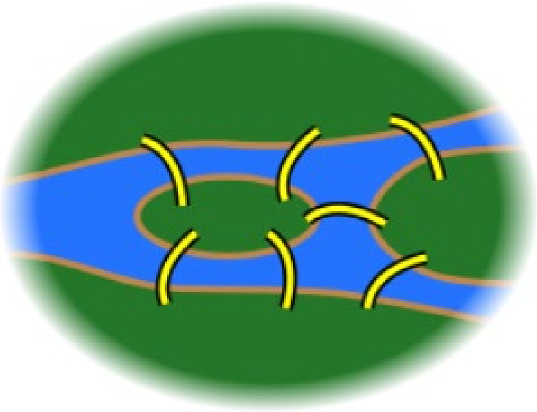
An Euler circuit is a circuit that contains every edge and every vertex.

No edges are repeated.

Seven Bridges of Königsberg

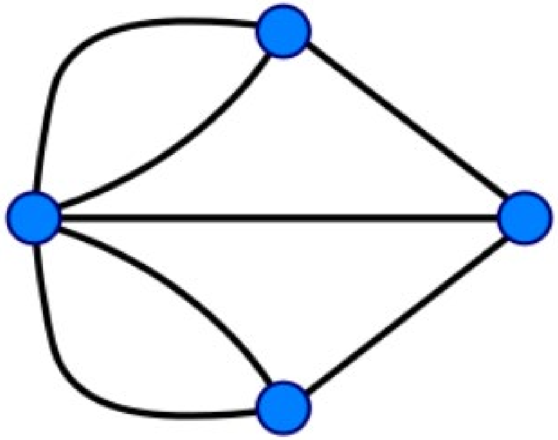


→



→

Can you find an Euler circuit in this graph?



Euler Circuits

What are some examples of graphs in which we would want to visit every edge once and only once?

Plowing roads (vertices are intersections and roads are edges)

Mail delivery

Euler Circuits

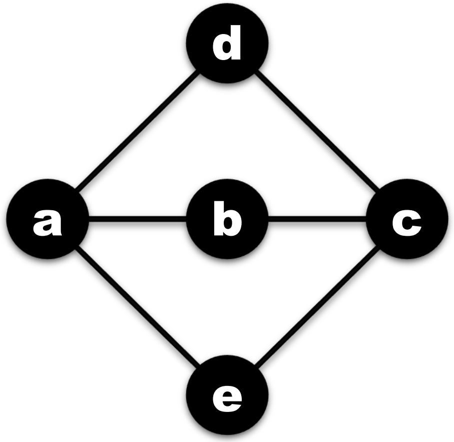
A **circuit** has no repeated edges.

If an undirected graph G has an Euler circuit, then G is connected and **every** vertex in G has an **even degree**.

If a graph has a vertex with an odd degree, then the graph cannot have an Euler circuit.

If an undirected graph G is connected and every vertex in G has an even degree, then G has an Euler circuit.

Try to find an Euler circuit in this graph:



Start at **a**.

If we start at **a**, we must end at **a** because it is a circuit.

But if we leave **a**, then come back to **a**, there is still one more edge we haven't traversed.

If we traverse that last edge, we are not longer ending at **a**, so it is not a circuit.

An undirected graph G has an Euler circuit **if and only if G is connected** and **every vertex in G has even degree**.

Finding a circuit in a graph

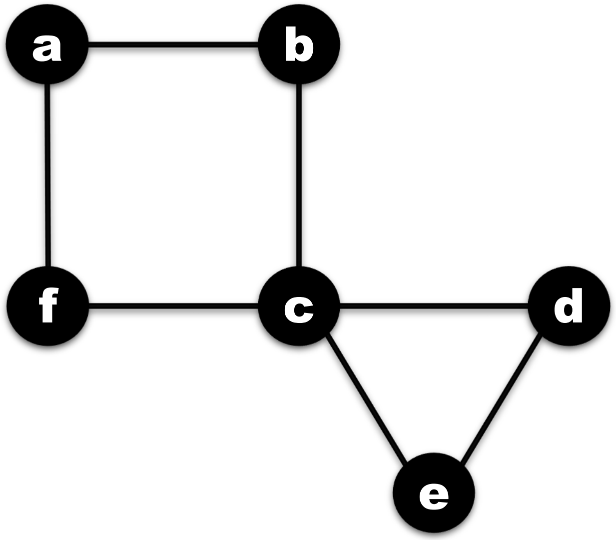
All vertices must have even degree.

Find a vertex w, that is not an isolated vertex.

Select any edge {w, x} incident to w. (Since w is not isolated, there is always at least one such edge.)

