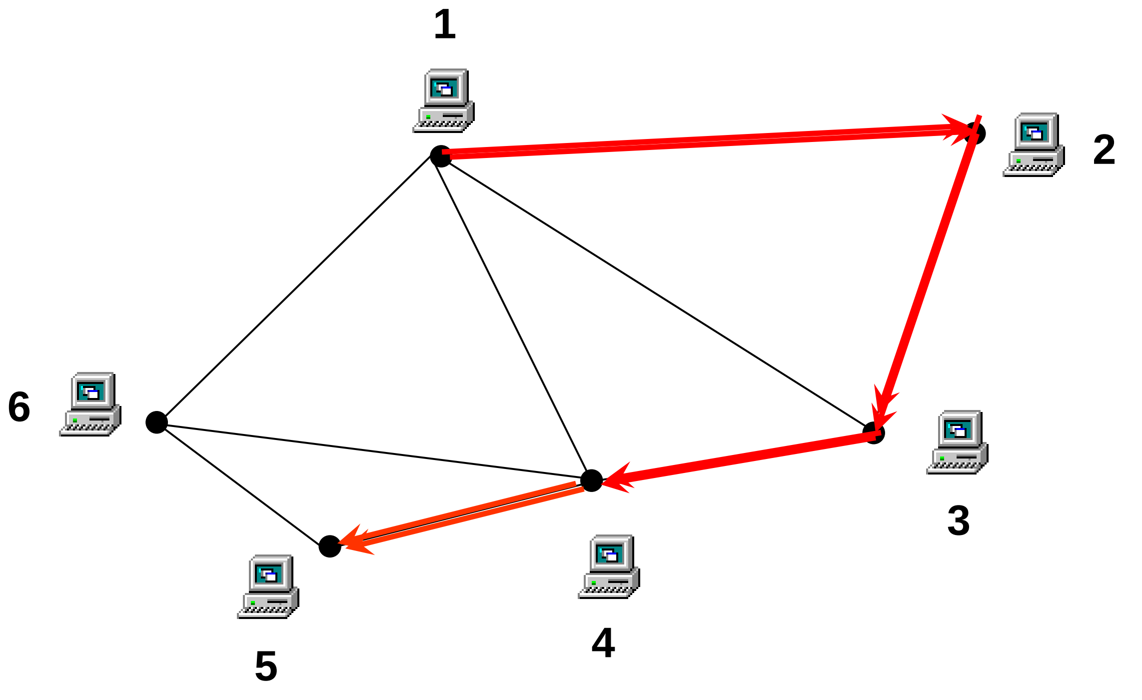
Lecture 11. Paths,Circuits&Trees

Ref.: K. H. Rosen, Chapter 8 & 9

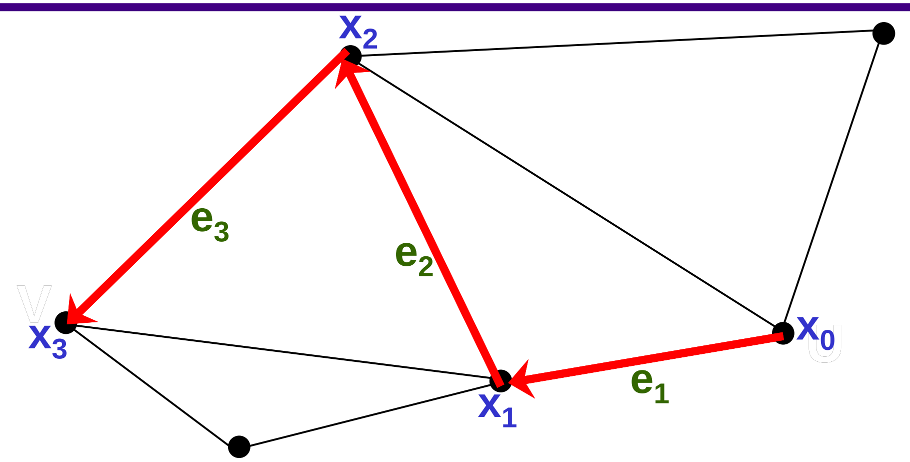
Paths and Circuits



Path from 1 to 5

{1,2},{2,3},{3,4}, {4,5}

Paths and Circuits



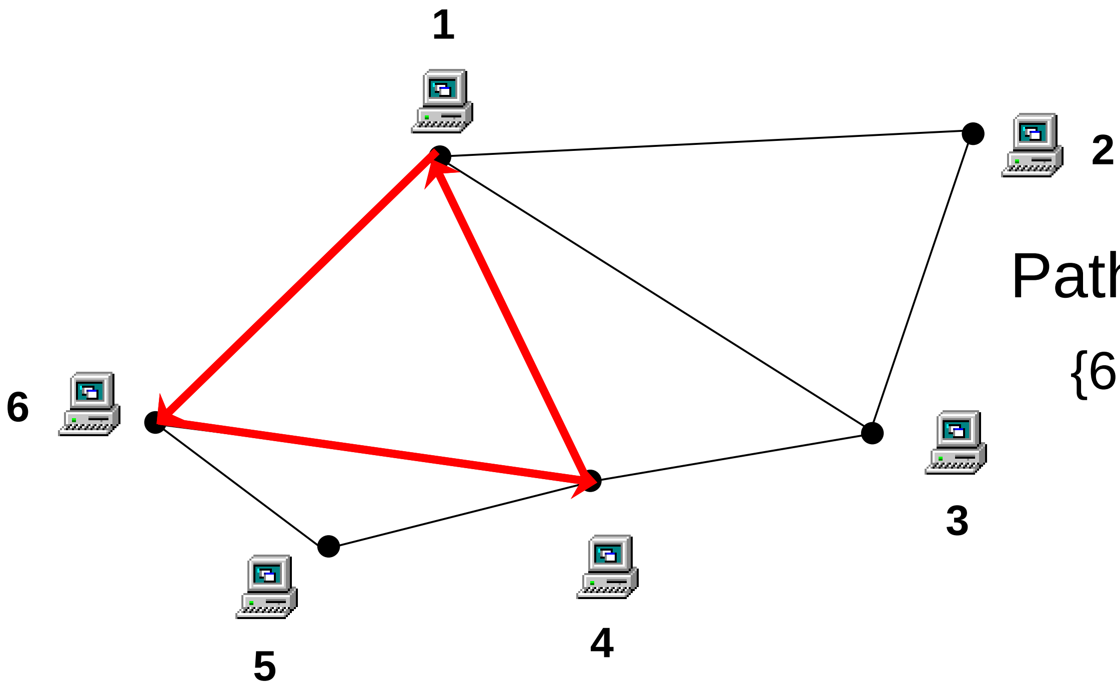
Definition

A <u>path of length *n*</u> from *u* to *v* where *n* is a positive integer in an <u>undirected graph</u> is the sequence of edges $e_1,\ldots,e_n$ of the graph such that $f(e_1) = \{x_0,x_1\}$, $f(e_2) = \{x_1,x_2\}$, … , $f(e_n) = \{x_{n-1},x_n\}$

Where $x_0 = u$ and $x_n = v$

When the graph is simple we denote this path by its vertex sequence $x_0,x_1,\ldots,x_n$

Paths and Circuits

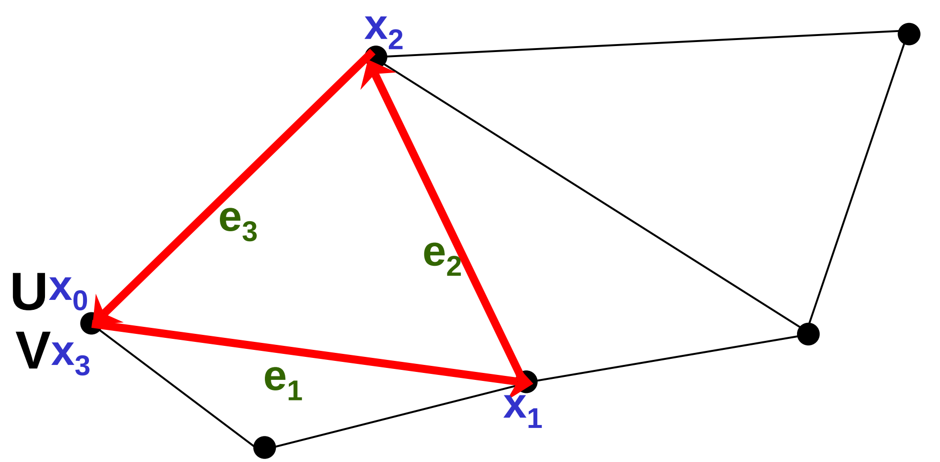


Path from 6 to 6

{6,4},{4,1},{1,6}

A path which begins and ends at the same vertex is called a circuit (or *cycle*)

