Chapter 9: Graphs

9.1 Graphs and Graph Model

9.2 Graph Terminology and Special Types of Graphs

9.3 Representing Graphs and graph Isomorphism

9.4 Connectivity

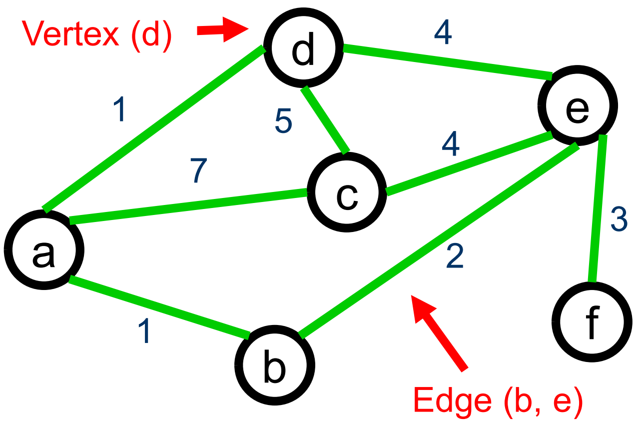
Dr Patrick Chan
School of Computer Science and Engineering
South China University of Technology

Agenda

- Graph
- Terminology
- Connectivity
- Isomorphism

Graph

- A graph **G = (V, E)** consists of a set of vertices **V**, and a set of edges **E**



V = { a, b, c, d, e, f }

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

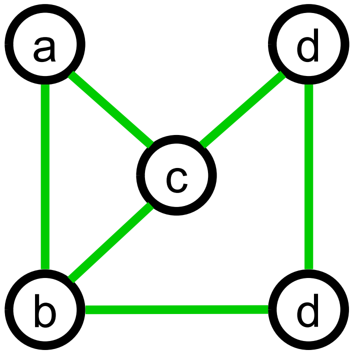
- Vertices (V)
  - |V|: the number of vertices
- Edges (E)
  - Sometimes referred as arc
  - Connection between a pair of vertices (v, w), where v and w belong to V
  - |E|: the number of edges
  - Weight may be included

Graph Structure

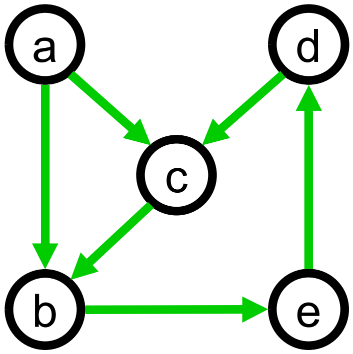
- Key questions about Graph Structure
  - Directed / Undirected Edge?
  - Single / Multiple Connection?
  - Loop?

Directed/Undirected?

- **Undirected Graph**
  - Edges are not directed
  - If (a,b), then (b,a)

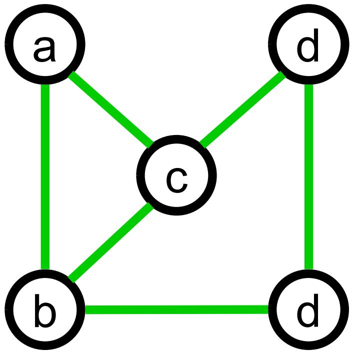


- **Directed Graph (Digraph)**
  - Edges are directed
  - (a,b) does not mean (b,a)

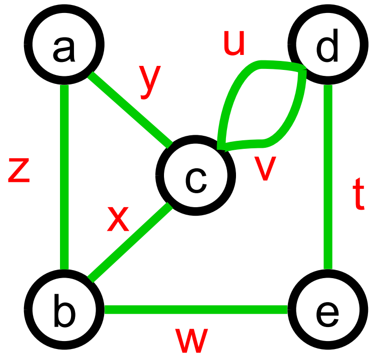


Single/Multiple Connection?

- **Simple Graph**
  - No two edges connects the same pair of vertices
  - Loop is not allowed

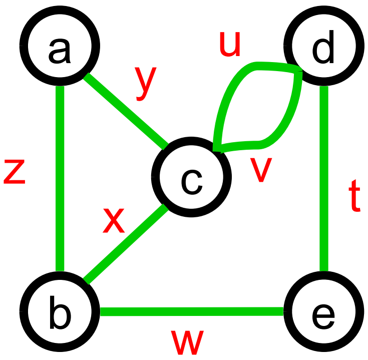


- **Multigraph**
  - Two vertices may be connected by more than one edges
  - An edge cannot be identified uniquely by a pair of vertices
    - Additional name is needed
    - E.g. (c,d) means u or v

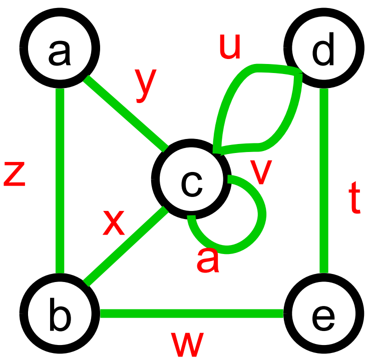


Loop?