Current trail T := ⟨w, x⟩
last := x

While there is an edge {last, y} that has not been used in T:
    Add y to the end of T
    last := y

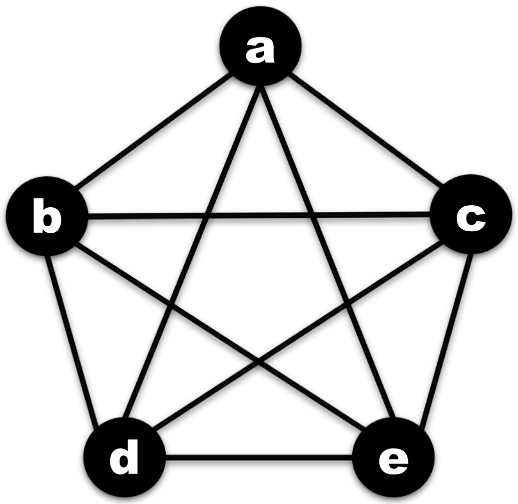


Example:

$T = (a, b, c, f, a)$

Finding an Euler Circuit in a graph

Find a Euler Circuit by tracing the algorithm in on the following graph:



Prerequisites:

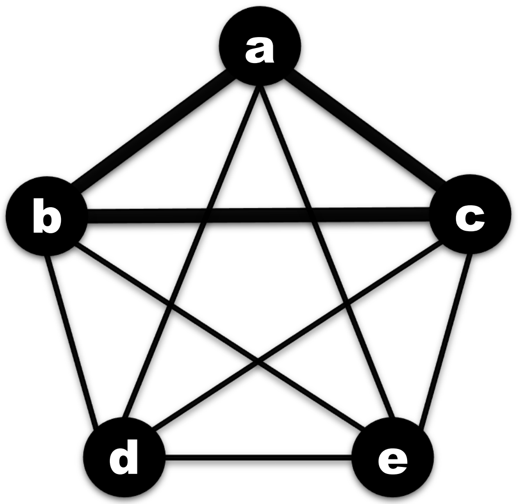
- Is the graph connected?
- Do all vertices have an even degree?
- Find any circuit in graph G and call it C

Procedure:

1. Remove all edges in C from G. Remove any isolated vertices and call the new graph G'.
2. Find a vertex w that is in G' and C.
3. Find a circuit in G' that begins and ends with w. Call this circuit C'.
4. Combine C and C'. Start with any vertex v in C and follow the edges in C until w is reached. Then follow the edges in C' back to w and the rest of the edges in C back to v. Call this new circuit C for the next iteration.
5. Repeat until all edges are in C.

Finding an Euler Circuit in a graph

Find a Euler Circuit by tracing the algorithm in on the following graph:



C = (a, b, c, a)

Prerequisites:

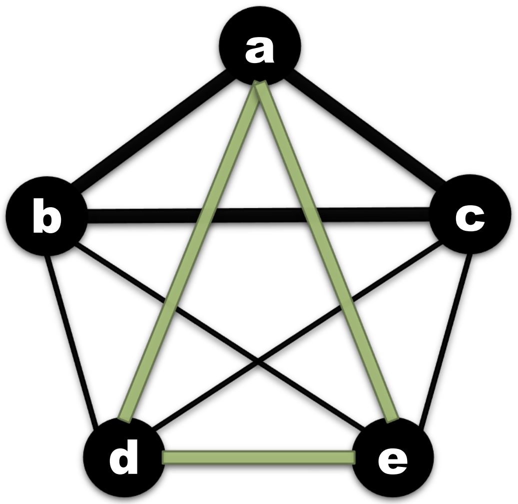
- Is the graph connected?
- Do all vertices have an even degree?
- Find any circuit in graph G and call it C

Procedure:

1. Remove all edges in C from G. Remove any isolated vertices and call the new graph G'.
2. Find a vertex w that is in G' and C.
3. Find a circuit in G' that begins and ends with w. Call this circuit C'.
4. Combine C and C'. Start with any vertex v in C and follow the edges in C until w is reached. Then follow the edges in C' back to w and the rest of the edges in C back to v. Call this new circuit C for the next iteration.
5. Repeat until all edges are in C.

Finding an Euler Circuit in a graph

Find a Euler Circuit by tracing the algorithm in on the following graph:



C = (a, b, c, a)

C' = (a, d, e, a)

Prerequisites:

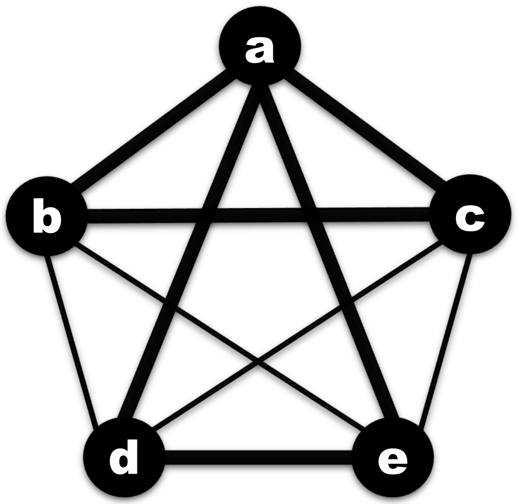
- Is the graph connected?
- Do all vertices have an even degree?
- Find any circuit in graph G and call it C

Procedure:

1. Remove all edges in C from G. Remove any isolated vertices and call the new graph G'.
2. Find a vertex w that is in G' and C.
3. Find a circuit in G' that begins and ends with w. Call this circuit C'.
4. Combine C and C'. Start with any vertex v in C and follow the edges in C until w is reached. Then follow the edges in C' back to w and the rest of the edges in C back to v. Call this new circuit C for the next iteration.
5. Repeat until all edges are in C.