Paths and Circuits



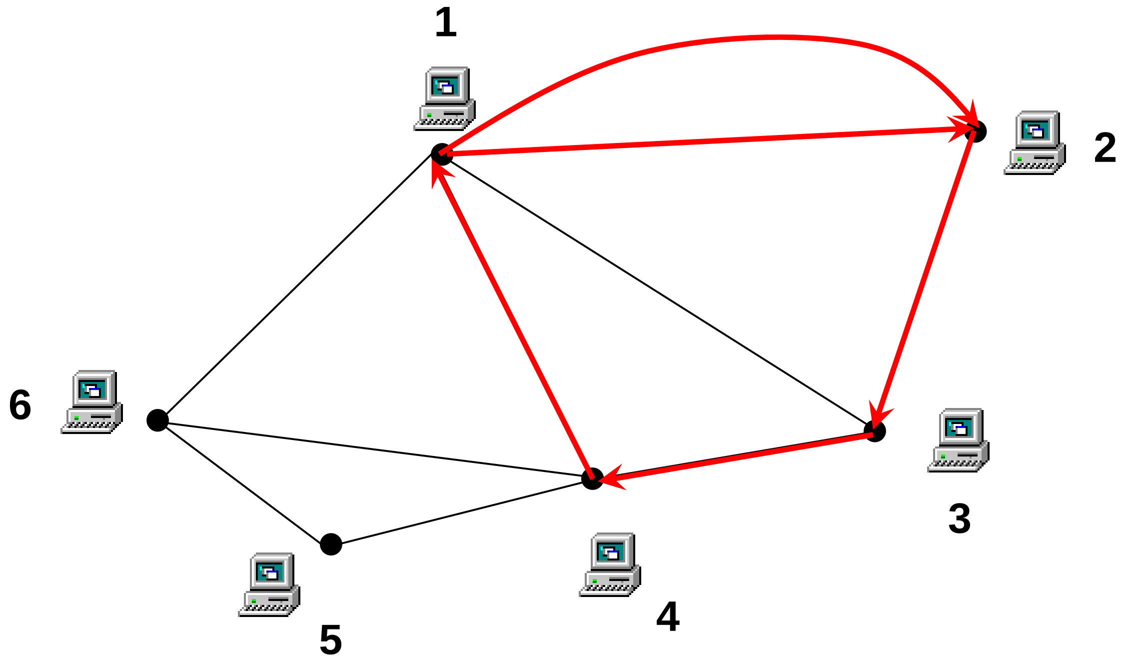
Definition

A **circuit** is a special case of a path such that

$f(e_1) = \{x_0,x_1\}$, $f(e_2) = \{x_1,x_2\}$, … , $f(e_n) = \{x_{n-1},x_n\}$

Where x_0=u and x_n=v and u=v

Paths and Circuits



A **path is simple** if it never traverses a single edge more than once.

This circuit from 1 to 1 is simple.

if we extend the circuit to be a path from 1 to 2, it would be not simple because the edge from {1, 2} is used **twice**.

Paths and Circuits

Definition

A *path* of length n from u to v where n is a positive integer in a directed multigraph is the sequence of edges $e_1, \ldots, e_n$ of the graph such that

$f(e_1) = (x_0, x_1)$, $f(e_2) = (x_1, x_2)$, … , $f(e_n) = (x_{n-1}, x_n)$

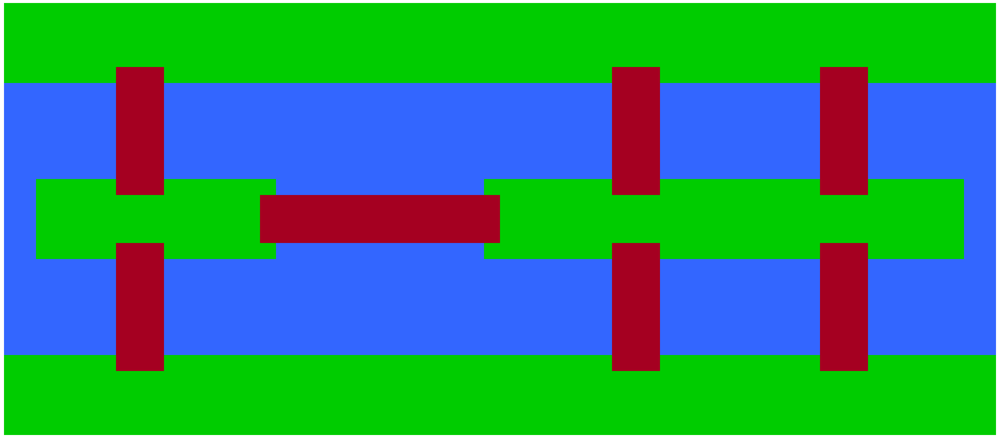
Where $x_0 = u$ and $x_n = v$

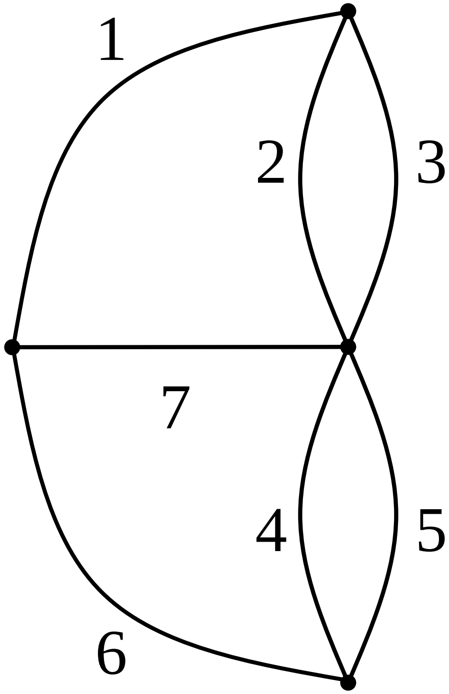
When there are no multiple edges in the graph, this path is denoted by its vertex sequence $x_0, x_1, \ldots, x_n$

A path that begins and ends at the same vertex is called a *circuit* or *cycle*.

A path or circuit is called ***simple*** if it does not contain the same edge more than once.

Euler Paths & Circuits





Is it possible to walk across every bridge without crossing any bridge more than once?

Euler Paths & Circuits

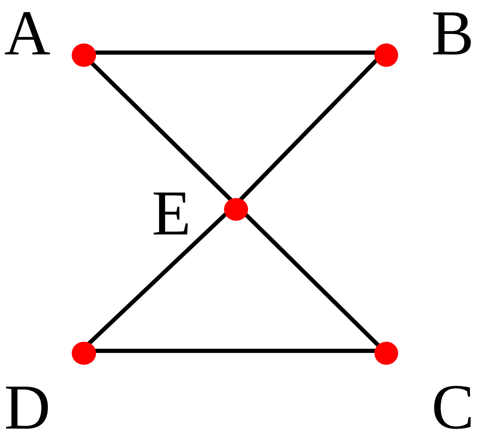
Definition:

An Euler CIRCUIT in a graph *G* is a simple circuit containing every edge of *G*.

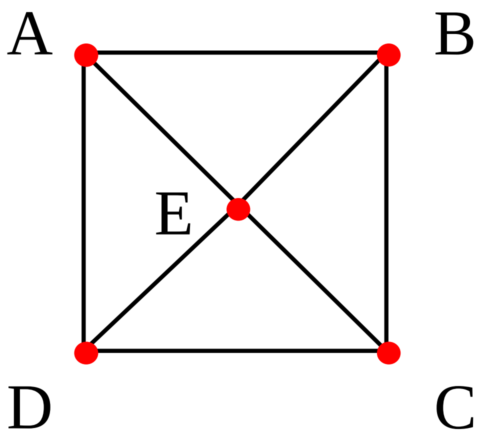
An Euler PATH in a graph *G* is a simple path containing every edge of G.

Euler Paths & Circuits

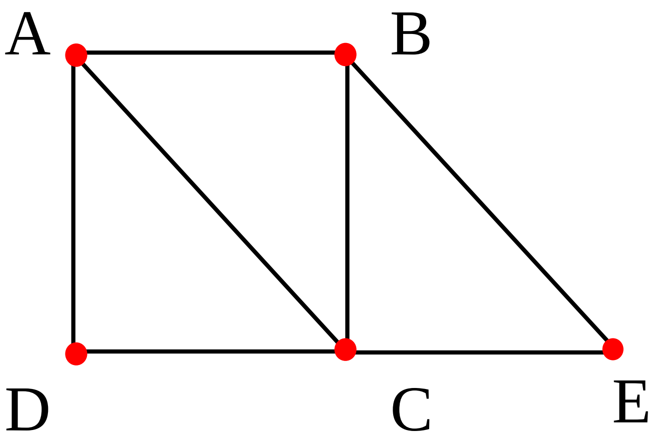
Which of the graphs below have a Euler CIRCUIT?
Of those that haven't, which have a Euler PATH?



G_1

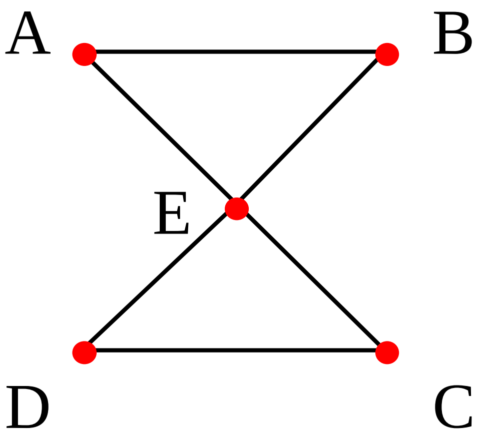


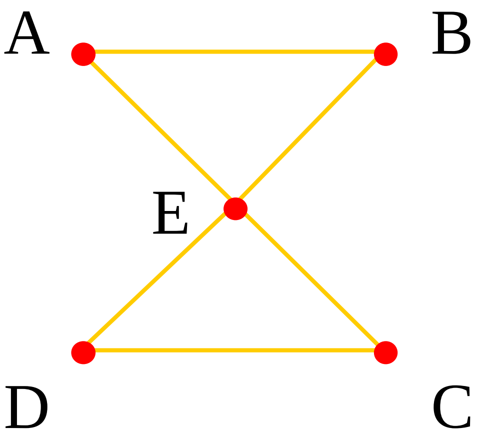
G_2



G_3

Euler Paths & Circuits

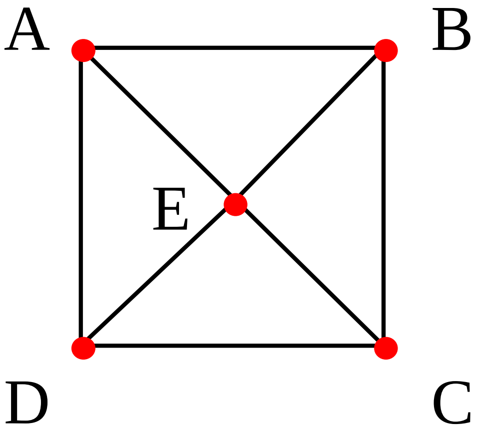




G_1 contains a Euler Circuit.
One solution is:
A, *E*, *C*, *D*, *E*, *B*, *A*.

