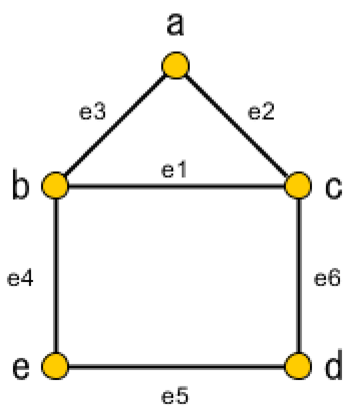
Introduction to Graphs and Trees

Graph

A graph $\boldsymbol{G} = (\boldsymbol{V}, \boldsymbol{E})$ is a set V of vertices connected by an edge set E.

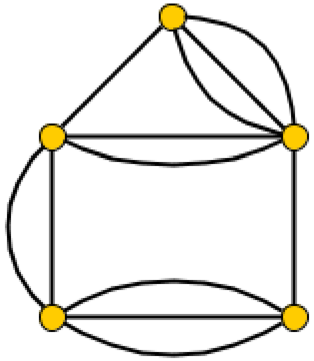


$V = \{a, b, c, d, e\}$

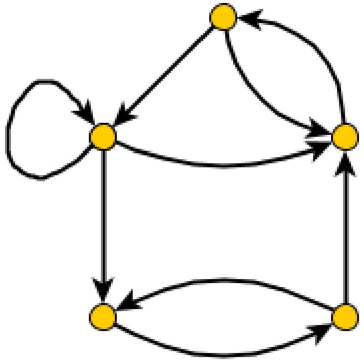
$E = \{e1, e2, e3, e4, e5, e6\} = \{(b, c), (c, a), (a, b), (b, e), (e, d), (d, c)\}$

Graph Variations

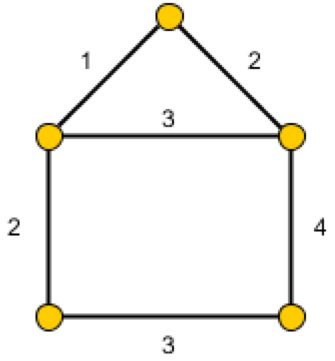
Multi-Graph: Multiple edges between two vertices.
Directed: Edges have a direction.
Weighted: Vertices and/or edges have weights.
Simple: No multiple edges, no loops.



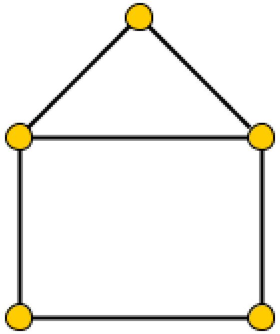
Multi-graph



Directed



Weighted Undirected



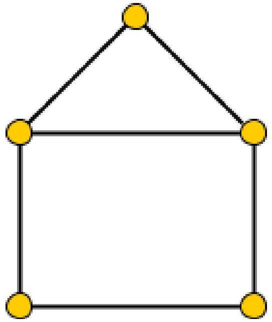
Simple Undirected

Undirected and Directed graphs

Simple Undirected Graph

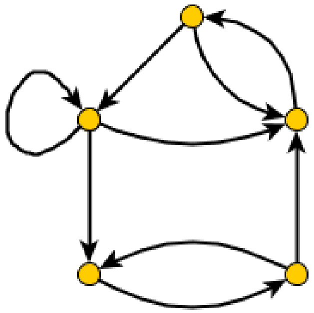
A Simple undirected graph is a set of vertices that are connected by the set of edges, where edges are an unordered pair of distinct vertices.

- In a simple undirected graph both multiple edges and loops are not allowed.



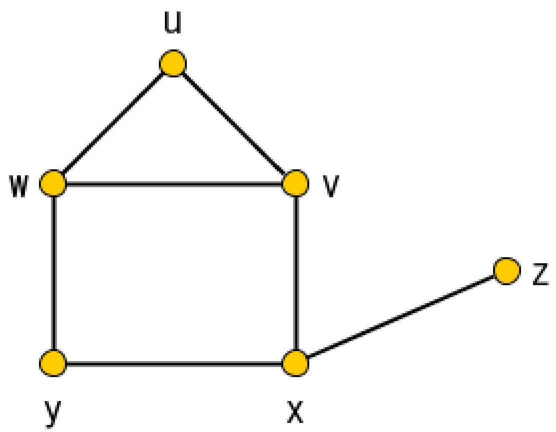
Directed Graph

A *directed graph* (or *digraph*) *(V ,E)* consists of a nonempty set of vertices *V* and a set of *directed edges* (or *arcs*) *E*. Each directed edge is associated with an ordered pair of vertices. The directed edge associated with the ordered pair (*u, v*) is said to *start* at *u* and *end* at *v*.



Simple Undirected Graphs

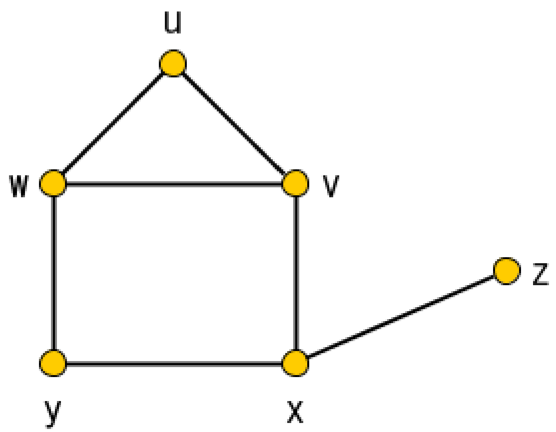
- Two vertices *u* and *v* are called *adjacent* (or *neighbors*) in undirected graph G if *u* and *v* are endpoints of an edge e of G. Such an edge e is called *incident with* the vertices *u* and *v* and e is said to *connect* *u* and *v*.