Finding an Euler Circuit in a graph

Find a Euler Circuit by tracing the algorithm in on the following graph:



C = (a, b, c, a)

C' = (a, d, e, a)

C = (a, d, e, a, b, c, a)

Prerequisites:

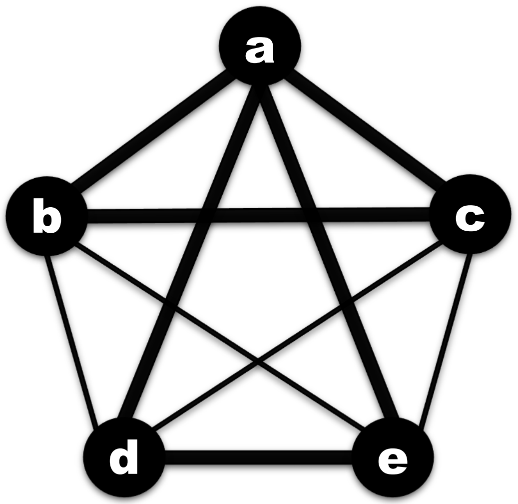
- Is the graph connected?
- Do all vertices have an even degree?
- Find any circuit in graph G and call it C

Procedure:

1. Remove all edges in C from G. Remove any isolated vertices and call the new graph G'.
2. Find a vertex w that is in G' and C.
3. Find a circuit in G' that begins and ends with w. Call this circuit C'.
4. Combine C and C'. Start with any vertex v in C and follow the edges in C until w is reached. Then follow the edges in C' back to w and the rest of the edges in C back to v. Call this new circuit C for the next iteration.
5. Repeat until all edges are in C.

Finding an Euler Circuit in a graph

Find a Euler Circuit by tracing the algorithm in on the following graph:



C = (a, d, e, a, b, c, a)

Prerequisites:

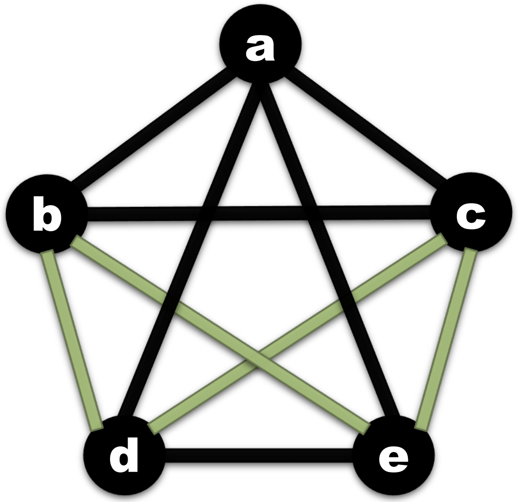
- Is the graph connected?
- Do all vertices have an even degree?
- Find any circuit in graph G and call it C

Procedure:

1. Remove all edges in C from G. Remove any isolated vertices and call the new graph G'.
2. Find a vertex w that is in G' and C.
3. Find a circuit in G' that begins and ends with w. Call this circuit C'.
4. Combine C and C'. Start with any vertex v in C and follow the edges in C until w is reached. Then follow the edges in C' back to w and the rest of the edges in C back to v. Call this new circuit C for the next iteration.
5. Repeat until all edges are in C.

Finding an Euler Circuit in a graph

Find a Euler Circuit by tracing the algorithm in on the following graph:



C = (a, d, e, a, b, c, a)

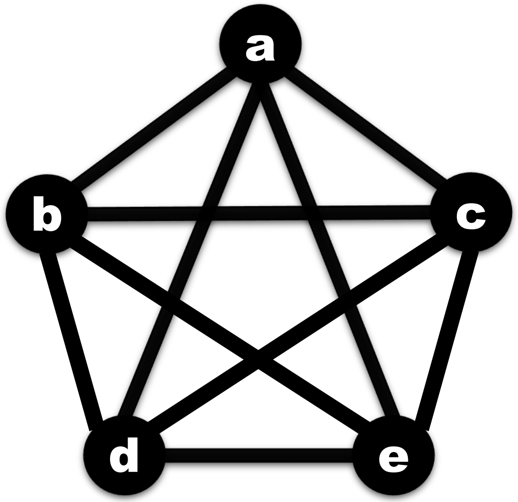
C' = (b, d, c, e, b)

Prerequisites:
- Is the graph connected?
- Do all vertices have an even degree?
- Find any circuit in graph G and call it C

Procedure:
1. Remove all edges in C from G. Remove any isolated vertices and call the new graph G'.
2. Find a vertex w that is in G' and C.
3. Find a circuit in G' that begins and ends with w. Call this circuit C'.
4. Combine C and C'. Start with any vertex v in C and follow the edges in C until w is reached. Then follow the edges in C' back to w and the rest of the edges in C back to v. Call this new circuit C for the next iteration.
5. Repeat until all edges are in C.