- Multigraph does not allow loop



- Pseudograph is a special multigraph allows loop



- Sometimes, the meanings of Pseudograph and Multigraph are the same

Summary

Undirected		No Loop	Loop
	Single Edge	Simple Graph	/
	Multiple Edge	Multigraph	Pesudograph (Multigraph)

Directed		No Loop	Loop
	Single Edge	Simple Directed Graph	/
	Multiple Edge	Directed Multigraph	Mixed Graph

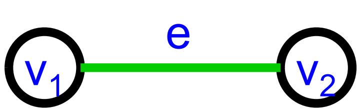
Adjacent / Neighbor

- **Undirected graph**
  - Let (v_1, v_2) is an edge
  - v_1 and v_2 are endpoints
  - v_1 is adjacent to v_2
  - Also mean "v_2 is adjacent to v_1" since $(v_1, v_2) = (v_2, v_1)$

- **Directed graph**
  - Let (v_1, v_2) is an edge
  - v_1 is initial vertex
  - v_2 is terminal (end) vertex
  - v_1 is adjacent to v_2
  - v_2 is adjacent from v_1
  - Do not mean "v_2 is adjacent to v_1"

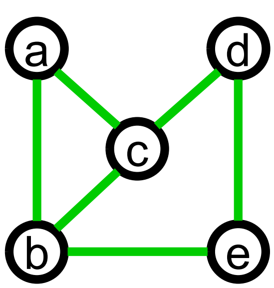
Adjacent / Neighbor

- e incidents with v_1 and v_2
- e connects v_1 and v_2

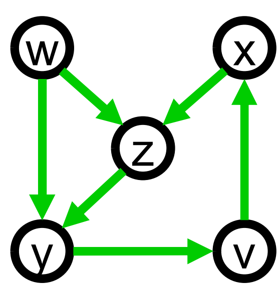


- Example:

a & b are adjacent
b & a are adjacent

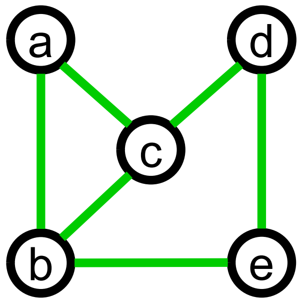


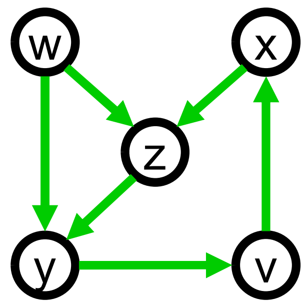
w is adjacent to z
z is not adjacent to w



Neighbor Set

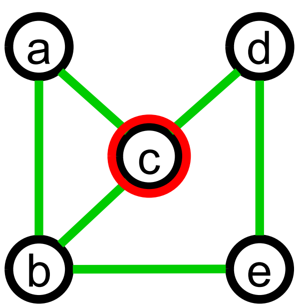
- **Neighbor Set N(v)** contains all adjacent vertices of v
- For example: N(c) = {a,b,d}
  N(z) = {y}

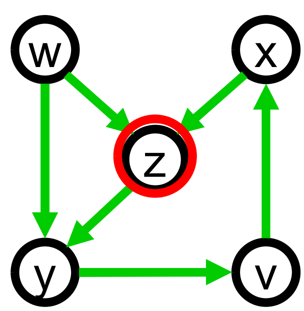




Degree

- **Undirected graph**
  - Degree: number of edges containing that vertex (Adjacent vertex number)
  - Isolated vertex: deg = 0
  - Pendant vertex: deg = 1
  - E.g. deg(c) = 3
- **Directed graph**
  - In-Degree: in-bound edge number
  - Out-Degree: out-bound edge number
  - E.g. $deg^-(z)$ = 2
    $deg^+(z)$ = 1

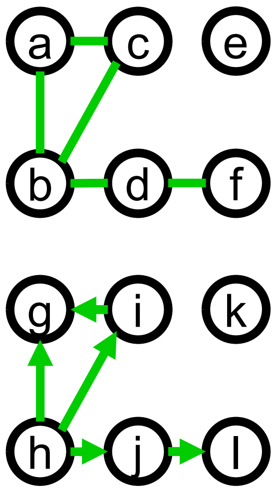




Degree: Example

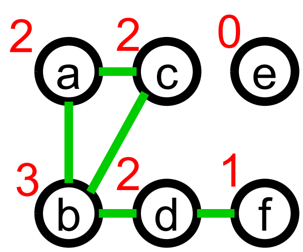
- What are the degrees of the following vertices?

deg(a) = 2	$\deg^-(g)$ = 2	$\deg^+(g)$ = 0
deg(b) = 3	$\deg^-(h)$ = 0	$\deg^+(h)$ = 3
deg(c) = 2	$\deg^-(i)$ = 1	$\deg^+(i)$ = 1
deg(d) = 2	$\deg^-(j)$ = 1	$\deg^+(j)$ = 1
deg(e) = 0	$\deg^-(k)$ = 0	$\deg^+(k)$ = 0
deg(f) = 1	$\deg^-(l)$ = 1	$\deg^+(l)$ = 0



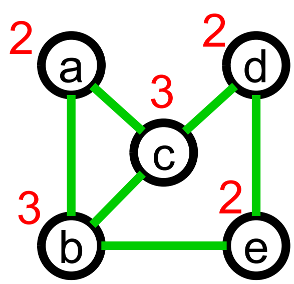
Degree Sequence

- A degree sequence is a monotonic nonincreasing sequence of the degrees of vertices in an undirected graph.



(2,3,2,2,0,1) Not monotonic nonincreasing

(3,2,2,2,1,0) Degree sequence



(3,3,2,2,2) Degree sequence

Degree

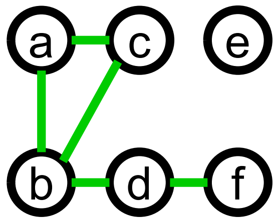
Handshaking Theorem 1

- For any undirected graph G = (V, E),

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

  - Twice number of edges = sum of degrees

- Each edge maps to two vertices (start & end)
- It also applies to multiple edges and loop



|E| = 5

deg(a) = 2
deg(b) = 3
deg(c) = 2
deg(d) = 2
deg(e) = 0
deg(f) = 1

Degree: Handshaking Theorem 1

Example 1

- How many edges are there in a graph with 10 vertices each of degree six?