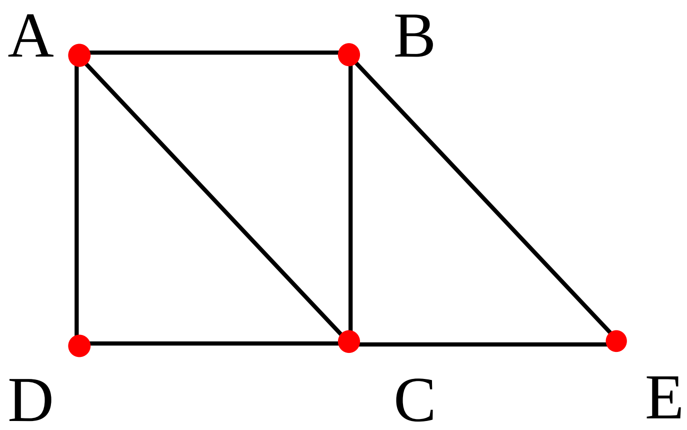
A is both the start and finish point - thereby satisfying the criteria for a Euler Circuit.

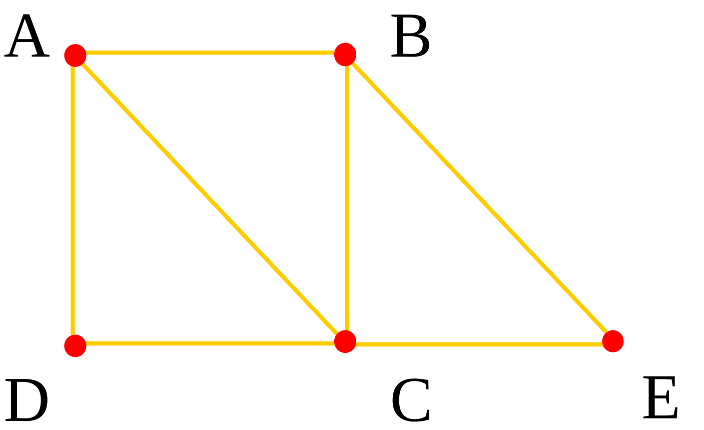
Euler Paths & Circuits



G_2 contains no Euler Circuit, nor does it contain a Euler Path.

Euler Paths & Circuits





G_3 does NOT contain a Euler Circuit.

One such Path is:
A, *D*, *C*, *E*, *B*, *C*, *A*, *B*.

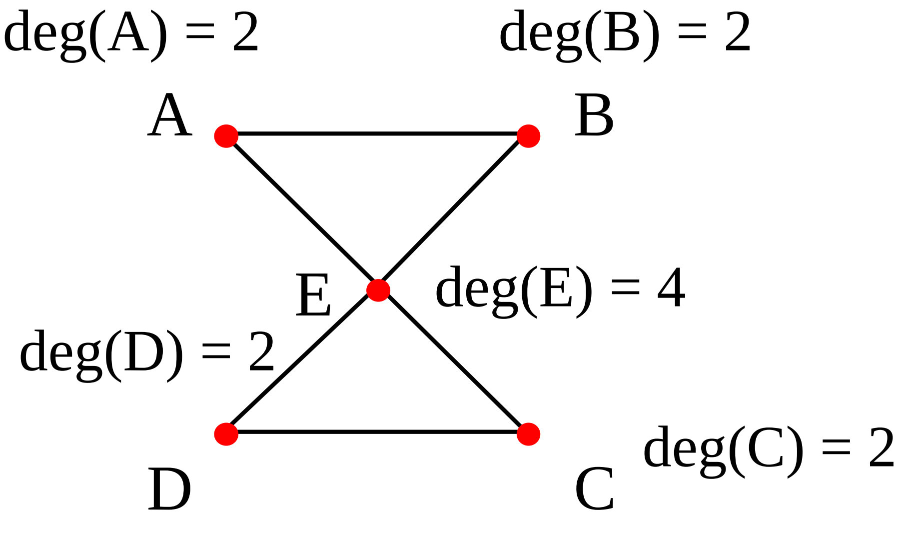
It DOES however contain a Euler Path.

A is the start, but unlike a Circuit a Path has a different end, namely *B*.

Euler Paths & Circuits

G_1 contains an Euler Circuit. What can be said about the degrees in G_1?

All degrees are even!



G_1 contains a Euler Circuit.
One solution is:
$A, E, C, D, E, B, A.$

Conditions for Euler Circuits

Theorem 1a:

Let *G* =(V,E) be a connected multigraph.

If *G* has an Euler Circuit, then every vertex of the graph has even degree.

Conditions for Euler Circuits

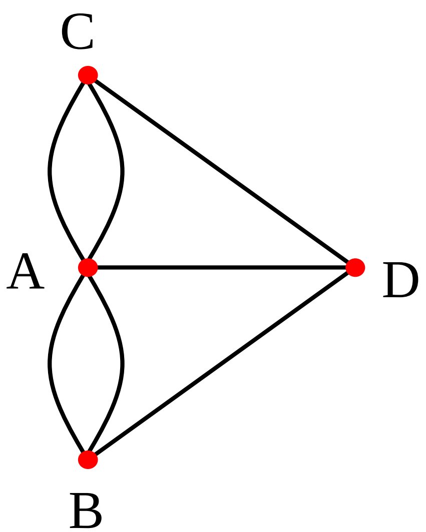
Contrapositive version of Theorem 1a:

Let G =(V,E) be a connected multigraph.

If some vertex of G has odd degree, then G does not have an Euler Circuit.

Conditions for Euler Circuits

We can now prove that Königsberg DOES NOT contain a Euler Circuit



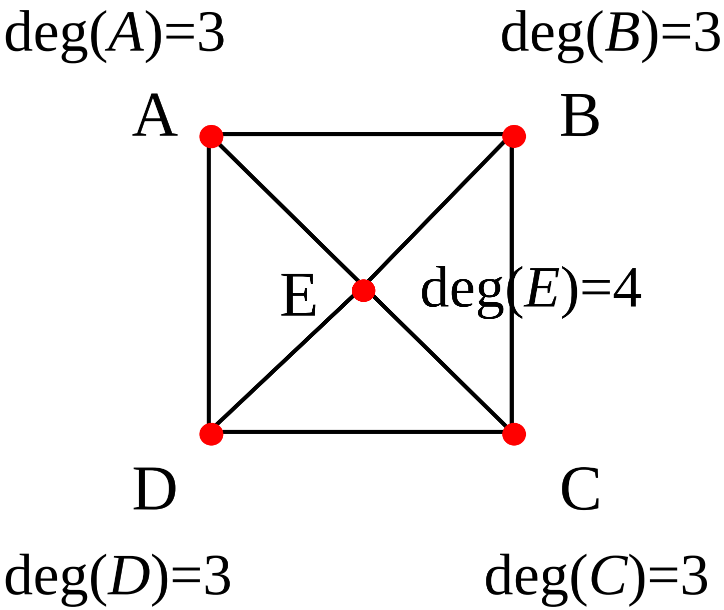
Königsberg Multigraph

deg(A) = 5
deg(B) = 3
deg(C) = 3
deg(D) = 3

Every Vertex in a connected Multigraph MUST have EVEN DEGREE to contain a Euler Circuit.

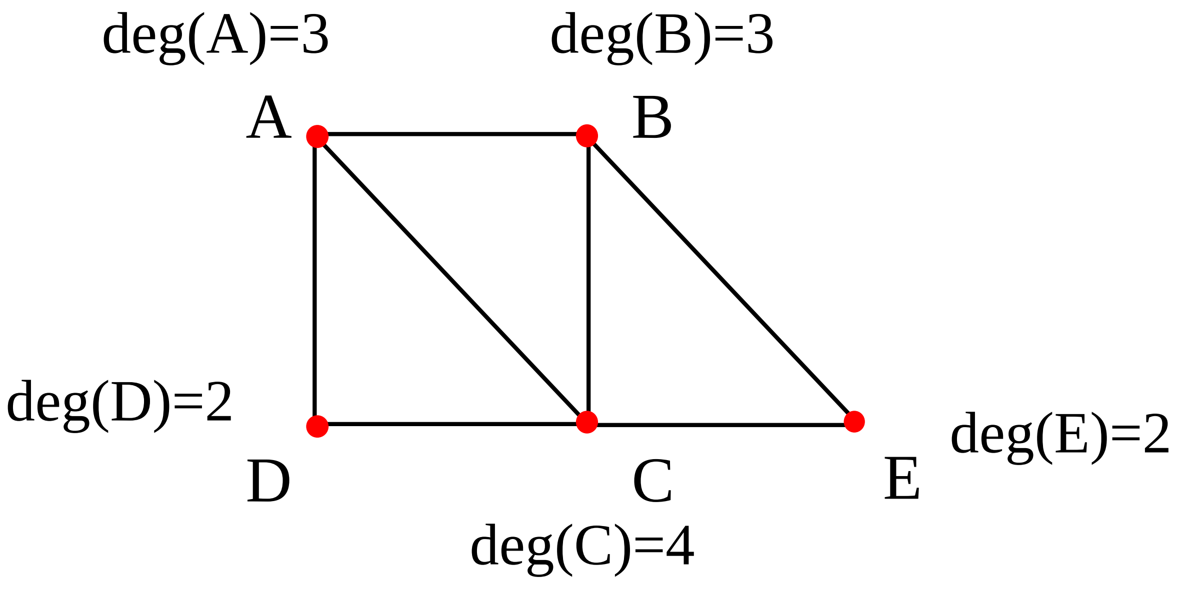
Therefore it is not possible to start at region in Königsberg, cross each bridge only once, and arrive back at the same region.

Conditions for Euler Circuits



G_2 contains no Euler Circuit !