Finding an Euler Circuit in a graph

Find a Euler Circuit by tracing the algorithm in on the following graph:



C = (a, d, e, a, b, c, a)

C' = (b, d, c, e, b)

C = (a, d, e, a, b, d, c, e, b, c, a)

Prerequisites:

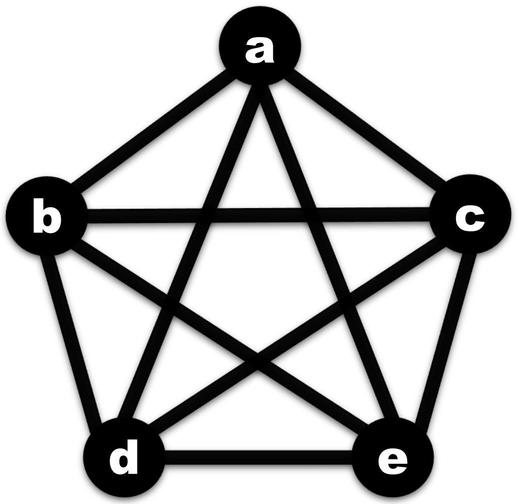
- Is the graph connected?
- Do all vertices have an even degree?
- Find any circuit in graph G and call it C

Procedure:

1. Remove all edges in C from G. Remove any isolated vertices and call the new graph G'.
2. Find a vertex w that is in G' and C.
3. Find a circuit in G' that begins and ends with w. Call this circuit C'.
4. Combine C and C'. Start with any vertex v in C and follow the edges in C until w is reached. Then follow the edges in C' back to w and the rest of the edges in C back to v. Call this new circuit C for the next iteration.
5. Repeat until all edges are in C.

Finding an Euler Circuit in a graph

Find a Euler Circuit by tracing the algorithm in on the following graph:



C = (a, d, e, a, b, d, c, e, b, c, a)

Prerequisites:

- Is the graph connected?
- Do all vertices have an even degree?
- Find any circuit in graph G and call it C

Procedure:

1. Remove all edges in C from G. Remove any isolated vertices and call the new graph G'.
2. Find a vertex w that is in G' and C.
3. Find a circuit in G' that begins and ends with w. Call this circuit C'.
4. Combine C and C'. Start with any vertex v in C and follow the edges in C until w is reached. Then follow the edges in C' back to w and the rest of the edges in C back to v. Call this new circuit C for the next iteration.
5. Repeat until all edges are in C.

With a partner, do Additional Exercises

13.6.1 a, b, c

Hamiltonian Cycles

A **cycle** has no repeated edges **or** vertices.

A Hamiltonian cycle is a cycle that includes every **vertex** in the graph (but not necessarily every edge).

A Hamiltonian path is a path that contains every vertex in the graph.

Hard to find efficiently. Brute force is the only known approach.

How would we find a Hamiltonian Path using brute force?

List every permutation of the orderings of the vertices and check each one to see whether it is a path.

We can sometimes prove a graph does or does not have a Hamiltonian cycle.

Any graph containing a node with degree 1 does not have a Hamiltonian cycle.

Any K graph K_3 or above does have a Hamiltonian cycle.

Hamiltonian Cycles

What are some situations in which this would be useful?

UPS delivery. Stops are the vertices and edges are the roads
(Don't need to visit every edge, but we do need to visit every vertex)

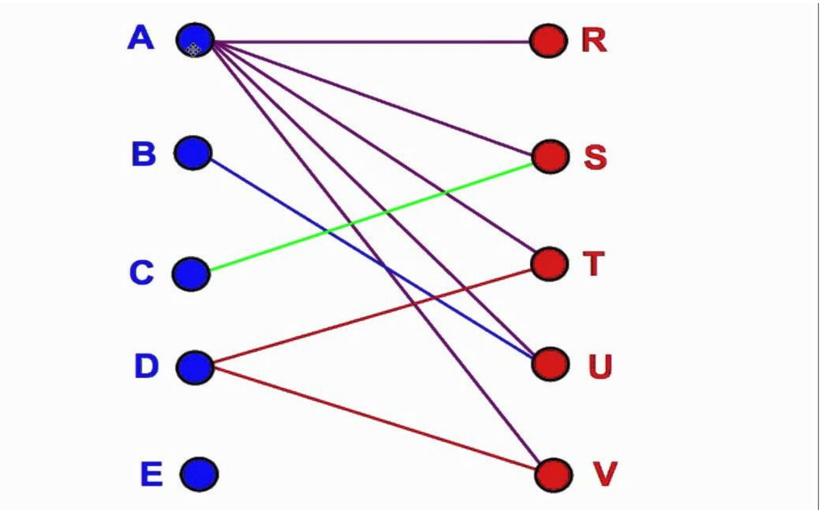
Traveling Salesman. The salesman wants to visit every city and get back home.

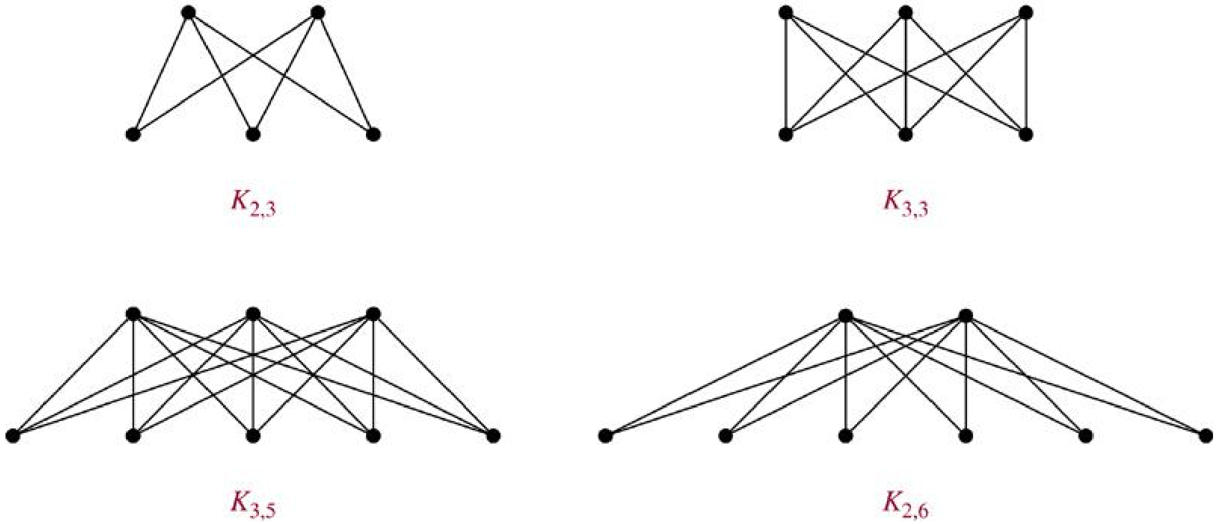
With a partner:

Do additional exercise 13.7.1 and 13.7.2

Bipartite Graph

- The graph consists of two sub-graphs. Each element of each subgraph has adjacent nodes ONLY in the other sub-graph



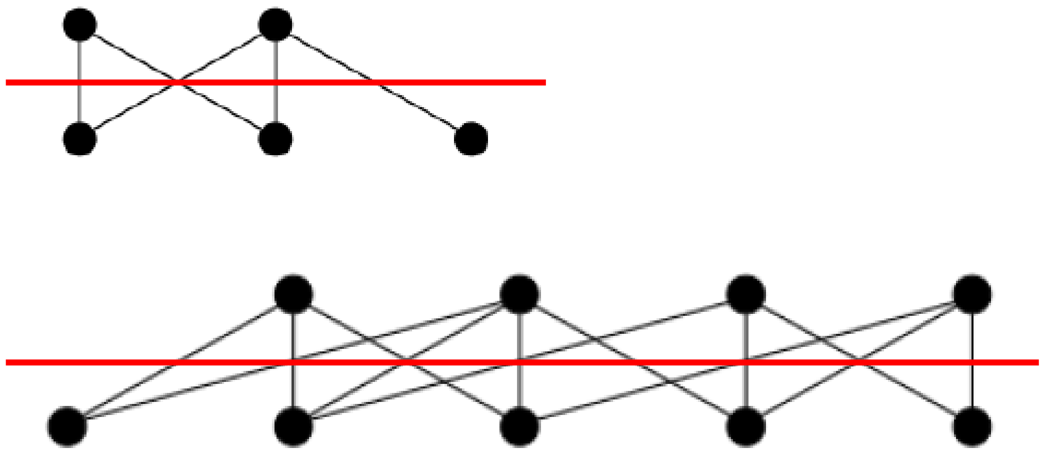


Bipartite Graphs

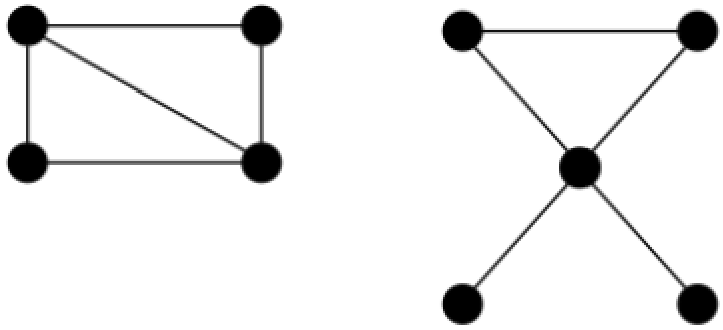
- Can be divided into two sets of vertices such that none of the vertices in either set are connected by an edge

Example:

Bipartite

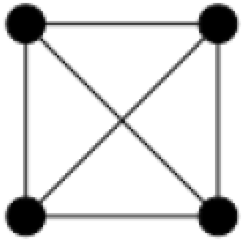


Not Bipartite



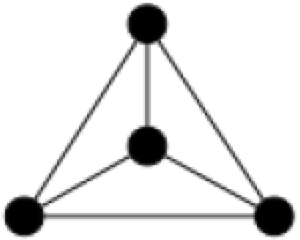
Planar Graphs

- A graph that can be drawn without lines crossing



K_4

Planar?



K_4 too

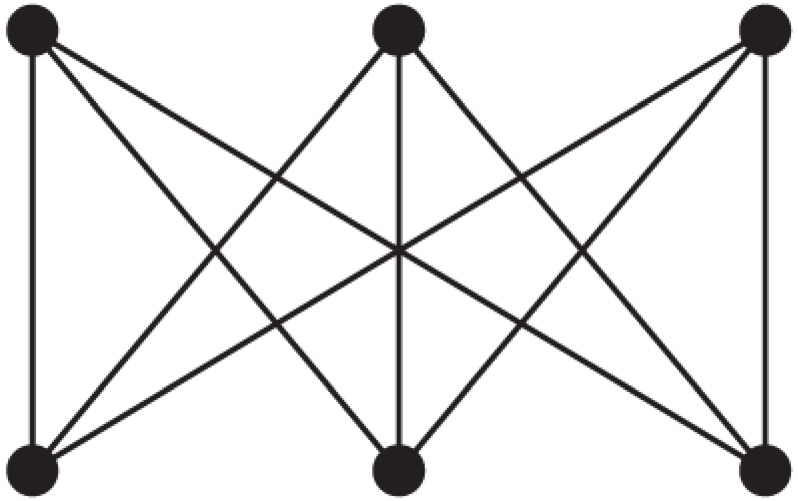
Planar?