- Total degree = 10 x 6 = 60
- According to Handshaking Theorem

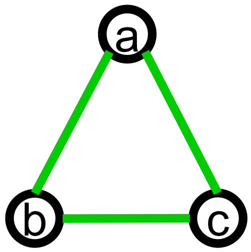
$$2|E| = \sum_{v \in V} \deg(v)$$

- |E| = 60/2 = 30

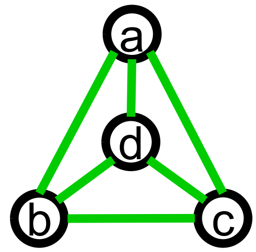
Degree: Handshaking Theorem 1

Example 1

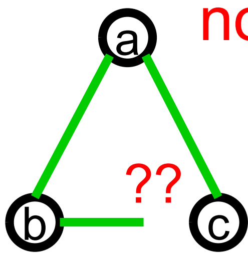
- Is there a graph with degree sequence…
  - (2,2,2)? Yes



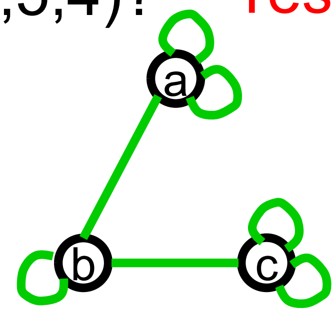
  - (3,3,3,3)? Yes



  - (2,2,1)? No, not even



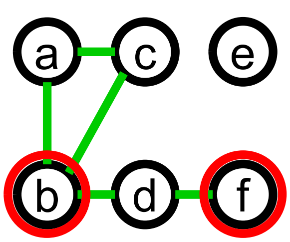
  - (5,5,4)? Yes



Degree

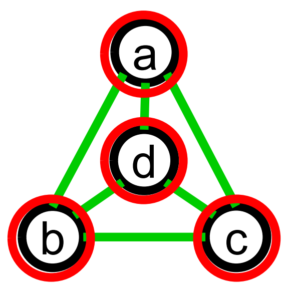
Handshaking Theorem 2

- Undirected graph has an even number of vertices of odd degree



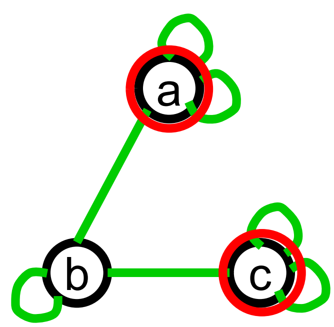
vertices of odd degree

2



vertices of odd degree

4



vertices of odd degree

2

Handshaking Theorem 2

- **Proof**
  - Let V_o and V_e be the set of vertices of odd and even degree

$$\underbrace{2|E|}_{\text{even}} = \sum_{v \in V} \deg(v) = \underbrace{\sum_{v \in V_o} \deg(v)}_{\text{also be even}} + \underbrace{\sum_{v \in V_e} \deg(v)}_{\text{Must be even}}$$

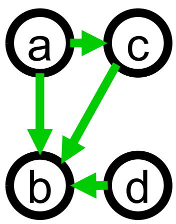
  - As summation of even degree (2nd term) is even
  - Summation of odd degree (1st term) is also even
    - As $\deg(v)$ is odd for $v \in V_o$
    - The number of $\deg(v)$ must be even for $v \in V_o$

Handshaking Theorem 3

- For any directed graph G = (V, E),

$$|E| = \sum_{v \in V} deg^{+}(v) = \sum_{v \in V} deg^{-}(v)$$

- Each edge maps to one initial and on end vertices



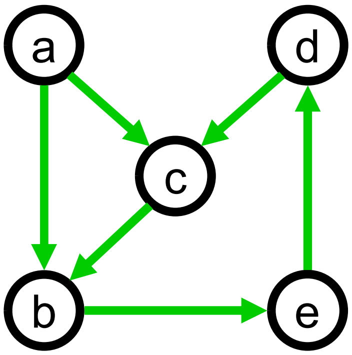
$|E| = 4$

$deg^{+}(a) = 2$ $deg^{-}(a) = 0$
$deg^{+}(b) = 0$ $deg^{-}(b) = 3$
$deg^{+}(c) = 1$ $deg^{-}(c) = 1$
$deg^{+}(d) = 1$ $deg^{-}(d) = 0$

- It also applies to multiple edges and loop

Path

- A sequence of vertices $v_1, v_2, \ldots, v_n$ of length *n*-1 with an edge from v_i to v_{i+1} for $1 \le i < n$
- A path is **simple** if all vertices on the path are distinct



Vertex a to e

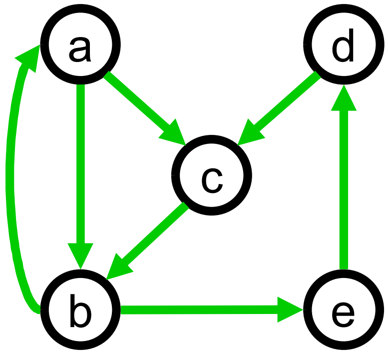
a > b > e	Length = 2	Simple
a > c > b > e	Length = 3	Simple

Vertex a to b

a > b	Length = 1	Simple
a > c > b	Length = 2	Simple
a > c > b > e > d > c > b	Length = 6	Not Simple

Cycle (Circuit)