- w is adjacent with u, v and y but not with x and z.
- y is adjacent with x and w but not u, v and z.

Simple Undirected Graphs

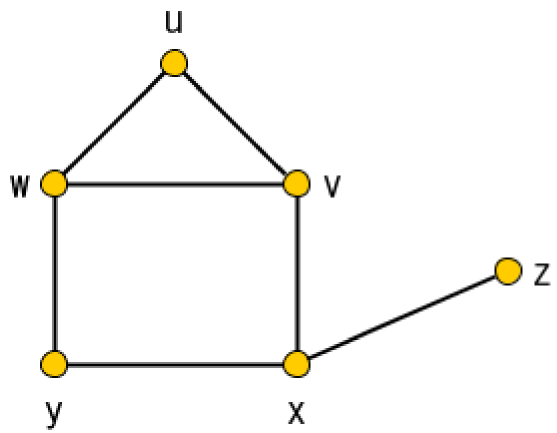
- Given a graph $G = (V, E)$
  - The open neighborhood $N(v) = \{ u \in V \mid u \neq v, uv \in E\}$ of a vertex v is the set of all vertices adjacent to v (not including v). The closed neighborhood $N[v] = N(v) \cup \{v\}$ includes v.



- $N(w) = \{u, v, y\}$
- $N[w] = \{u, v, y, w\}$
- $N(y) = \{w, x\}$

Simple Undirected Graphs - Degree

- The **degree** of a vertex v is the number of incident edges, denoted by $\deg(v)$.



- $\deg(w) = 3$
- $\deg(z) = 1$
- $\deg(y) = 2$

A vertex of degree one is called **pendant**. Consequently, a pendant vertex is adjacent to exactly one vertex.

- Vertex **z** is pendant.

Simple Undirected Graphs - Degree

- Let $G = (V, E)$ be a simple undirected graph, Then:

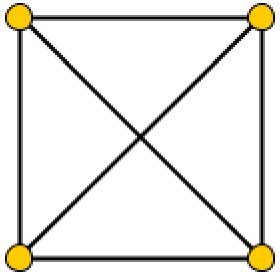
Lemma	$\sum_{v \in V} \deg(v) = 2\lvert E \rvert$

Corollary: Number of odd degree vertices is even.

Degree Sequence

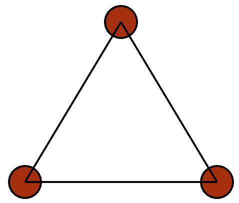
- It is impossible to make a graph with V = 6 where the vertices have degrees 1, 2, 2, 3, 3, 4.
  - This is because the sum of the degrees: 1+2+3+3+4=15
  - Sum of the degrees is always an even number but 15 is odd!

- A graph has 4 vertices with degrees 3, 3, 3, and 3. How many edges are there? What does this graph look like?
  - 6 edges

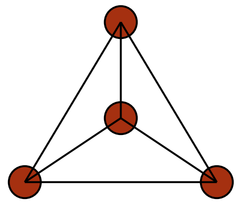


Degree Sequence

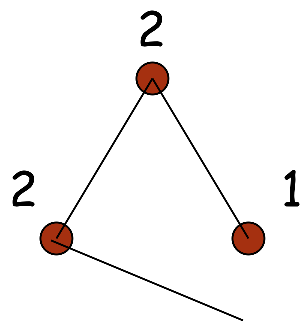
Is there a graph with degree sequence (2,2,2)? YES.



Is there a graph with degree sequence (3,3,3,3)? YES.

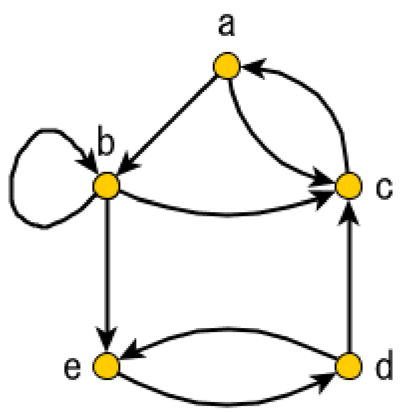


Is there a graph with degree sequence (2,2,1)? NO.



Directed Graphs - Degree

- For a vertex v, the **indegree** $indeg(v)$ is the number of **incoming** edges, and the **outdegree** $\mathrm{outdeg}(v)$ the number of **outgoing** edges.



- $\mathrm{indeg}(a) = 1, outdeg(a) = 2$
- $\mathrm{indeg}(b) = 2, outdeg(b) = 3$
- $\mathrm{indeg}(e) = 2, outdeg(e) = 1$

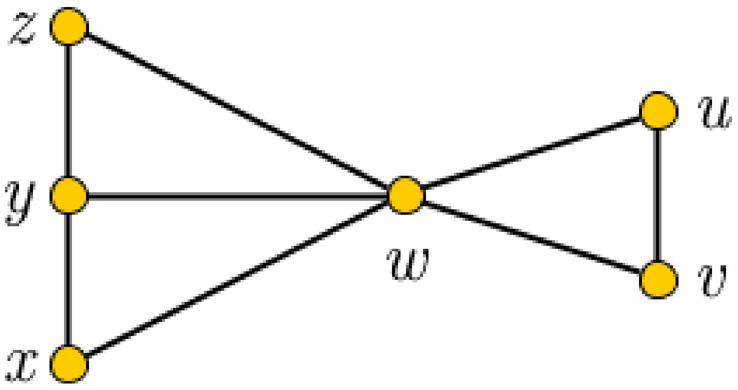
Directed Graphs - Degree

- If G is directed graph, Then:

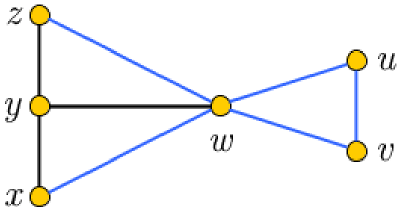
Lemma	$\sum_{v \in V} \text{indeg}(v) = \sum_{v \in V} \text{outdeg}(v) = \lvert E \rvert$

Path

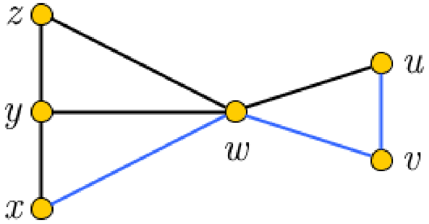
- **Path**: Is a sequence of adjacent vertices.
  - The **length** of a path from a vertex v to a vertex u is the **number of edges** in the path.
  - A path **P** of length l is sequence of $l + 1$ adjacent vertices.
- A path is **simple** if all vertices are different.



Path: (x,w,v,u,w,z)

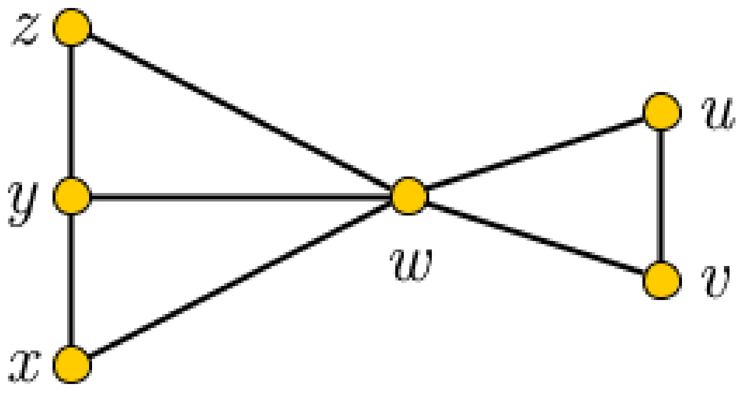


Simple Path: (x,w,v,u)

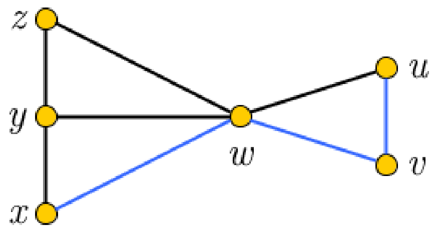


Shortest Paths

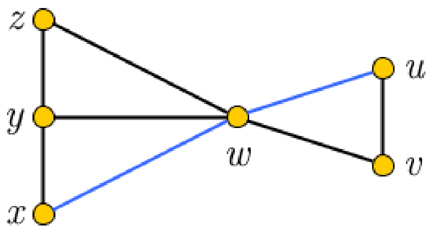
- A **Shortest path** between two vertices $u\ and\ v$ is path with the **minimum** number of edges.



A Path (x, w, v, u) is a simple path between $x\ and\ u$, but not shortest.

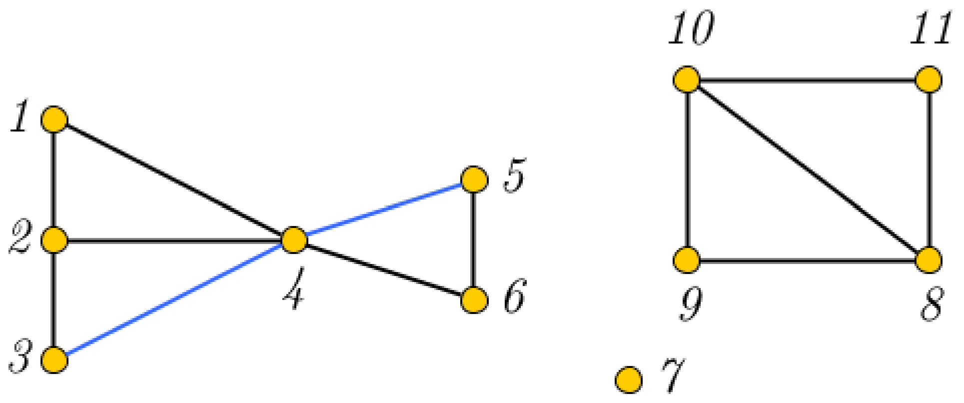


A Path (x, w, u) is a shortest path between $x\ and\ u$.



Connectedness

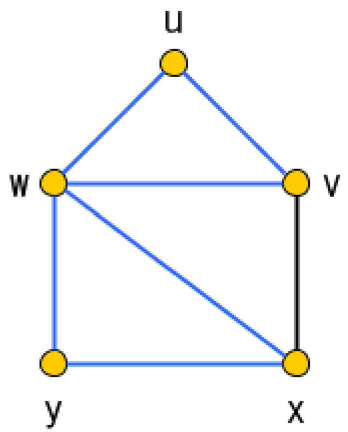
- Vertices v, w are connected **if and only if** there is a path starting at v and ending at w.
- A graph is **connected** iff every pair of vertices are connected. So a graph is connected if and only if it has only 1 connected component.
- Every graph consists of separate connected pieces called connected components



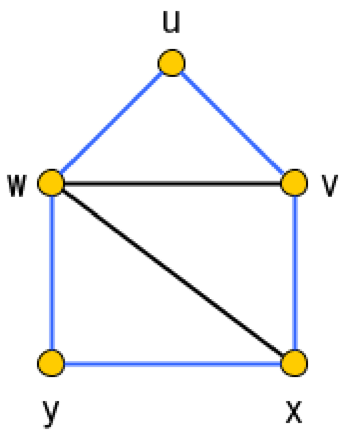
3 connected components

Cycle

- A **cycle** is a path that begins and ends with the same vertex.
- A cycle is **simple**, if it doesn't cross itself.