Note these are the same graph, just drawn differently

Planar Graphs

Example:

Is it possible to connect three houses to three utilities (e.g., electricity, fiber, gas) without the lines crossing?



Can you draw $K_{3,3}$ as a planar graph?

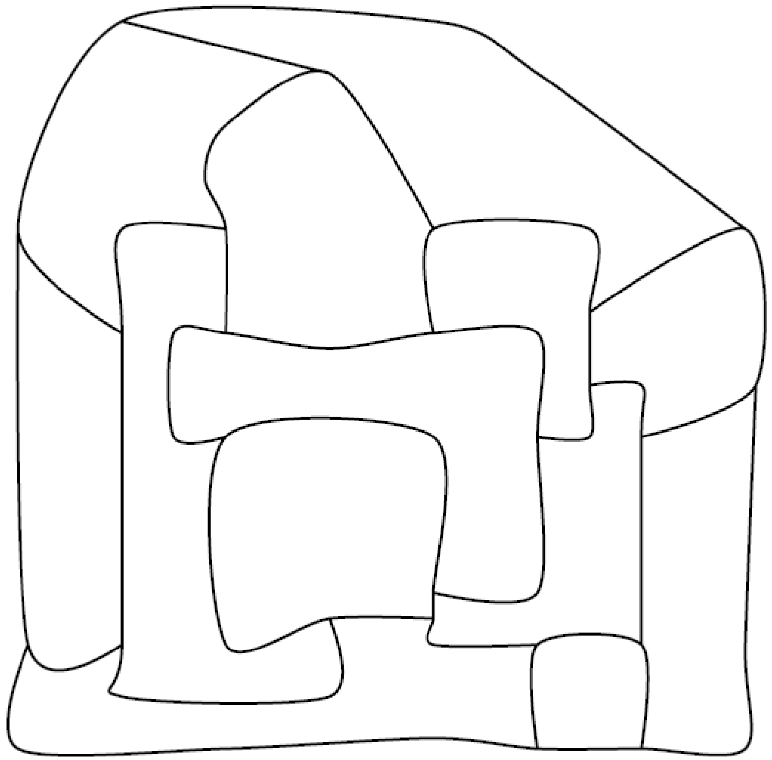
No, $K_{3,3}$ is not planar.

Coloring Graphs

Mapmakers in the fictional land of Euleria have drawn the borders of the various dukedoms of the land.

To make the map pretty, they wish to color each region. Adjacent regions must be colored differently, but it is perfectly fine to color two distant regions with the same color.

What is the fewest colors the mapmakers can use and still accomplish this task?

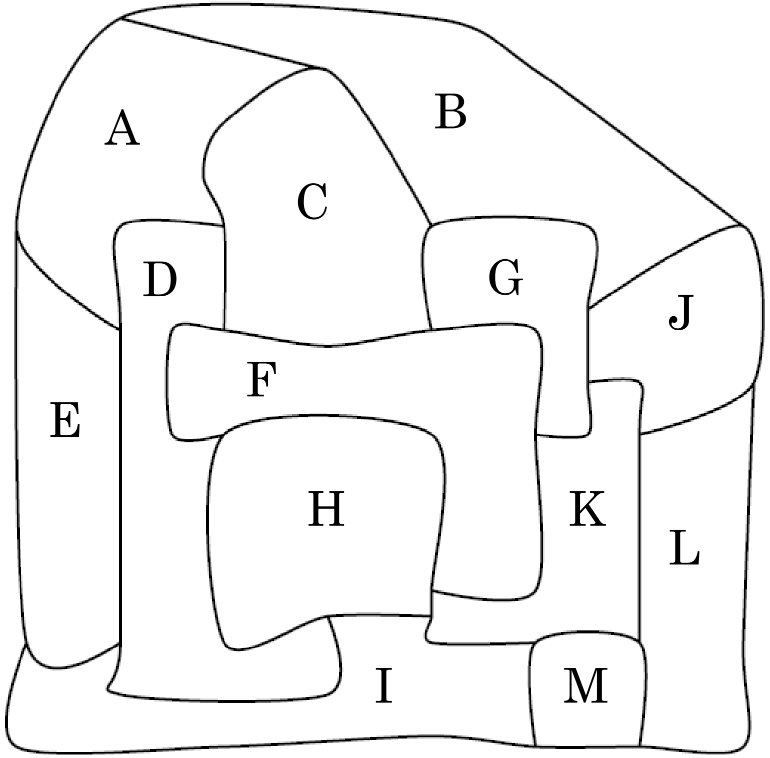


Coloring Graphs

Mapmakers in the fictional land of Euleria have drawn the borders of the various dukedoms of the land.