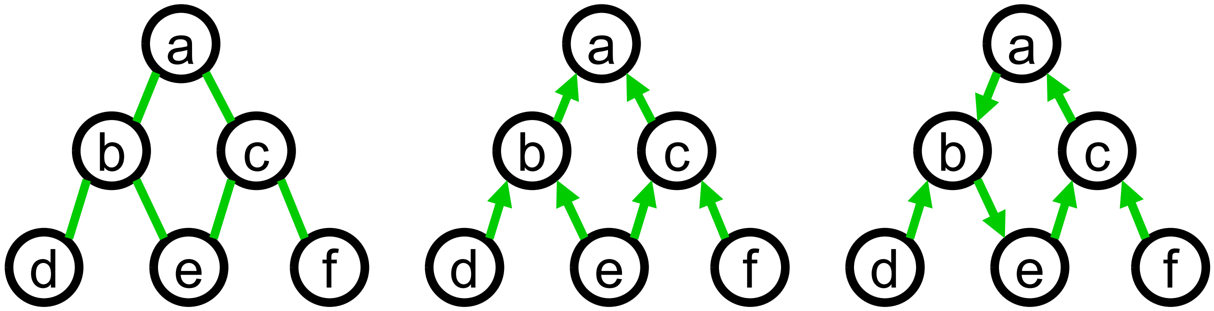
- A path connects v_i to itself
- A cycle is **simple** if the path is simple, except the first and last vertices are the same



a > c > b > a	Simple Cycle
b > e > d > c > b	Simple Cycle
b > e > d > c > b > a > c > b	Cycle

Acyclic

- A graph without cycle is called **acyclic**
- A directed graph without cycles is called a **Directed Acyclic Graph (DAG)**



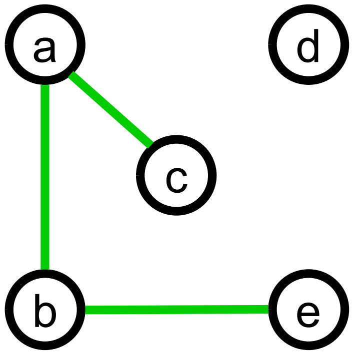
Not a
Undirected Acyclic

DAG

Not a DAG

Connectedness

- Vertices v, w are connected if and only if there is a path starting at v and ending at w
- Every graph consists of separate connected pieces called connected components



Are a and e connected? Yes

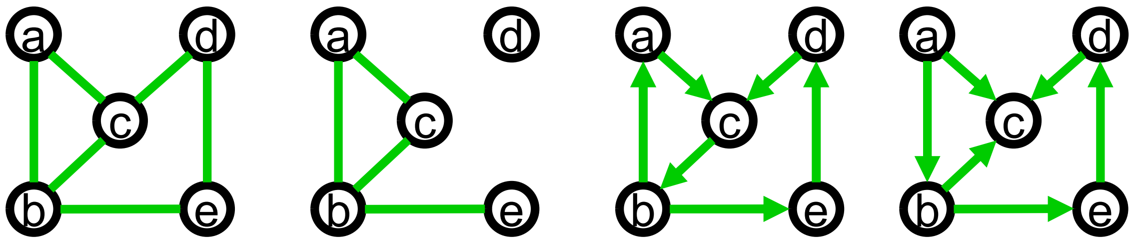
Are a and d connected? No

How many connected components?

2

Connectedness

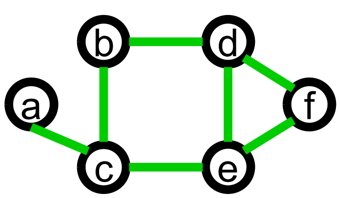
- **Undirected graph**
  - Connected: if there is at least one path from any vertex to any other (Only one connected component)
- **Directed graph**
  - Weakly connected: Directed graph without considering directions is connected
  - Strongly connected: Directed graph with considering direction is connected

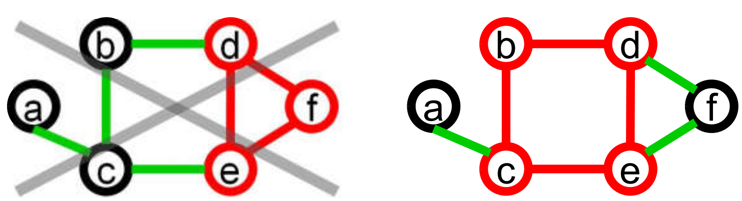


Connected | Not Connected | Strongly Connected | Weakly Connected

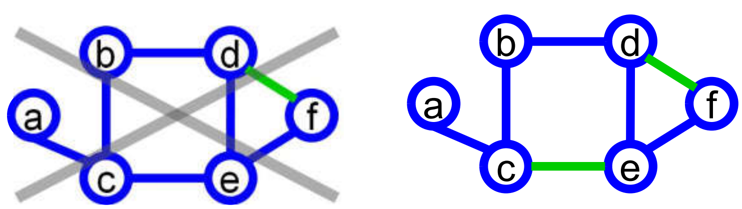
Maximal/Minimal graph

- A graph G is said to be a maximal graph (minimal graph) with respect to a property P if G has property P and no proper supergraph (subgraph) of G has the property P