Cycle
(u,v,w,x,y,w,a)

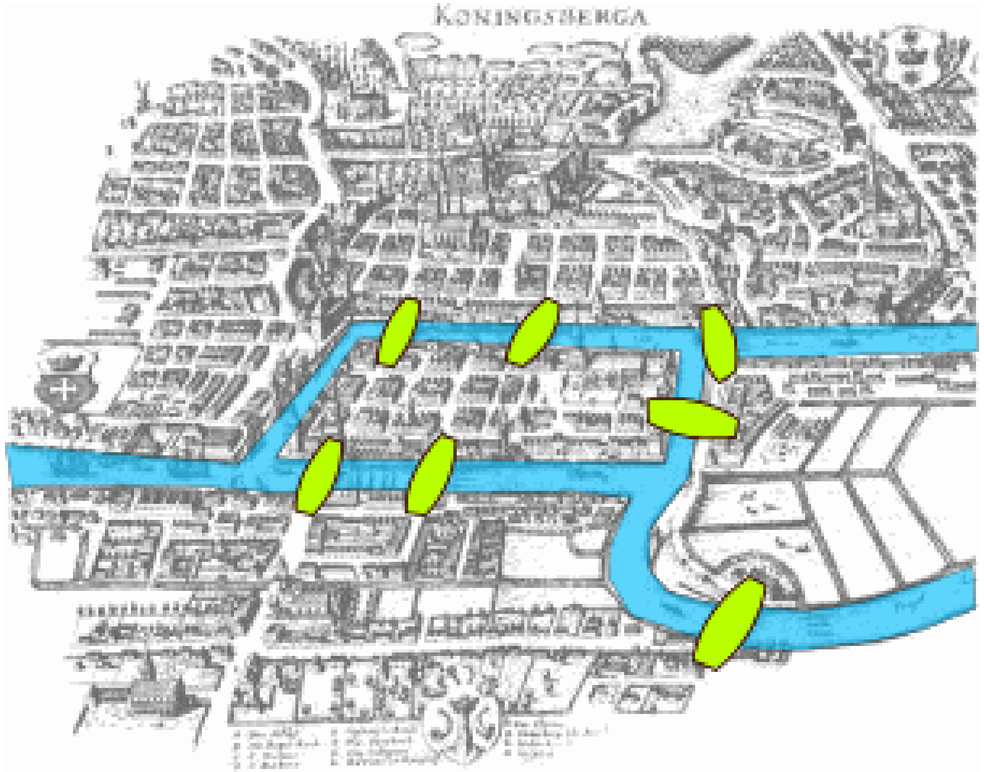


Simple Cycle
(u,v,x,y,w,a)

Eulerian Graphs

Seven Bridges of Königsberg

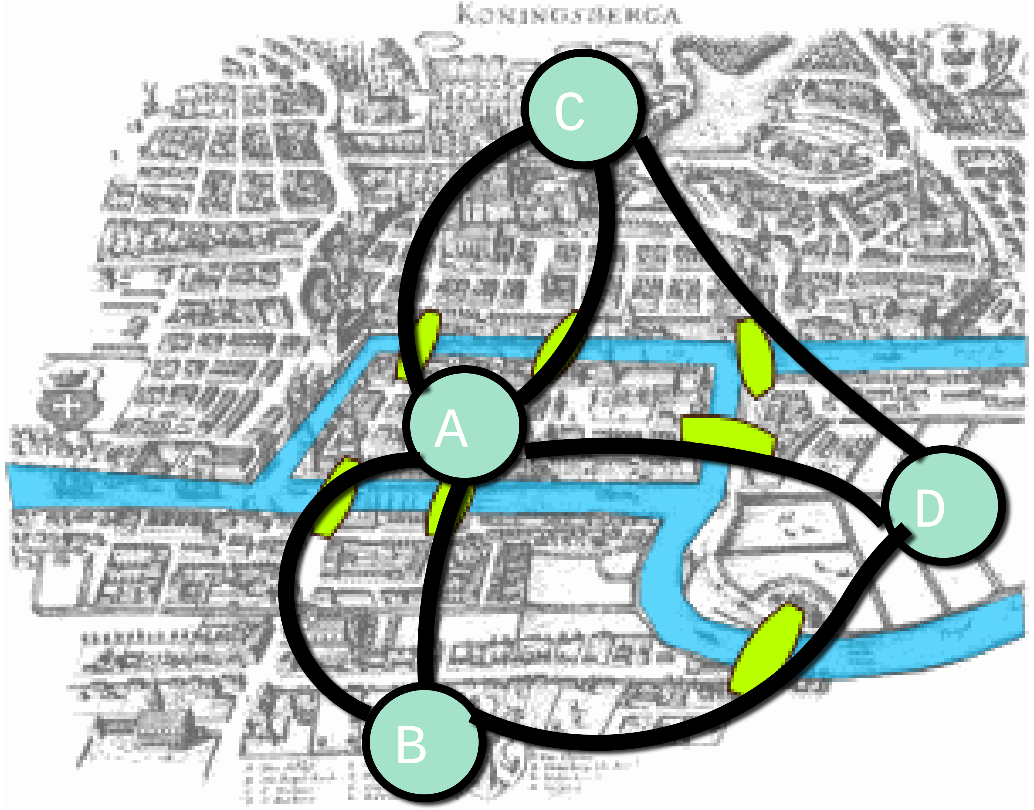
- **Königsberg** (now Kaliningrad, Russia) around 1735



Problem: Is it possible to walk with a route that crosses each bridge exactly once?