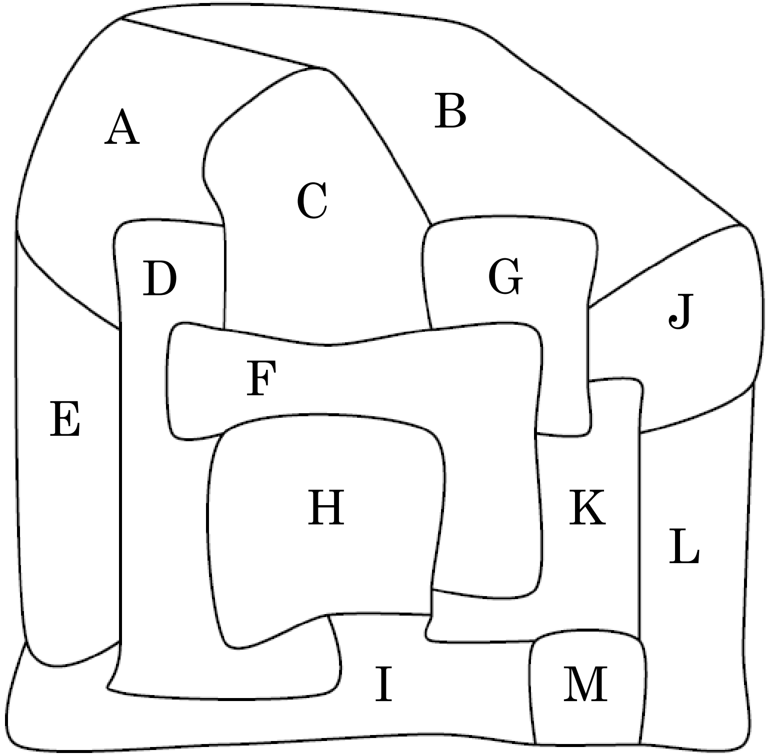
To make the map pretty, they wish to color each region. Adjacent regions must be colored differently, but it is perfectly fine to color two distant regions with the same color.

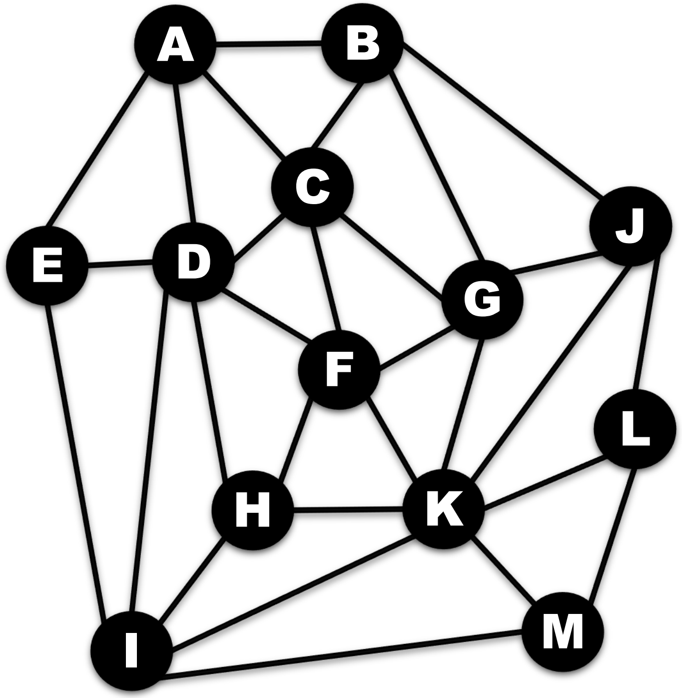
What is the fewest colors the mapmakers can use and still accomplish this task?