maximal cycle

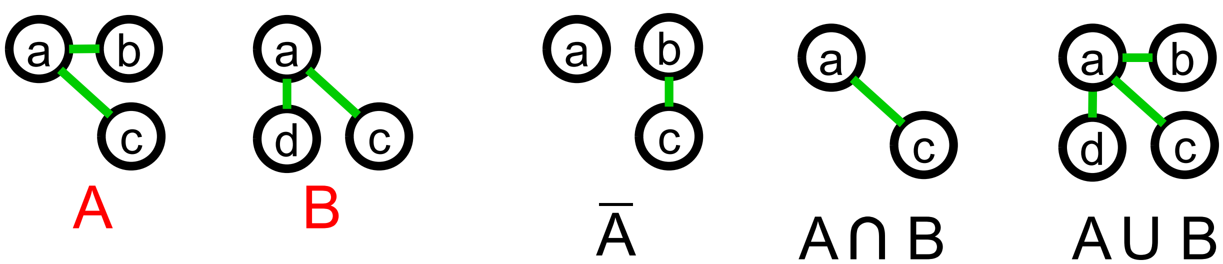


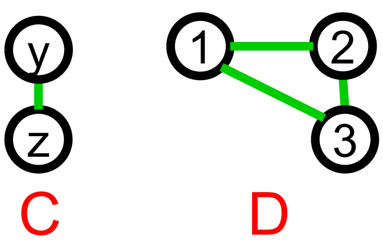
minimal connected graph

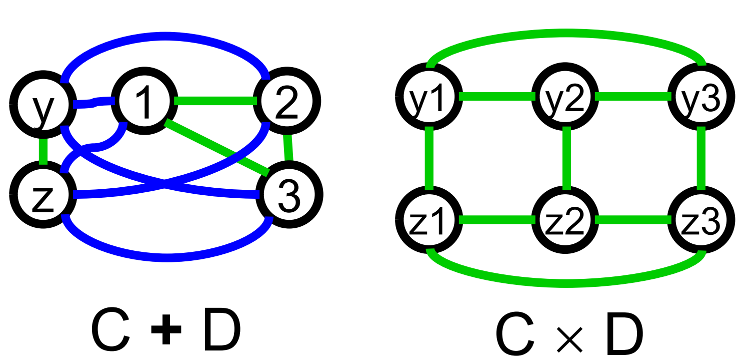
Graph Operation

- Given $G_1=(V_1,E_1)$ and $G_2=(V_2,E_2)$
  - **Complement** $\overline{G}_1$
    $(V_1, \{uv \mid u \neq v, uv \notin E_1\})$
  - **Intersection** $G_1 \cap G_2$
    $(V_1 \cap V_2, E_1 \cap E_2)$
  - **Union** $G_1 \cup G_2$
    $(V_1 \cup V_2, E_1 \cup E_2)$
  - **Join** $G_1 + G_2$ if $V_1 \cap V_2 = \emptyset$
    $(V_1 \cup V_2, E_1 \cup E_2 \cup \{uv \mid u \in V_1 \text{ and } v \in V_2\})$
  - **Cartesian Product** $G_1 \times G_2$
    $(V_1 \times V_2, \{(u_1, u_2), (v_1, v_2) \mid$
    $(u_1 = v_1 \text{ and } \{u_2, v_2\} \in E_2)$ or
    $(u_2 = v_2 \text{ and } \{u_1, v_2\} \in E_1)\})$

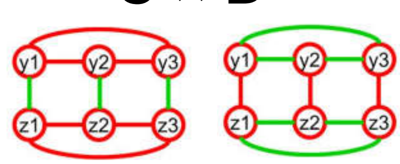
Example 1



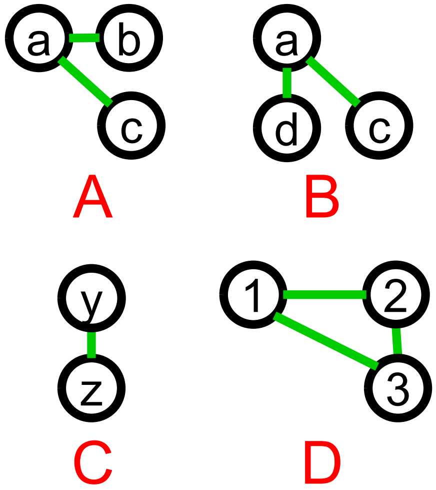




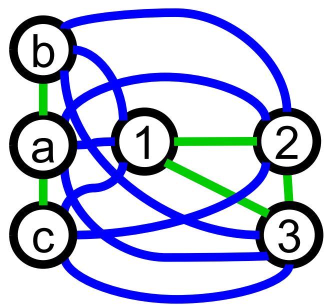
- **Complement** $\overline{G}_1$
  $(V_1, \{uv \mid u \neq v, uv \notin E_1\})$
- **Intersection** $G_1 \cap G_2$
  $(V_1 \cap V_2, E_1 \cap E_2)$
- **Union** $G_1 \cup G_2$
  $(V_1 \cup V_2, E_1 \cup E_2)$
- **Join** $G_1 + G_2$ if $V_1 \cap V_2 = \emptyset$
  $(V_1 \cup V_2, E_1 \cup E_2 \cup \{uv \mid u \in V_1 \text{ and } v \in V_2\})$
- **Cartesian Product** $G_1 \times G_2$
  $(V_1 \times V_2, \{(u_1, u_2), (v_1, v_2) \mid$
  $(u_1 = v_1 \text{ and } \{u_2, v_2\} \in E_2)$ or
  $(u_2 = v_2 \text{ and } \{u_1, v_2\} \in E_1)\})$