Seven Bridges of Königsberg

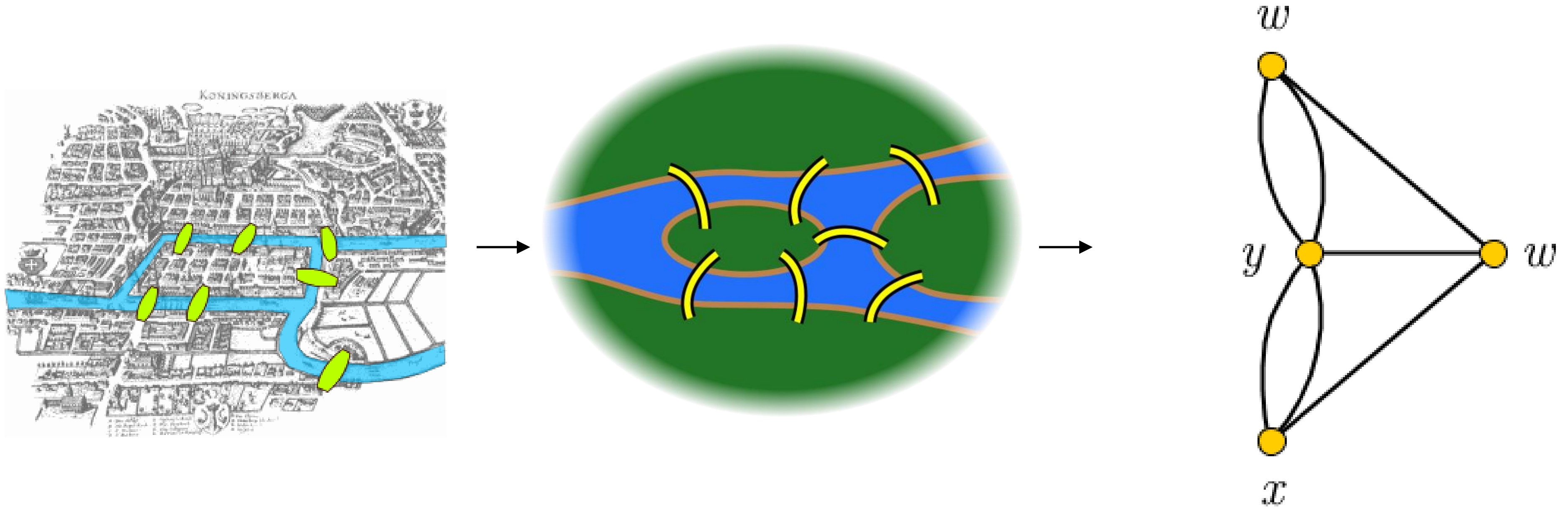
- **Königsberg** (now Kaliningrad, Russia) around 1735



Problem: Is it possible to walk with a route that crosses each bridge exactly once?

Seven Bridges of Königsberg

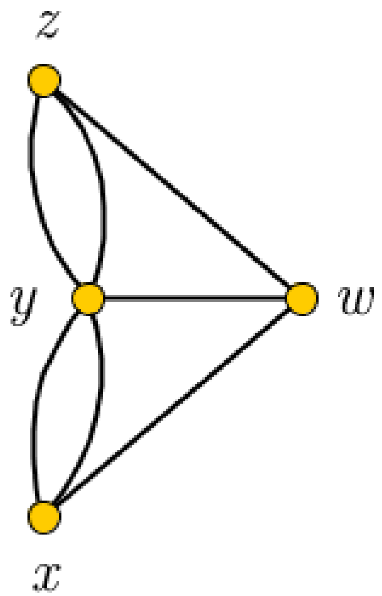
Represent the problem as graph:



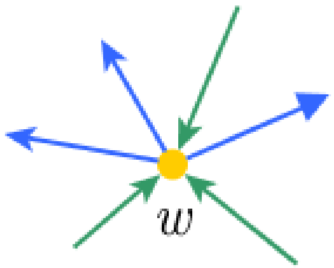
Graph problem: Is it possible to find a path in graph G that visits each edge exactly once?

Euler's Solution

- **Question:** Is it possible to find a walk that visits each edge exactly once.



Suppose there is such a walk, there is a starting point and an endpoint point.



For every "intermediate" point v, there must be the same number of incoming and outgoing edges, and so v must have an **even number of edges**.

So, at most **two** vertices can have odd number of edges.

Eulerian Cycle and Path

Eulerian Cycle: Is a cycle which visits every edge in a graph G exactly once.

Eulerian Path: Is a path which visits every edge in a graph G exactly once.

- An Euler cycle **starts** and **ends** at the **same vertex**.
- An Euler path **starts** and **ends** at **different vertices**.

Euler's Solution

Theorem

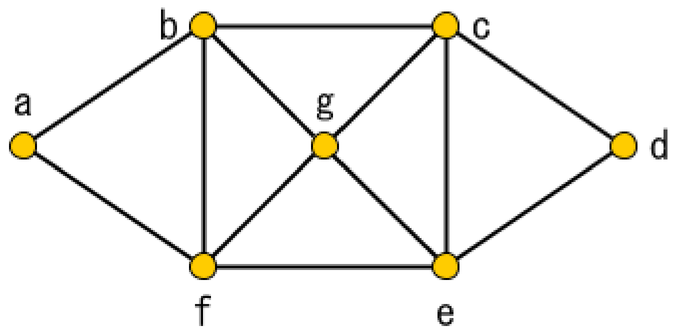
A connected graph has an **Eulerian cycle** if and only if the **degree** of all vertices is **even**.