Conditions for Euler Circuits



G_3 does NOT contain a Euler Circuit.

Conditions for Euler Circuits

<u>Theorem 1b:</u>

Let *G* =(V,E) be a connected multigraph.

If all vertices have even degree, then *G* has an Euler Circuit.

Conditions for Euler Circuits

Let G =(V,E) be a connected multigraph.

Theorem 1a:

If G has an Euler Circuit, then all vertices of the graph have even degree.

Theorem 1b:

If all vertices have even degree, then G has an Euler Circuit.

Theorem 1:

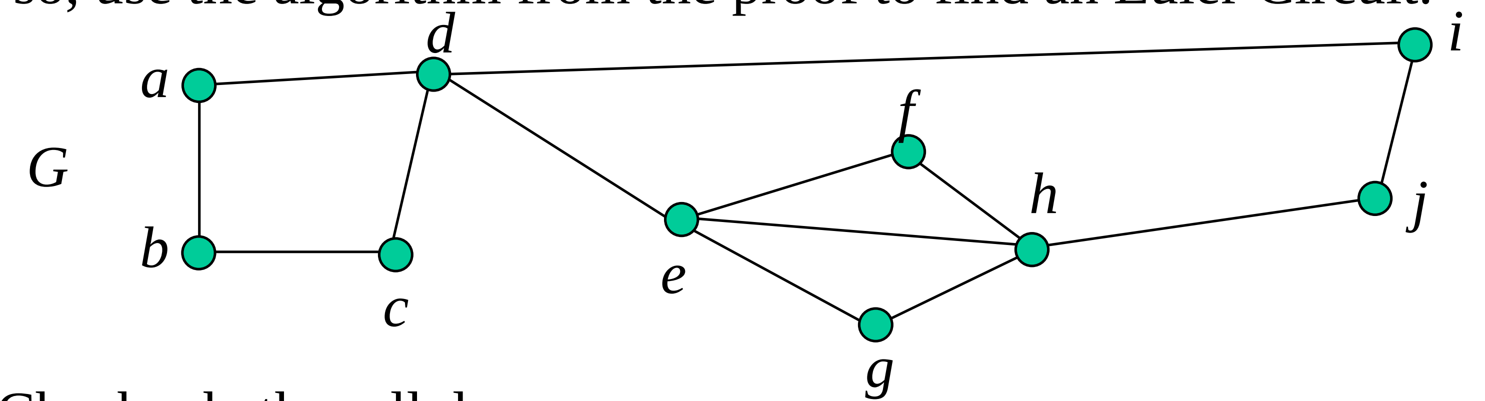
Let G =(V,E) be a connected multigraph.

G has an Euler Circuit if, and only if each of all vertices have even degree.

Conditions for Euler Circuits

Has G an Euler Circuit?

If so, use the algorithm from the proof to find an Euler Circuit.



Check whether all degrees are even:

$\deg(a) = \deg(b) = \deg(c) = \deg(f) = \deg(g) = \deg(i) = \deg(j) = 2$

$\deg(d) = \deg(e) = \deg(h) = 4$

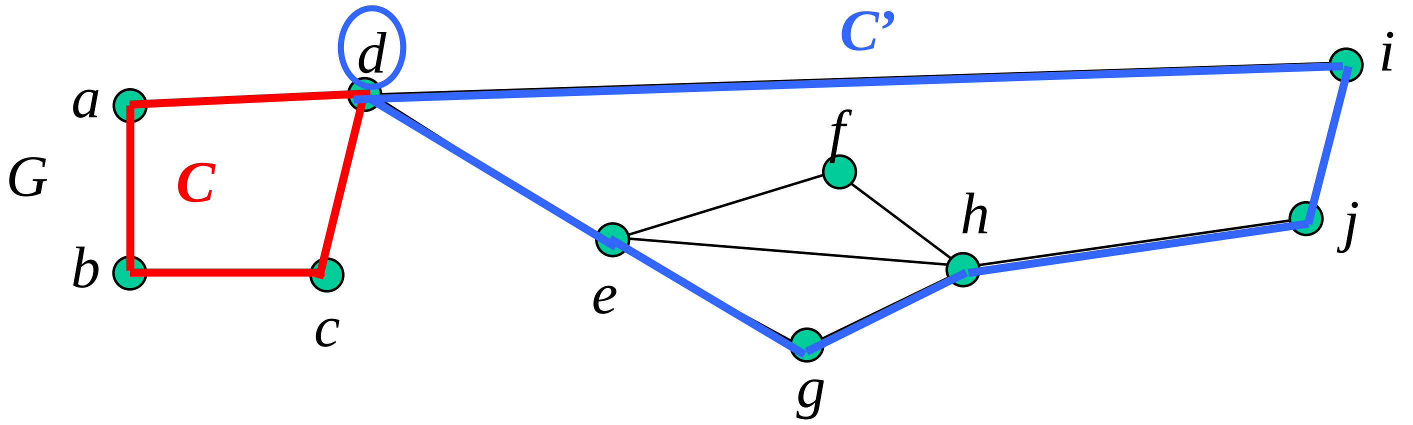
Hence all degrees are even and the graph is connected.

Thus, G has an Euler Circuit.

Conditions for Euler Circuits

Step 1: choose *a* as start vertex.

Step 2: create a circuit *C* which starts and ends in *a*: *abcda*



Step 3: *C* is not an Euler Circuit.

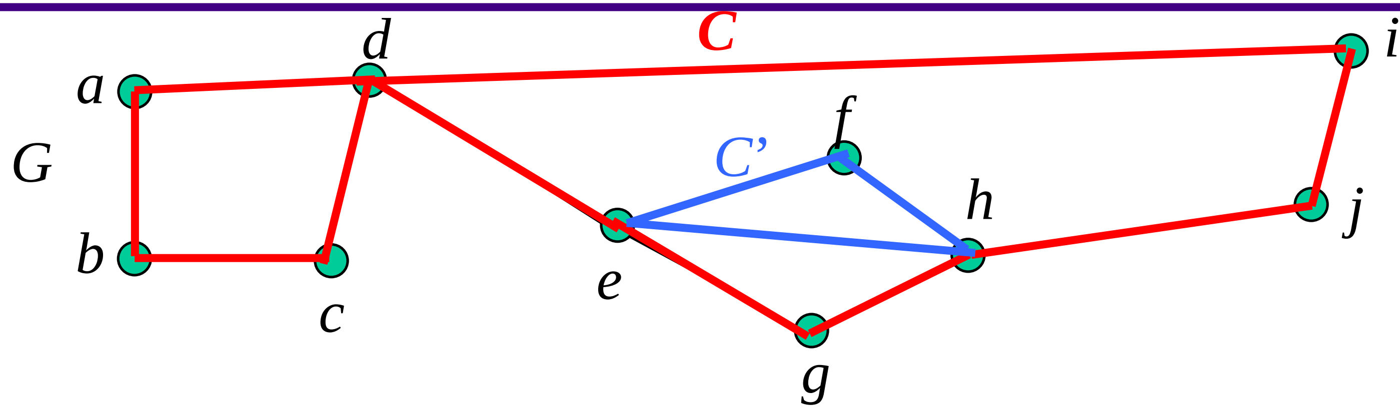
Step 3a: Remove *C* from *G* to build subgraph *G'*

Step 3b: Find a vertex which intersects *C* and *G'*: *d*

Step 3c: Create circuit *C'* which starts and ends in *d*: *deghjid*

Step 3d: Patch *C* and *C'* together to create *C''*: *abcdeghjida*

Conditions for Euler Circuits



Step 3e: Set C = C''. Back to Step 3

Step 3: *C* is not an Euler Circuit.

Step 3a: Remove *C* from *G* to build subgraph *G'*

Step 3b: Find a vertex which intersects *C* and *G'*: *e* or *h*. choose *e*

Step 3c: Create circuit *C'* which starts and ends in *e*: *efhe*

Step 3d: Patch *C* and *C'* together to create *C''*: *abcdefheghjida*

Step 3e: Set C = C''. Back to Step 3.

Step 3: C is a Euler Circuit.

Conditions for Euler Circuits

<u>Theorem 2:</u>

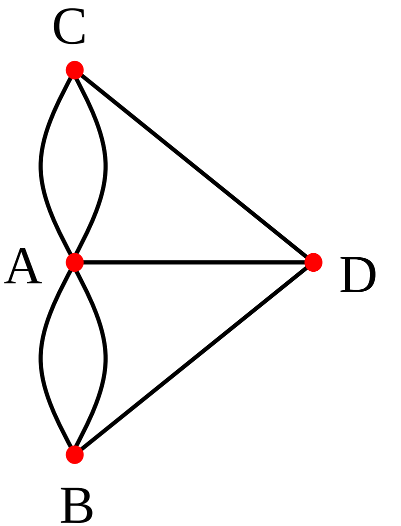
Let G =(V,E) be a connected multigraph.

G has a Euler Path but not an Euler Circuit if, and only if it has exactly two vertices of odd degree.

Conditions for Euler Circuits

Does Konigsberg contain a Euler Path?

A connected Multigraph MUST contain EXACTLY two vertices with odd degree for the graph to contain a Euler Path.