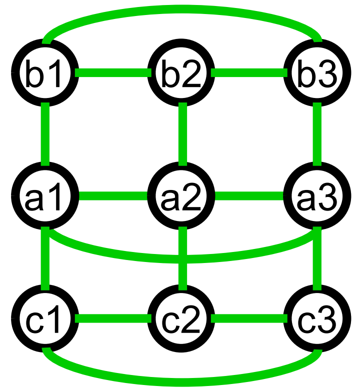
Graph Operation
Example 2



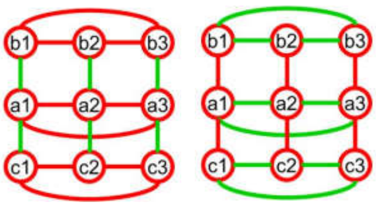
- **Complement** $\bar{G}_1$
  $(V_1, \{uv \mid u \neq v, uv \notin E_1\})$
- **Intersection** $G_1 \cap G_2$
  $(V_1 \cap V_2, E_1 \cap E_2)$
- **Union** $G_1 \cup G_2$
  $(V_1 \cup V_2, E_1 \cup E_2)$
- **Join** $G_1 + G_2$ if $V_1 \cap V_2 = \emptyset$
  $(V_1 \cup V_2, E_1 \cup E_2 \cup \{uv \mid u \in V_1 \text{ and } v \in V_2\})$
- **Cartesian Product** $G_1 \times G_2$
  $(V_1 \times V_2, \{(u_1, u_2), (v_1, v_2) \mid$
  $(u_1 = v_1 \text{ and } \{u_2, v_2\} \in E_2)$ or
  $(u_2 = v_2 \text{ and } \{u_1, v_1\} \in E_1)\})$



A + D

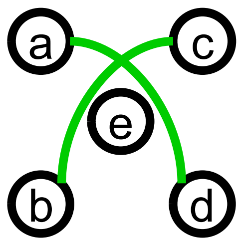


A × D

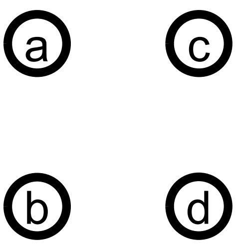


Graph Operation
Example 3

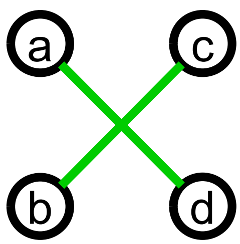
- **Complement** $\bar{G}_1$
  $(V_1, \{uv \mid u \neq v, uv \notin E_1\})$
- **Intersection** $G_1 \cap G_2$
  $(V_1 \cap V_2, E_1 \cap E_2)$
- **Union** $G_1 \cup G_2$
  $(V_1 \cup V_2, E_1 \cup E_2)$
- **Join** $G_1 + G_2$ if $V_1 \cap V_2 = \emptyset$
  $(V_1 \cup V_2, E_1 \cup E_2 \cup \{uv \mid u \in V_1 \text{ and } v \in V_2\})$
- **Cartesian Product** $G_1 \times G_2$
  $(V_1 \times V_2, \{(u_1, u_2), (v_1, v_2) \mid$
  $(u_1 = v_1 \text{ and } \{u_2, v_2\} \in E_2)$ or
  $(u_2 = v_2 \text{ and } \{u_1, v_1\} \in E_1)\})$



$\bar{W}_4$



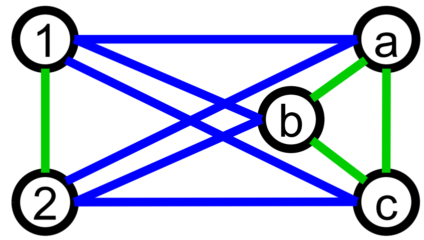
$\bar{K}_4$



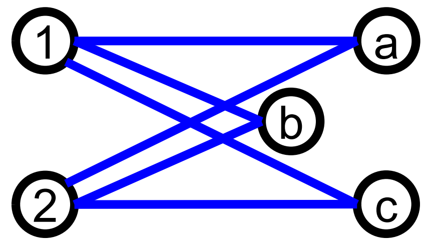
$\bar{C}_4$

Graph Operation
Example 4

- **Complement** $\overline{G}_1$
  $(V_1, \{uv \mid u \neq v, uv \notin E_1\})$
- **Intersection** $G_1 \cap G_2$
  $(V_1 \cap V_2, E_1 \cap E_2)$
- **Union** $G_1 \cup G_2$
  $(V_1 \cup V_2, E_1 \cup E_2)$
- **Join** $G_1 + G_2$ if $V_1 \cap V_2 = \emptyset$
  $(V_1 \cup V_2, E_1 \cup E_2 \cup \{uv \mid u \in V_1 \text{ and } v \in V_2\})$
- **Cartesian Product** $G_1 \times G_2$
  $(V_1 \times V_2, \{(u_1, u_2), (v_1, v_2) \mid$
  $(u_1 = v_1 \text{ and } \{u_2, v_2\} \in E_2)$ or
  $(u_2 = v_2 \text{ and } \{u_1, v_2\} \in E_1)\})$



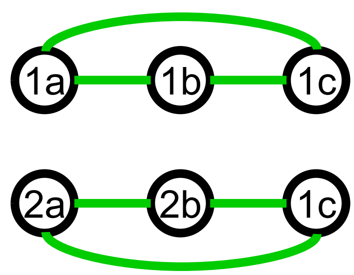
$K_2 + K_3$



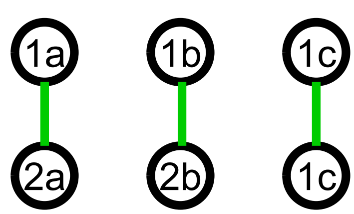
$\overline{K}_2 + \overline{K}_3$

Graph Operation
Example 5

- **Complement** $\overline{G}_1$
  $(V_1, \{uv \mid u \neq v, uv \notin E_1\})$
- **Intersection** $G_1 \cap G_2$
  $(V_1 \cap V_2, E_1 \cap E_2)$
- **Union** $G_1 \cup G_2$
  $(V_1 \cup V_2, E_1 \cup E_2)$
- **Join** $G_1 + G_2$ if $V_1 \cap V_2 = \emptyset$
  $(V_1 \cup V_2, E_1 \cup E_2 \cup \{uv \mid u \in V_1 \text{ and } v \in V_2\})$
- **Cartesian Product** $G_1 \times G_2$
  $(V_1 \times V_2, \{(u_1, u_2), (v_1, v_2) \mid$
  $(u_1 = v_1 \text{ and } \{u_2, v_2\} \in E_2)$ or
  $(u_2 = v_2 \text{ and } \{u_1, v_2\} \in E_1)\})$