Levin, Oscar. *Discrete Mathematics, An Open Introduction, 3rd Edition*. 2021

Coloring Graphs

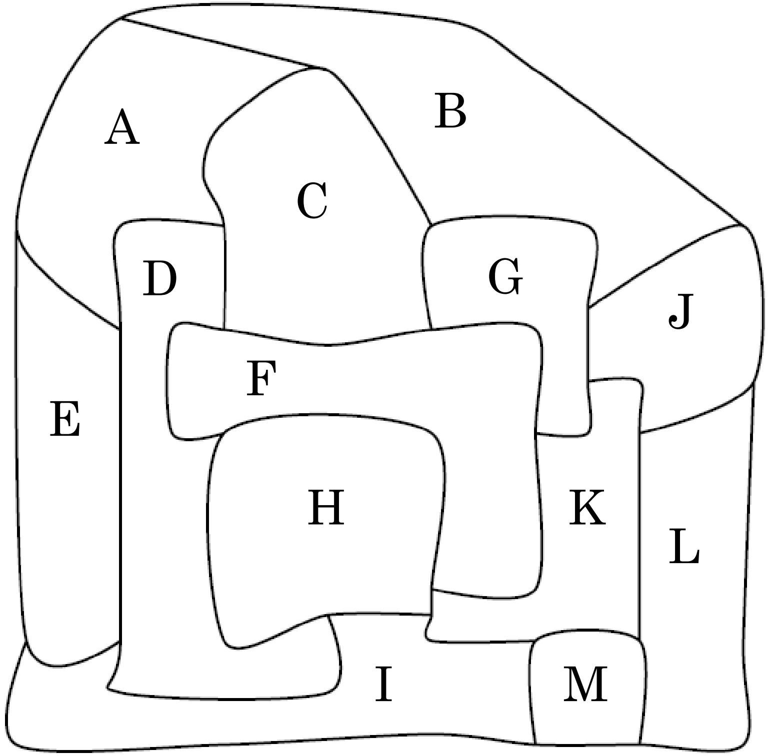


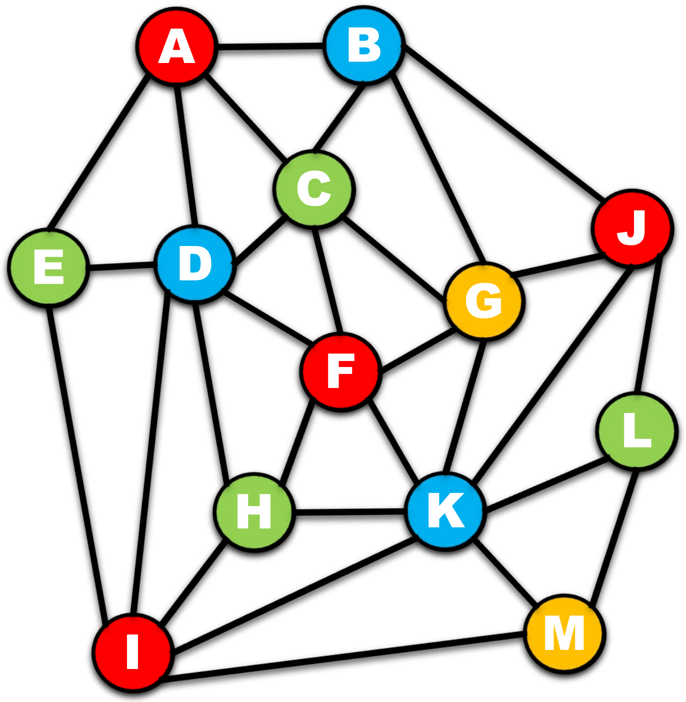


Levin, Oscar. *Discrete Mathematics, An Open Introduction, 3rd Edition*. 2021

Coloring Graphs

Let's color the nodes using the greedy algorithm, in alphabetical order.



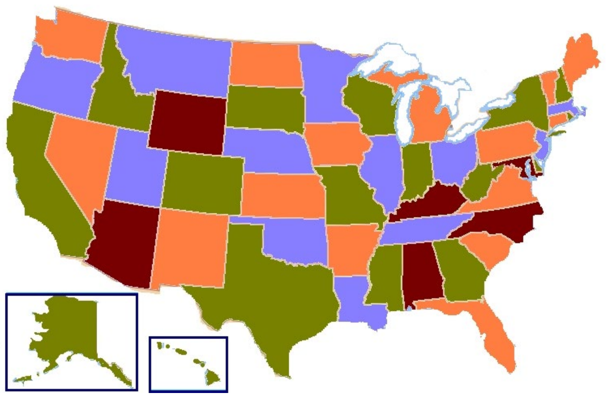


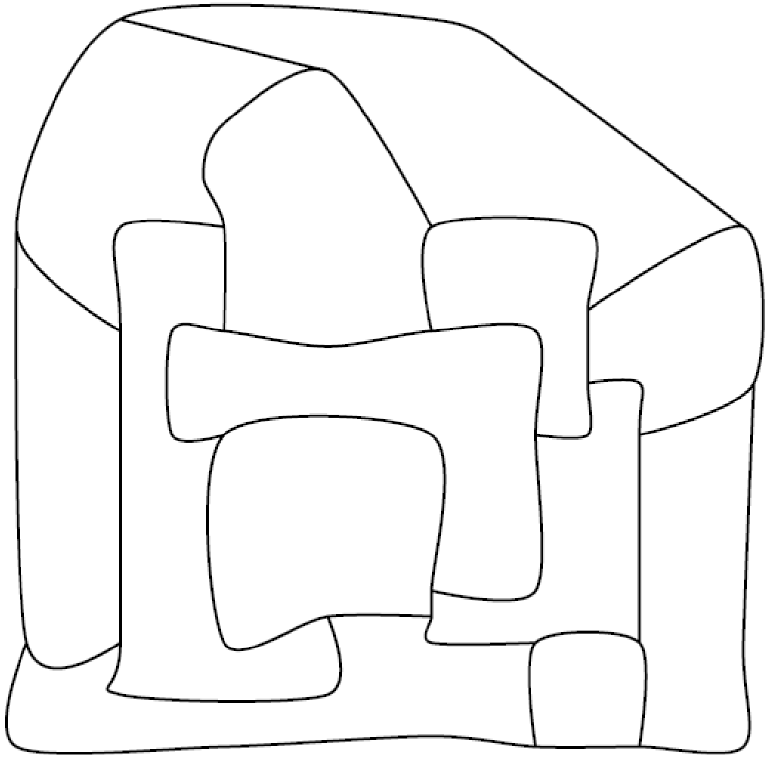
Levin, Oscar. *Discrete Mathematics, An Open Introduction, 3rd Edition*. 2021

Coloring Graphs

The Four Color Theorem (proved in 1976), states that if a graph G is planar, then its regions can be colored using 4 or fewer colors.

The current best proof requires computers to check 633 reducible configurations....not an easy proof.





Levin, Oscar. *Discrete Mathematics, An Open Introduction, 3rd Edition*. 2021

Additional exercises

13.9.1