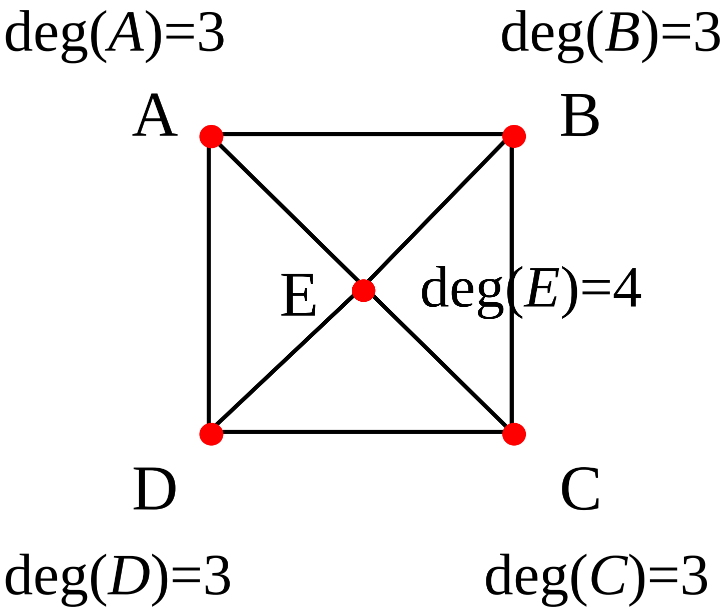
Konigsberg Multigraph

deg(A) = 5 deg(B) = 3

deg(C) = 3 deg(D) = 3

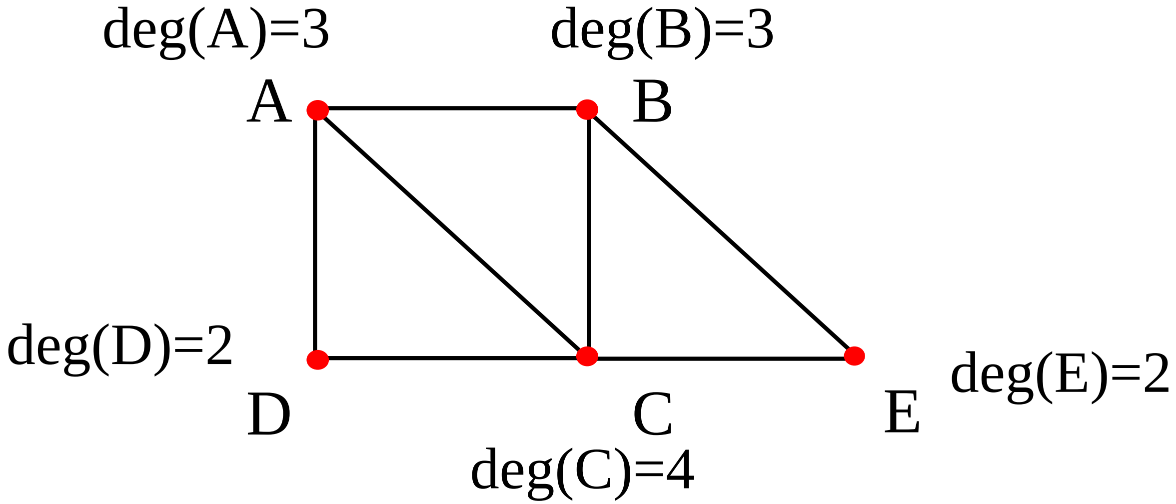
Konigsberg does not therefore contain a Euler Path.

Conditions for Euler Circuits



G_2 contains no Euler Circuit and no Euler Path !

Conditions for Euler Circuits



G_3 contains a Euler Path but not a Euler Circuit.

One Euler Path is: *A*, *D*, *C*, *E*, *B*, *C*, *A*, *B*.

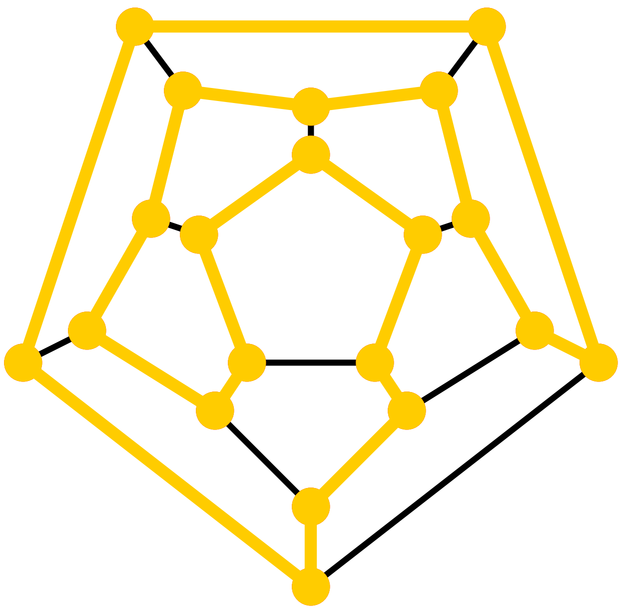
Start vertex *A* and terminal vertex *B*: have odd degree!

Each Euler Path, which is not a Euler Circuit starts with a vertex of odd degree and terminates at the other vertex of degree!

Hamilton Circuits

In Euler's Circuit, we answered the question:
Given a graph *G*, is it possible to find a circuit for *G* in which all **edges** of *G* appear exactly once?
A related question is this:
Given a graph *G*, is it possible to find a circuit for *G* in which all the **vertices** of *G* appear exactly once (except the first and the last)?
The terminology comes from a puzzle invented in 1857 by the Irish mathematician **Sir William Rowan Hamilton**.

Hamilton’ Puzzle



Puzzle consists of a dodecahedron (a polyhedron with 12 regular pentagons as faces).
Each of the 20 vertices is labelled with a name of a city - London, Paris, Hong Kong, New York, and so on
Problem: start at one city, tour the world by visiting each city exactly once and returning to the starting city.
Dodecahedron can be drawn in the plane as shown.
The circuit denoted with thick lines is one solution.

Hamilton Paths & Circuits

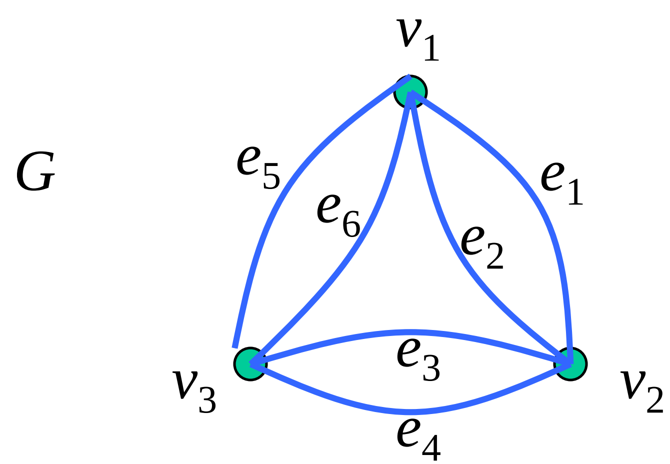
Definition:

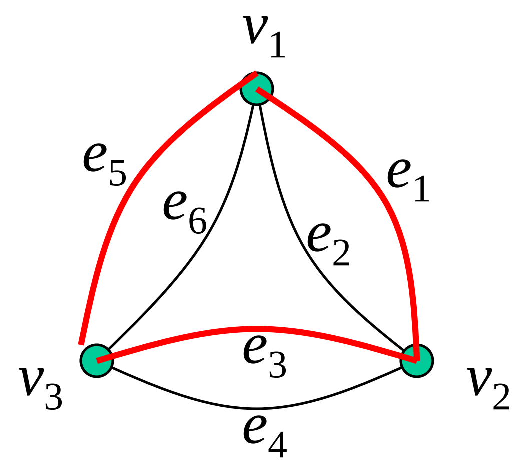
A path $x_0, x_1, \ldots, x_{n-1}, x_n$ in the graph $G = (V, E)$ is called a Hamilton path if $V = \{x_0, x_1, \ldots, x_{n-1}, x_n\}$ and $x_i \neq x_j$ for $0 \leq i < j \leq n$.

A circuit $x_0, x_1, \ldots, x_{n-1}, x_n, x_0$ (with $n \geq 2$) in a graph $G = (V, E)$ is called a Hamilton circuit if $x_0, x_1, \ldots, x_{n-1}, x_n$ is a Hamilton path.

Hamilton Paths & Circuits

Note that, while an Euler Circuit for a graph *G* must include every edge of *G*, it may visit some vertices more than once and hence may not be a Hamilton Circuit.
On the other hand, a Hamilton Circuit for *G* does not need to include all the edges of *G* and hence may not be a Euler Circuit.



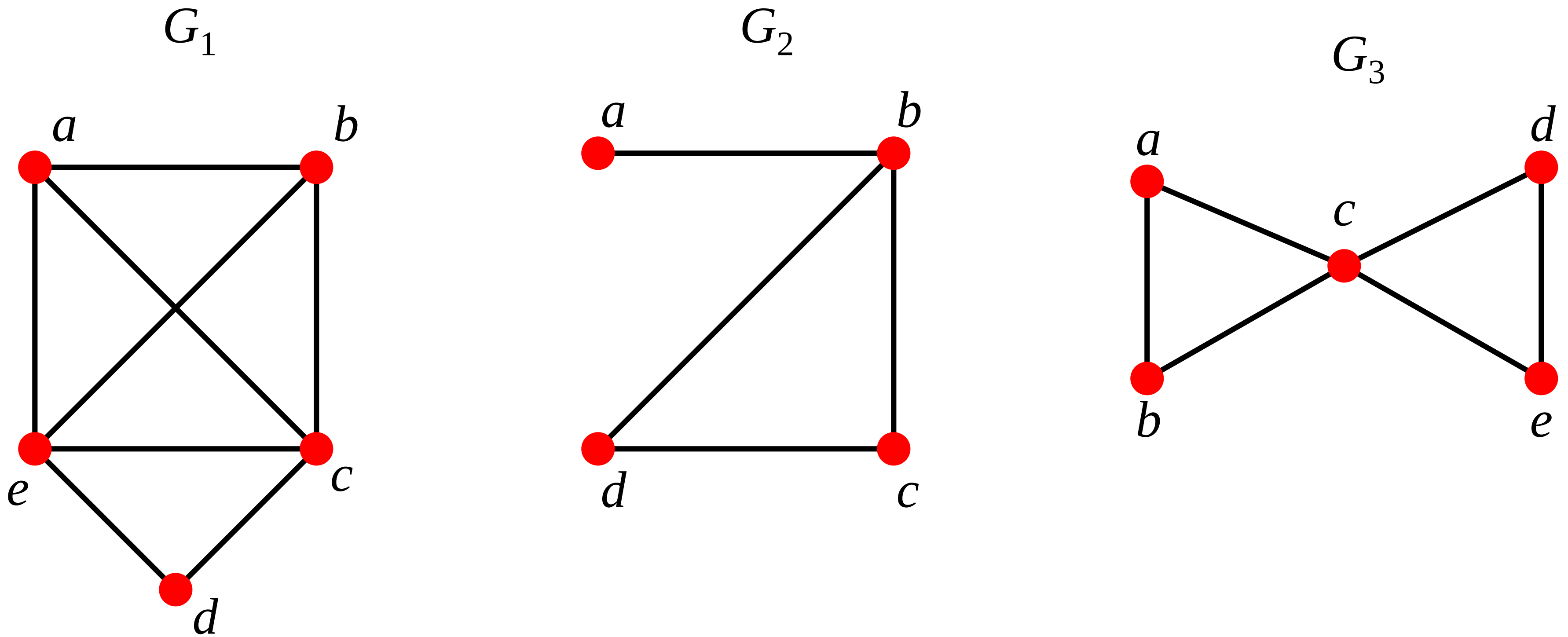


$v_1e_1v_2e_3v_3e_5v_1e_2v_2e_4v_3e_6v_1$ is an Euler Circuit but not a Hamilton Circuit