$\overline{K}_2 \times K_3$

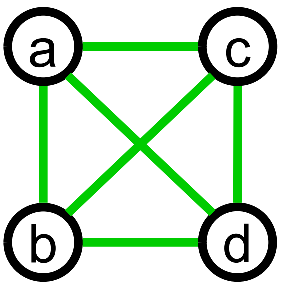


$K_2 \times \overline{K}_3$

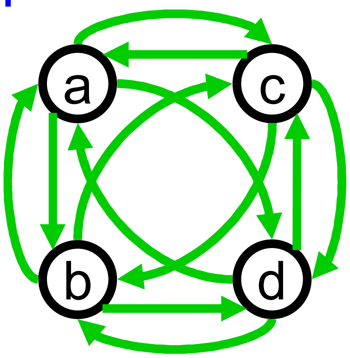
Complete Graph

- **Complete graph K_n** if there is an edge between every pair of vertices, where n is the number of vertices

- Complete Undirected Graph
- Complete Directed Graph

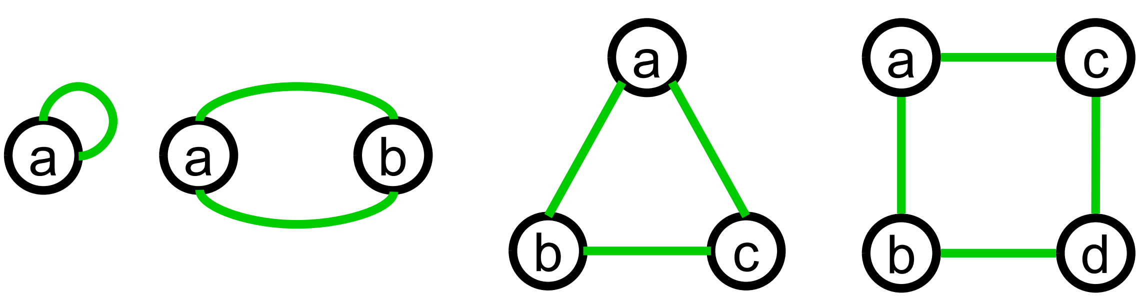


K_4



Cycle Graph

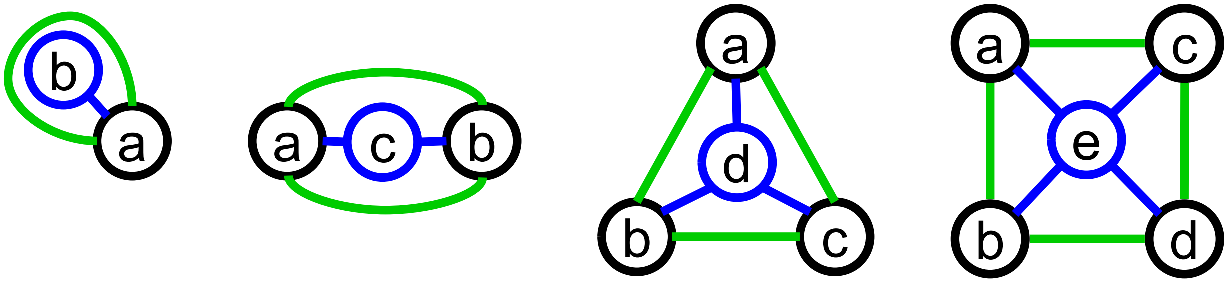
- **Cycle graph C_n** is a circular graph with V = {0,1,2,…,n-1} where vertex i is connected to (i+1) mod n and to (i-1) mod n
  - like a polygon



C_1 C_2 C_3 C_4

Type of Graph
Wheel Graph

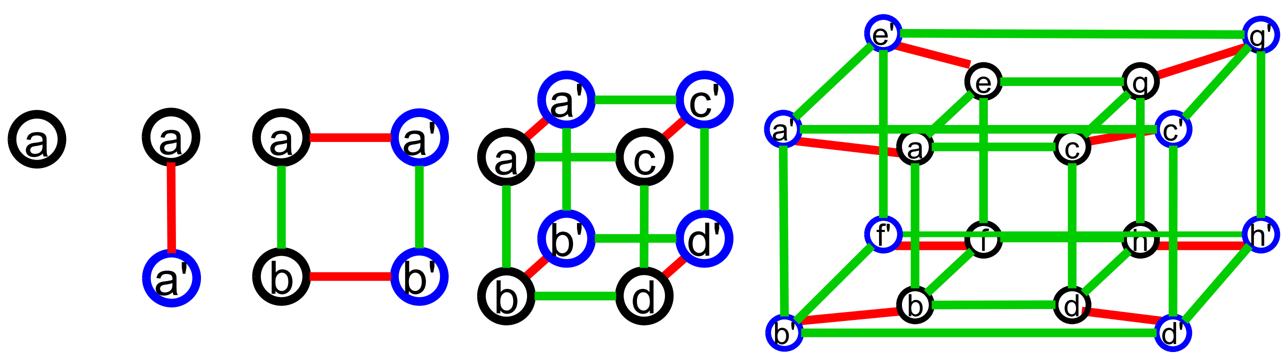
- **Wheel graph W_n** is a cycle graph with an extra vertex in the middle which contact to each of other vertices