Theorem

A connected graph has an **Eulerian path** if and only if it has at most two vertices with an odd degree.

Example

- Does the following graph have an Eulerian Cycle? If yes, list the edges in the order that they are walked. If no, explain why.



Yes, since all vertices have even degree.

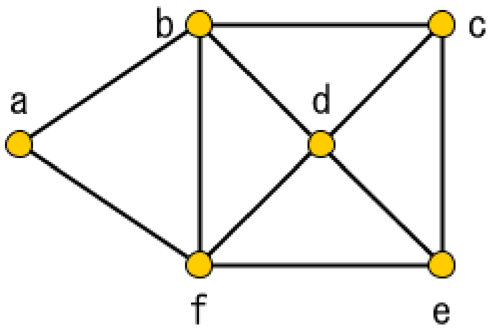
An Euler cycle: **abfgbcdecgefa**

Another Euler Cycle: **gfbafedcegcbg**

- Does the graph have an Eulerian Path?

Example

- Does the following graph have an Eulerian Cycle? If yes, list the edges in the order that they are walked. If no, explain why.



No, since not all vertices are even degree.
c and e have degree of 3.

- Does the graph have an Eulerian path? If yes, list the edges in the order that they are walked. If no, explain why.

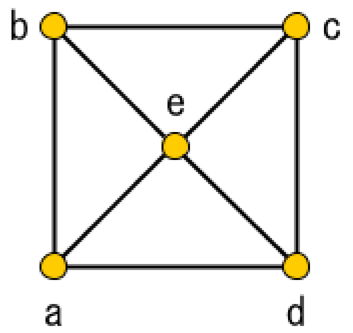
Yes, since the graph has at most two vertices with an odd degree..

An Euler path: **efabfdbcdec**

Another Euler path: **ecdefabfdbc**

Example

- Does the following graph have an Eulerian Cycle? If yes, list the edges in the order that they are walked. If no, explain why.



No, since not all vertices are even degree.

- Does the graph have an Eulerian path? If yes, list the edges in the order that they are walked. If no, explain why.

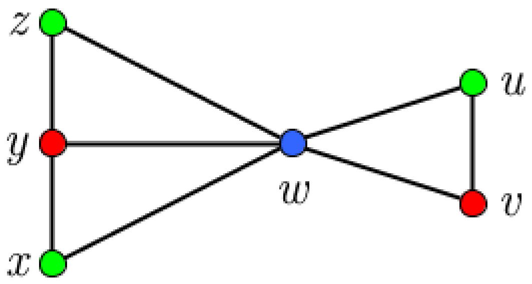
No, number of vertices with odd degree are greater than 2.

a, c, d and e have degree of 3.

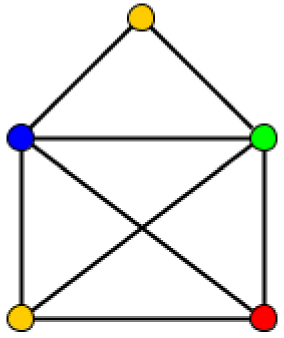
Graph Coloring

Graph Coloring

- A ***coloring*** of a simple graph is the assignment of a **color** to each vertex of the graph so that **no** two adjacent vertices are assigned the same color.
  - The **minimum number** of colors needed for coloring a graph is called the **chromatic number**.



The chromatic number of this graph is 3



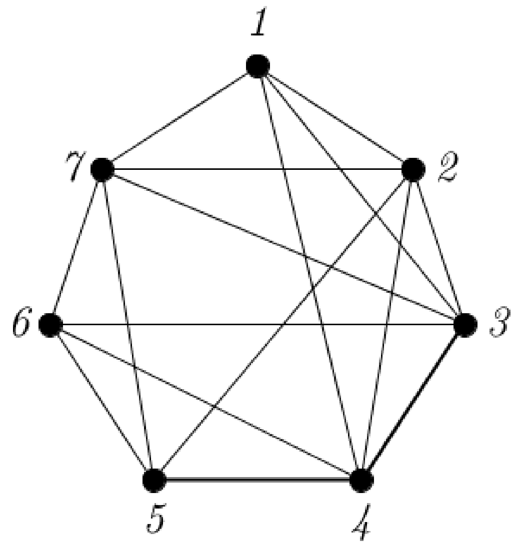
The chromatic number of this graph is 4

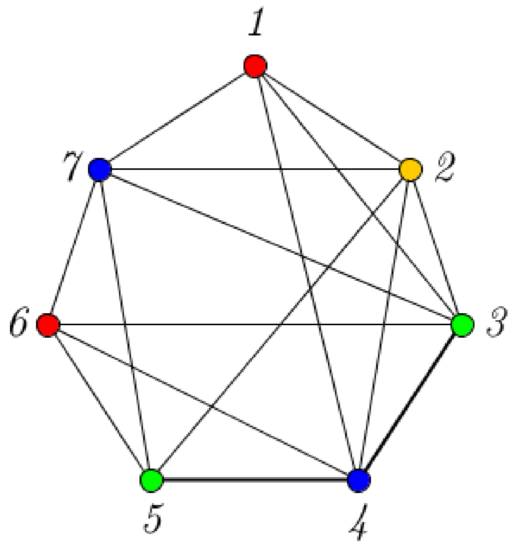
Applications of Graph Coloring

- **Scheduling Final Exams** How can the final exams at a university be scheduled so that no student has two exams at the same time?
- The scheduling problem can be solved using a graph model
  - with vertices representing courses and
  - with an edge between two vertices if there is a common student in the courses they represent.
  - Each time slot for a final exam is represented by a different color.