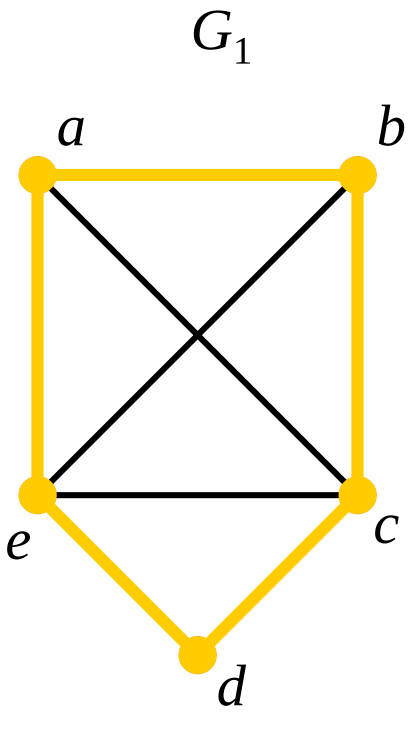
$v_1e_1v_2e_3v_3e_5v_1$ is a Hamilton Circuit but not an Euler Circuit

Hamilton Paths & Circuits

Which of these graphs have Hamilton Circuits? Or if not, Hamilton Paths?



Hamilton Paths & Circuits



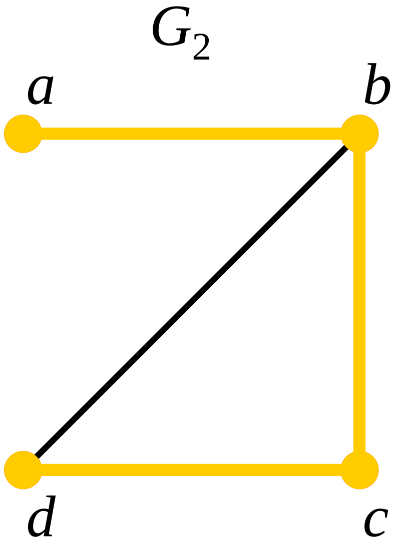
G_1 has a Hamilton circuit:

a, b, c, d, e, a for instance

Or c, a, b, e, d, c

Or d, e, b, a, c, d

Hamilton Paths & Circuits

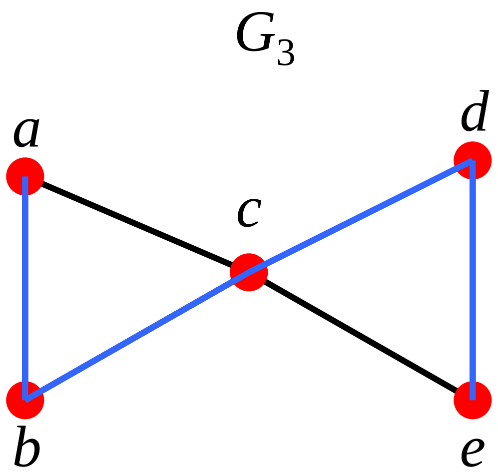


G_2 has no Hamilton circuit.

There is a Hamilton path though.

For instance: a, b, c, d

Hamilton Paths & Circuits



G_3 has no Hamilton Circuit

G_3 has a Hamilton Path: for instance: *a*, *b*, *c*, *d*, *e*

Conditions for Hamilton Circuits

There is a simple criterion to determine whether a given graph has an Euler Circuit.

Is there also a simple way to determine whether a graph has a Hamilton Circuit?

Unfortunately, there is no analogous criterion for determining whether or not a given graph has a Hamilton Circuit.

The best algorithms known for finding a Hamilton circuit in a graph or determining that no such circuit exists have exponential worst-case time complexity in the number of vertices of a graph (an NP-complete problem).

There are, however, certain properties that can be used in many cases to show that a graph has no Hamilton Circuit.

Conditions for Hamilton Circuits

<u>Theorem 1:</u>

Let *G*=(*V*, *E*) be a graph.

If *C* is a Hamilton Circuit of *G*, then every vertex of *G* is incident with exactly two distinct edges of *C*.

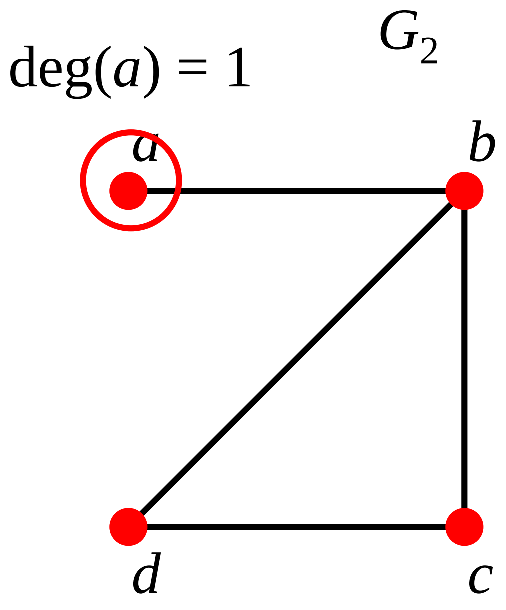
<u>Conclusions:</u>

(1) A graph with a vertex of degree 1 can not have a Hamilton Circuit.

(2) If a vertex in a graph has degree 2, then both edges that are incident with this vertex must be part of any Hamilton Circuit.

(3) When a Hamilton Circuit is being constructed and this circuit has passed through a vertex, then all remaining edges incident with this vertex, other than the two used can be removed from consideration.

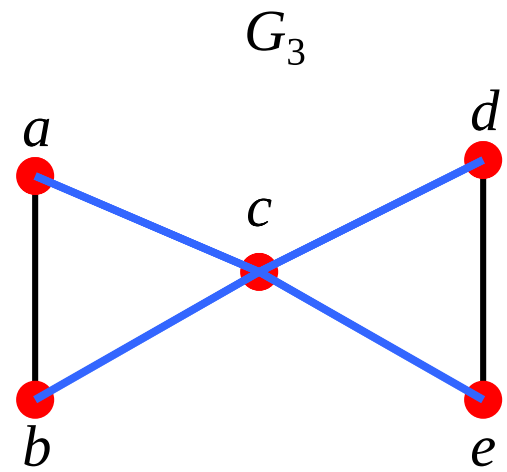
Conditions for Hamilton Circuits



G_2 has no Hamilton circuit, since deg(a) = 1

Conditions for Hamilton Circuits

G_3 has no Hamilton Circuit.



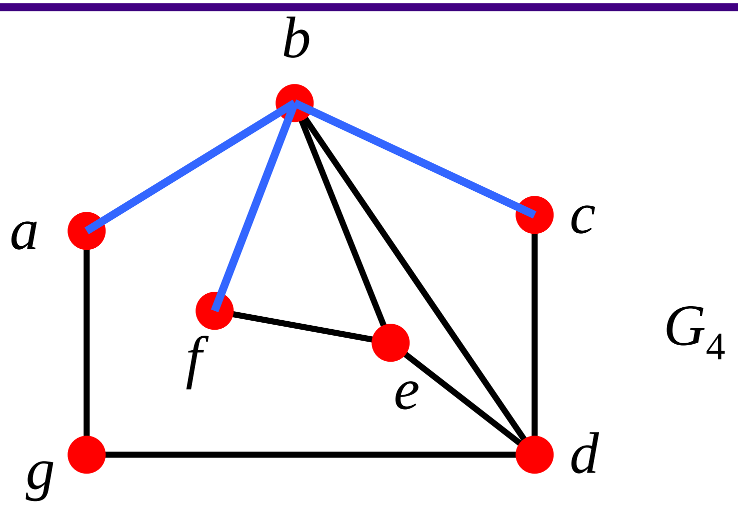
Suppose G_3 would have a Hamilton Circuit C.

Since deg(a) = deg(b) = deg(d) = deg(e) = 2, the edges $\{a,b\}$, $\{a,c\}$, $\{b,c\}$, $\{c,d\}$, $\{d,e\}$, $\{e,c\}$ must be part of C.

But then C contains 4 edges incident with c, i.e. $\{a,c\}$, $\{b,c\}$, $\{c,d\}$, $\{e,c\}$, which contradicts the fact that vertex c must have degree 2 in C.

Conditions for Hamilton Circuits

Has G_4 a Hamilton Circuit? No!



Suppose G_4 would have a Hamilton Circuit C .

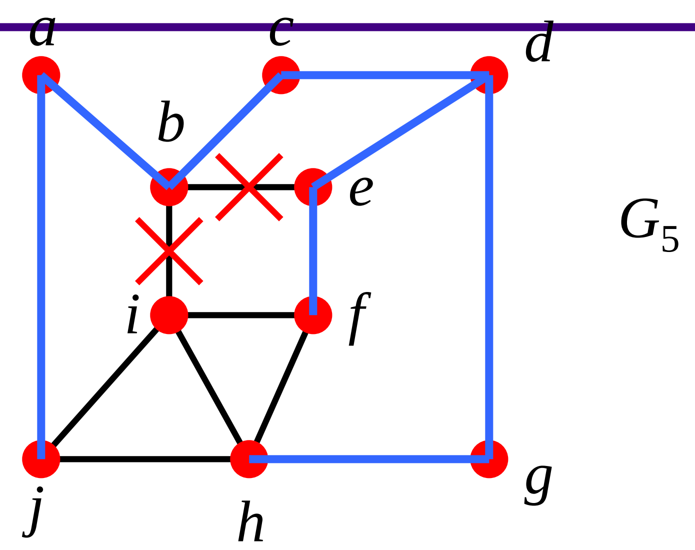
Since deg(a) = deg(c) = deg(f) = 2, the edges $\{a,b\}$, $\{a,g\}$, $\{b,c\}$, $\{c,d\}$, $\{b,f\}$, $\{f,e\}$ must be part of C.

But then C contains 3 edges incident with b, i.e. $\{a,b\}$, $\{b,c\}$, $\{b,f\}$, which contradicts the fact that vertex b must have degree 2 in C. .

Conditions for Hamilton Circuits

Has G_5 a Hamilton Circuit?

No!



Suppose G_5 would have a Hamilton Circuit C .

Since deg(a) = deg(c) = deg(g) = 2, the edges {a,b}, {a,j}, {b,c}, {c,d}, {d,g}, {g,h} must be part of C.

Since b is already incident with 2 edges in C, the other edges incident with b can not be part of C: {b,i} and {b,e}.

Since vertex e must have degree 2 in C the edges {e,f} and {e,d} must be part of C.

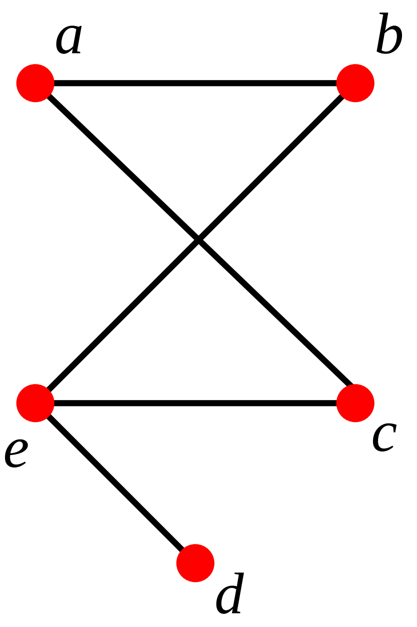
But then 3 edges incident with d must be part of C, which contradicts the fact that vertex d must have degree 2 in C.

Trees: Definition

- **A graph is called a <u>tree</u> if, and only if, it is circuit-free and connected.**
- **A <u>trivial tree</u> is a graph that consists of a single vertex, and an empty tree is a tree that does not have any vertices or edges.**
- **A graph is called a <u>forest</u> if, and only if, it contains no simple circuits.**

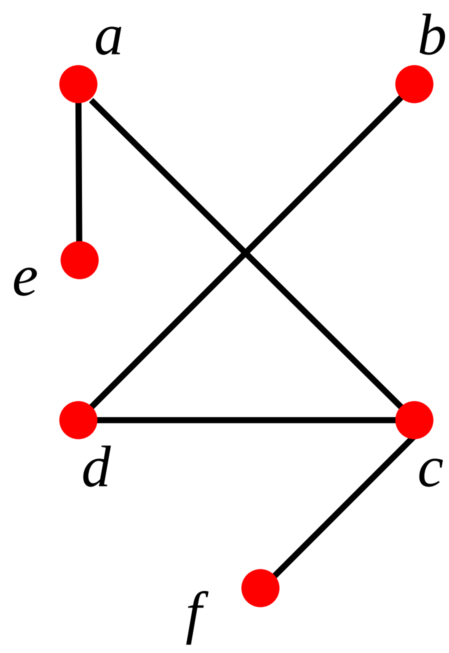
Example

G_1



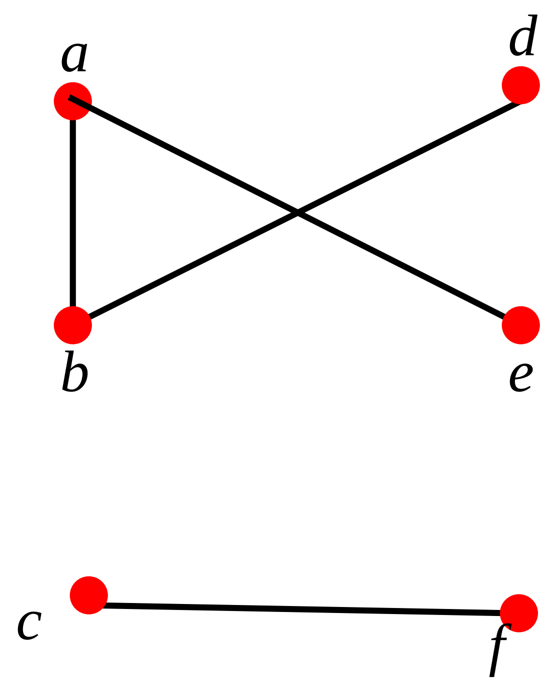
Not a tree

G_2



a tree

G_3

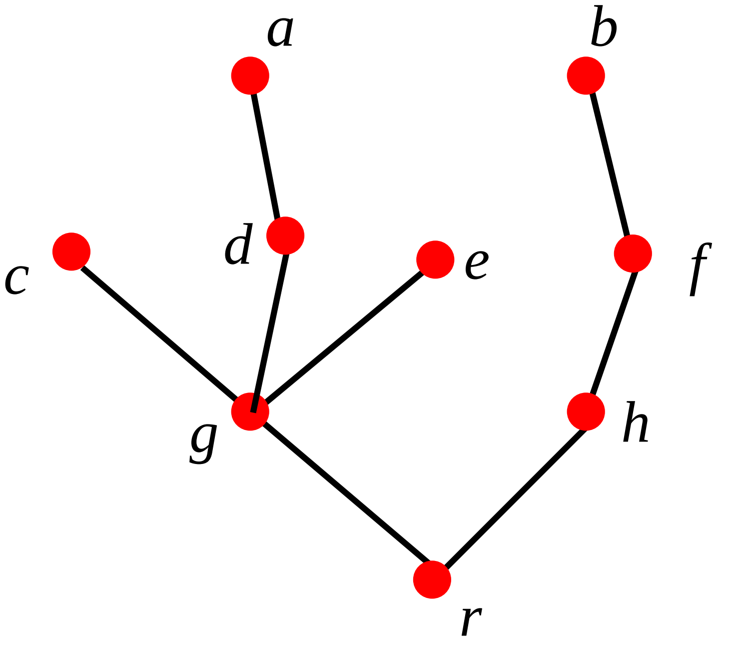


a forest

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.

Rooted Tree: example



Ordered Rooted Tree

A <u>ordered rooted tree</u> is a rooted tree where the children of each internal vertex are ordered.

Ordered rooted trees are drawn so that the children of each internal vertex are shown in order from left to right.

N-ary Tree

A rooted tree is called a <u>N-ary tree</u> if every internal vertex has no more than n children. The tree is called a full N-ary tree if every internal vertex has exactly n children.

Tree Properties

Theorem:

A tree with n vertices has n-1 edges

Theorem:

A full N-ary tree with *i* internal vertices contains $n=Ni+1$ vertices.

Binary Tree

- A **binary tree** is a rooted tree in which every internal vertex has at most two children. Each child in a binary tree is designated either a **left child** or a **right child** (but not both), and an internal vertex has at most one left and one right child.
- A **full binary tree** is a binary tree in which each internal vertex has exactly two children.

Binary Tree

- Given an internal vertex *v* of a binary tree *T*, the **left subtree** of *v* is the binary tree whose root is the left child of *v*, whose vertices consist of the left child of *v* and all its descendants, and whose edges consist of all those edges of *T* that connect the vertices of the left subtree together. The **right subtree** of *v* is defined analogously.

Binary Tree

- **Theorem:** If k is a positive integer and T is a full binary tree with k internal vertices, then T has a total of $2k + 1$ vertices and has $k + 1$ terminal vertices.
- **Theorem:** If T is a binary tree that has t terminal vertices and height h, then

$$t \leq 2h$$

Equivalently,

$$\log2\ t \leq h$$

Tree Traversal

Ordered rooted trees are often used to store information.

It is often need to visit each vertex of an ordered rooted tree to access data.

Procedures for systematically visiting each vertex of an ordered rooted tree are called traversal algorithms.