Example

- For instance,
  - suppose there are **seven** finals to be scheduled. Suppose the courses are numbered 1 through 7.
  - Suppose that the following pairs of courses have common students: 1 and 2, 1 and 3, 1 and 4, 1 and 7, 2 and 3, 2 and 4, 2 and 5, 2 and 7, 3 and 4, 3 and 6, 3 and 7, 4 and 5, 4 and 6, 5 and 6, 5 and 7, and 6 and 7.

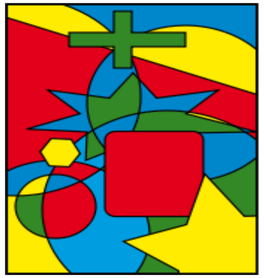




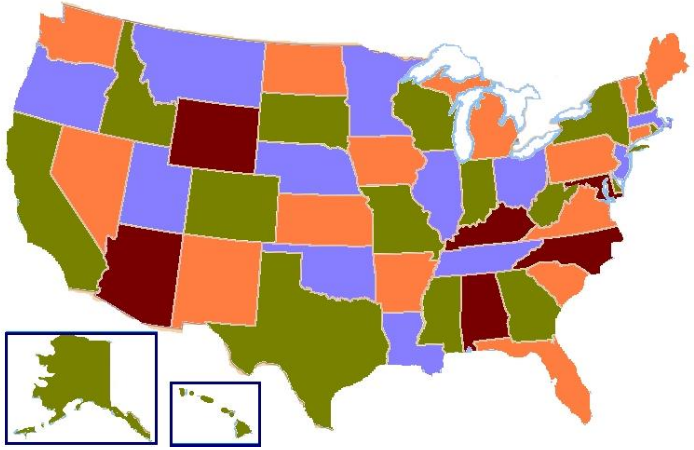
Because the chromatic number of this graph is 4, **four time slots are needed**.

Four Color Theorem

- The Four Color Theorem – ***the chromatic number of a planar graph is no greater than four.***
- Proof by Appel and Haken 1976.
- careful case analysis performed by computer;
- The computer program ran for hundreds of hours. The first significant *computer-assisted* mathematical proof. *Write-up was hundreds of pages including code!*

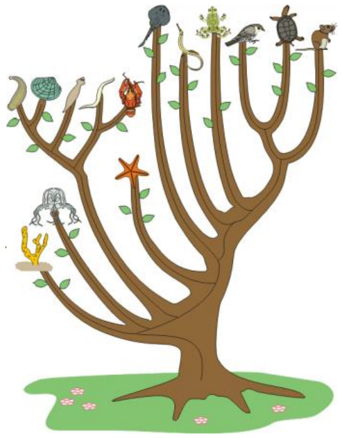


Four color map.

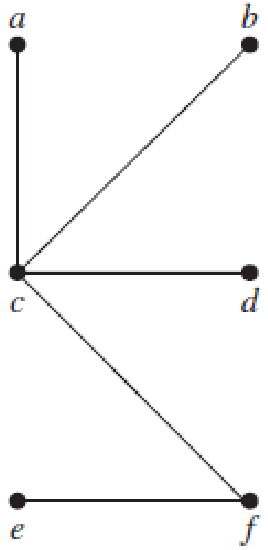


For more details:
https://en.wikipedia.org/wiki/Four_color_theorem

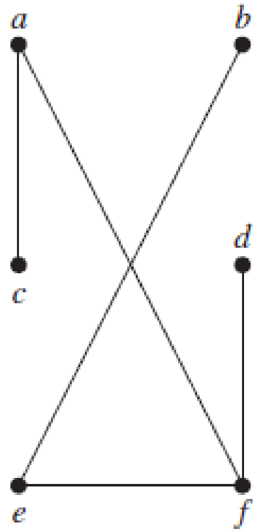
Tree



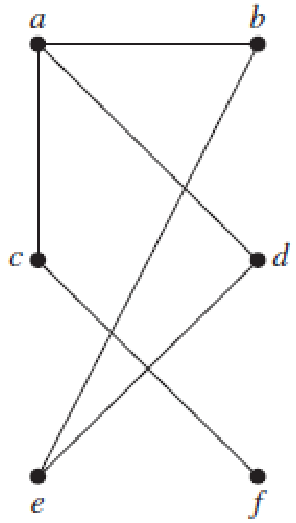
A *tree* is a connected undirected graph with no simple cycles.



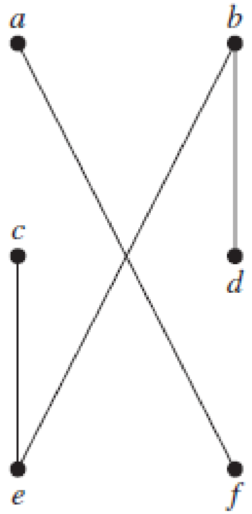
It is a tree since all vertices are connected and no cycles.



It is a tree since all vertices are connected and no cycles.



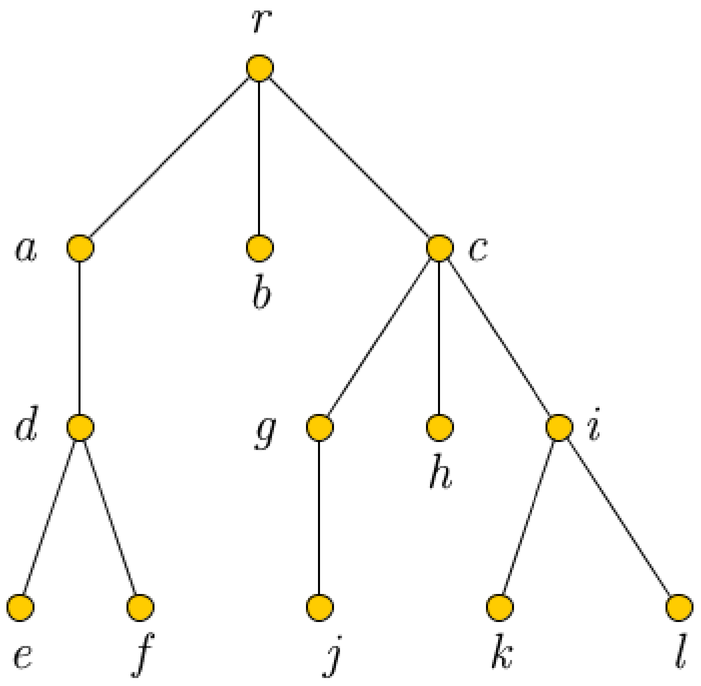
It is not a tree because *e*, *b*, *a*, *d*, *e* is a simple cycle.



It is not a tree because *it is connected*.

Rooted Tree

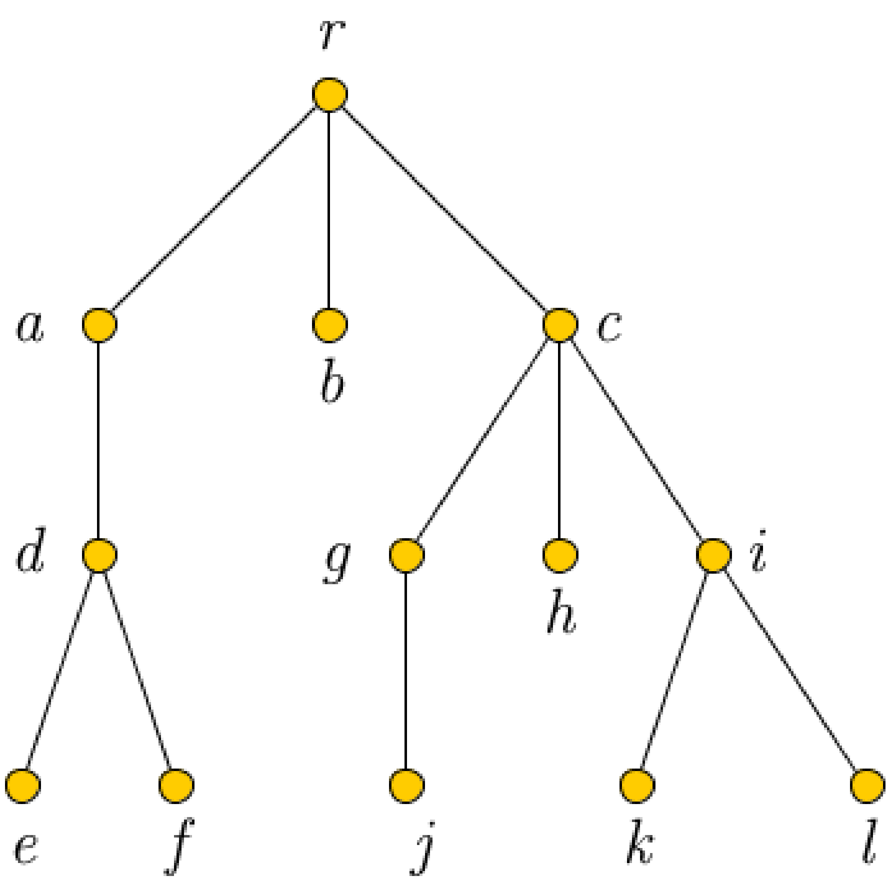
A ***rooted tree*** is a tree in which one vertex has been designated as the **root** and every edge is directed away from the root.



Tree Terminologies

- A vertex of a rooted tree is called a **leaf** if it has no children.
- Vertices that have children are called **internal vertices**.
- Vertices with the same parent are called **siblings**.
- The **ancestors** of a vertex are the vertices in the path from the root to this vertex.
- The **descendants** of a vertex v are those vertices that have v as an ancestor.
- **Height of a node v** is the number of edges on the *longest path* from v to a leaf. A leaf node will have a height of 0.
- **height of a tree:** the largest level of the vertices of a tree. It is he height of a root.

Example



- The parent of d is a.
- The children of c are $g, h,$ and i .
- The siblings of g are h and i.
- The ancestors of f are $d, a,$ and r.
- The descendants of a are $d, e,$ and f.
- The internal vertices are $r, a, d, c, g,$ and i .
- The leaves are $e, f, b, j, h, k,$ and l.
- The height of d is 1.
- The height of c is 2.
- The height of b is 0.
- The height of r is 3 which is the height of tree.

Properties of Trees

- **Theorem**: A tree with ***n*** vertices has $n - 1$ edges.

BASIS STEP: When $n = 1$, a tree with $n = 1$ vertex has no edges. It follows that the theorem is true for $n = 1$.

INDUCTIVE STEP: The inductive hypothesis states that every tree with k vertices has $k - 1$ edges, where k is a positive integer. Suppose that a tree T has $k + 1$ vertices and that v is a leaf of T (which must exist because the tree is finite), and let w be the parent of v. Removing from T the vertex v and the edge connecting w to v produces a tree T' with k vertices, because the resulting graph is still connected and has no simple circuits. By the inductive hypothesis, T' has $k - 1$ edges. It follows that T has k edges because it has one more edge than T', the edge connecting v and w. This completes the inductive step. ◁