Tree Traversal

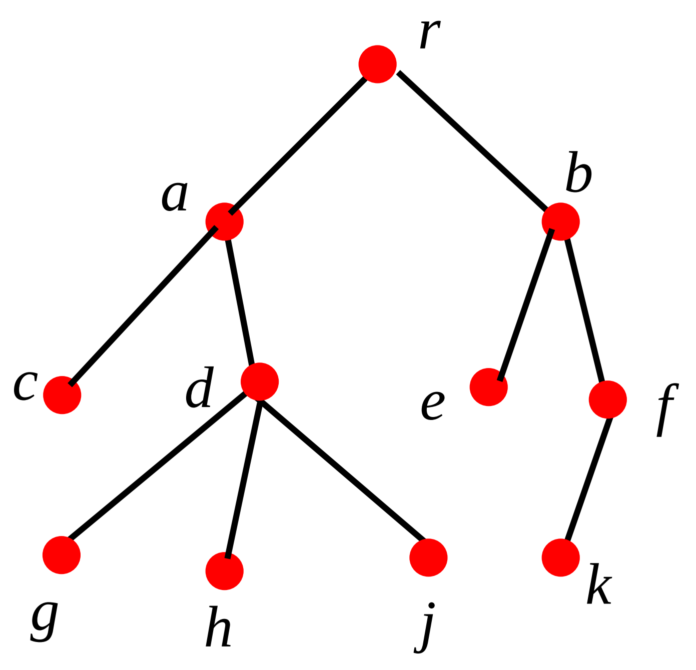
There are three most commonly used traversal algorithms:

- **Preorder traversal**
- **Inorder traversal**
- **Postorder traversal**

Preorder Traversal

Let T be an ordered rooted tree with root r. If T consists of only r, then r is the preorder traversal of T. Otherwise, suppose that T1, T2, ……Tn are the subtrees at r from lest to right in T. The preorder traversal of T begins by visiting r. It continues by traversing T1 in preorder, then T2 in preorder and so on, until Tn is traversed in preorder.

Preorder Traversal : Example



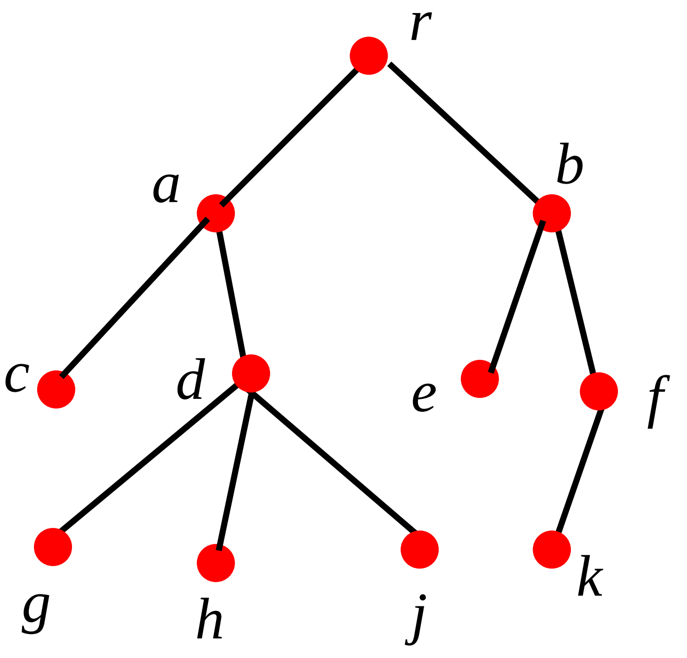
The preorder traversal of T:

racdghjbefk

Inorder Traversal

Let T be an ordered rooted tree with root r. If T consists of only r, then r is the inorder traversal of T. Otherwise, suppose that T1, T2, ……Tn are the subtrees at r from lest to right in T. The inorder traversal of T begins by traversing T1 in inorder, then visiting r. It continues by traversing T2 in inorder, then T3 in inorder and so on, until Tn is traversed in inorder.

Inorder Traversal : Example



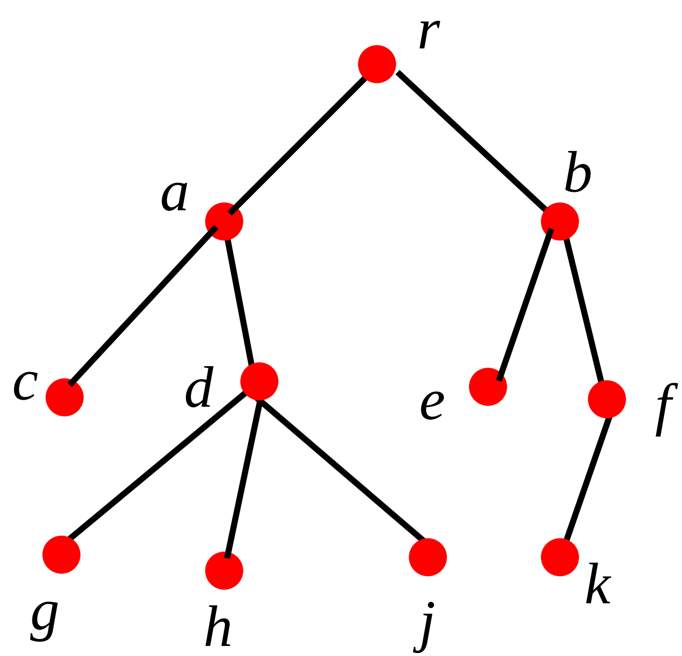
The inorder traversal of T:

cagdhjrebkf

Postorder Traversal

Let T be an ordered rooted tree with root r. If T consists of only r, then r is the postorder traversal of T. Otherwise, suppose that T1, T2, ……Tn are the subtrees at r from lest to right in T. The postorder traversal of T begins by traversing T1 in postorder, the T2 in postorder and so on, until Tn is traversed in postorder and ends by visiting r.

Postorder Traversal : Example



The postorder traversal of T:

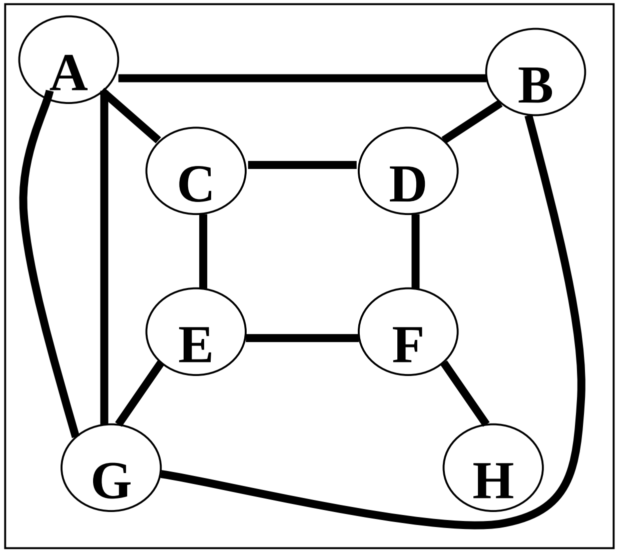
cghjdaekfbr

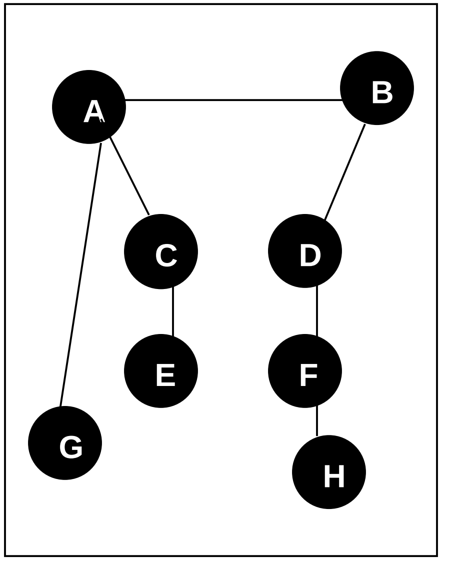
Spanning Trees

- A **spanning tree** for a graph *G* is a subgraph of *G* that contains every vertex of *G* and is a tree.
- **Proposition**:

1. Every connected graph has a spanning tree.
2. Any two spanning trees for a graph have the same number of edges.

Spanning Trees:BFS



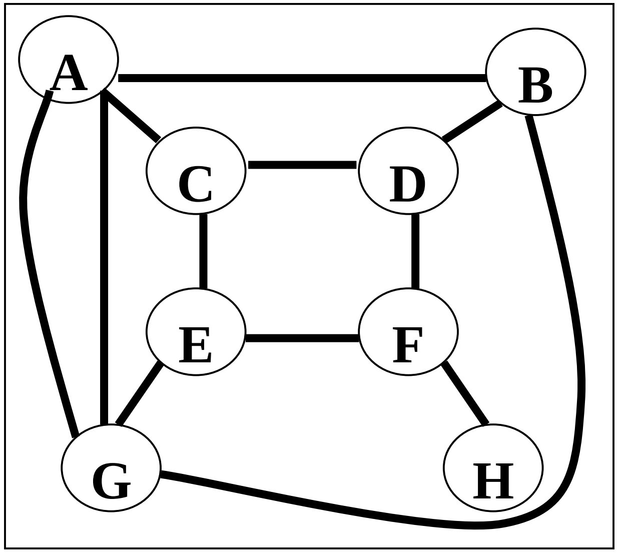


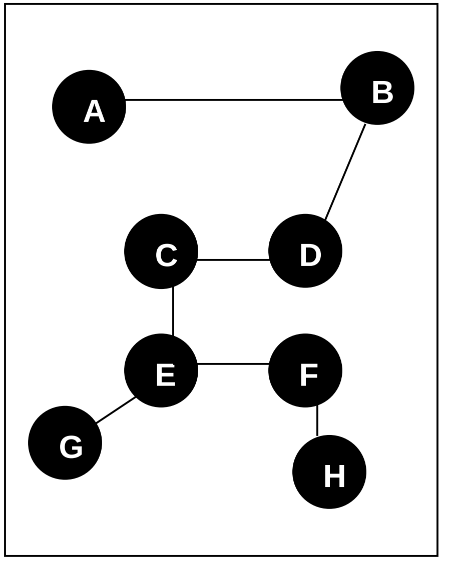
A graph with its spanning tree using BFS method

(typically "short and bushy")

- The idea of **breadth-first search (BFS)** is to process all the vertices on a given level before moving to the next level.

Spanning Trees: DFS





A graph with its spanning tree using DFS method

(typically “long and stringy”)

- The idea of **Depth-first search (DFS)** is to explore along each branch as far as possible before backtracking

Summary

- **Paths and Circuits**
- **Euler Path and Circuits**
- **Hamilton Path and Circuits**
- **Trees**