W_1 W_2 W_3 W_4

Type of Graph
Cube

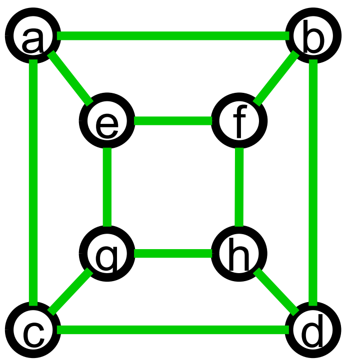
- **n-cube Q_n** is defined recursively.
  - Q_0 is just a vertex
  - Q_{n+1} is gotten by taking 2 copies of Q_n and joining each vertex v of Q_n with its copy v'

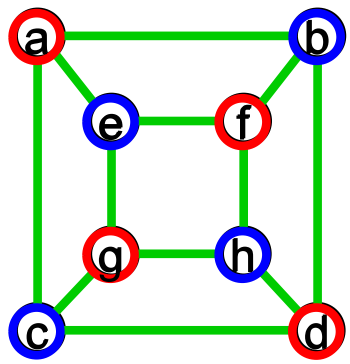


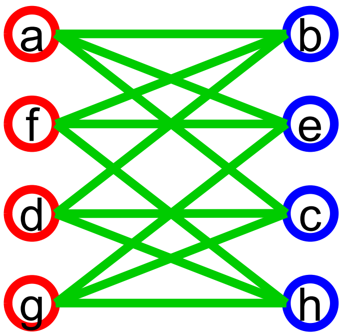
Q_0 Q_1 Q_2 Q_3 Q_4

Bipartite Graph

- A graph is bipartite if all vertices can be separated into two partitions, (i.e. $V = V_1 \cup V_2$ and $\varnothing = V_1 \cap V_2$) so that any two adjacent vertices are in different partitions
  - (V_1, V_2) is called a bipartition of V of G

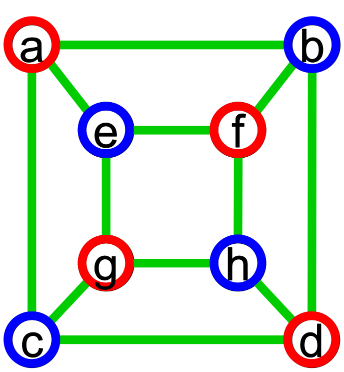


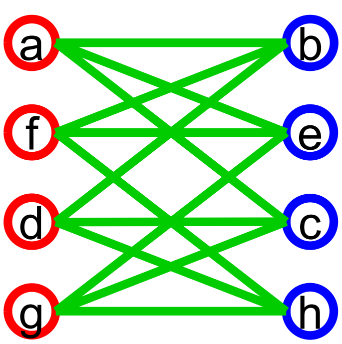




Bipartite Graph: Theorem

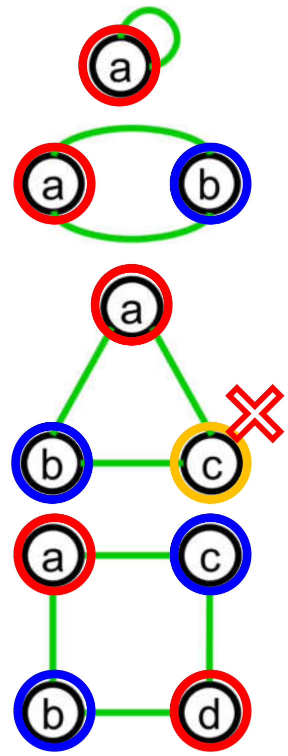
- A simple graph is bipartite if and only if it is possible to assign one of two different colors to each vertex of the graph so that no two adjacent vertices are assigned the same color





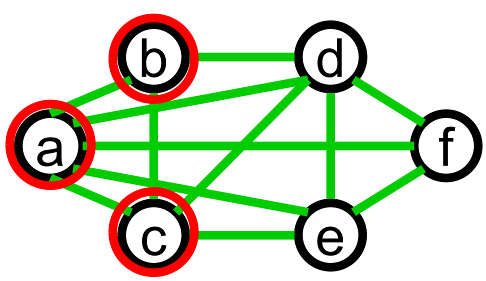
Bipartite Graph: Example 1

- Is C_n (**Cycle graph**) bipartite?
  - When n is even, Yes
    - All odd vertices are in a color and all vertices numbers are in another color
    - All vertices are only adjacent to opposite color
  - When n is odd, No (except n = 1)
    - Both n and 1 are odd, but n[th] vertex is next to the 1[st] vertex



Bipartite Graph: Example 2

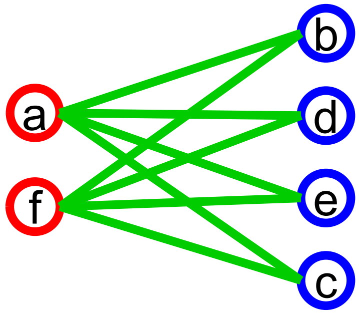
- Is the given graph bipartite?



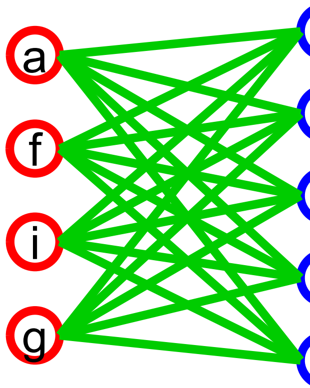
- NO
- For example, consider a, b, and c. There is two adjacent vertices are assigned the same color if only two colors are allowed

Complete Bipartite Graph

- When all possible edges exist in a simple bipartite graph with m and n vertices in two partitions, the graph is called complete bipartite $K_{m,n}$



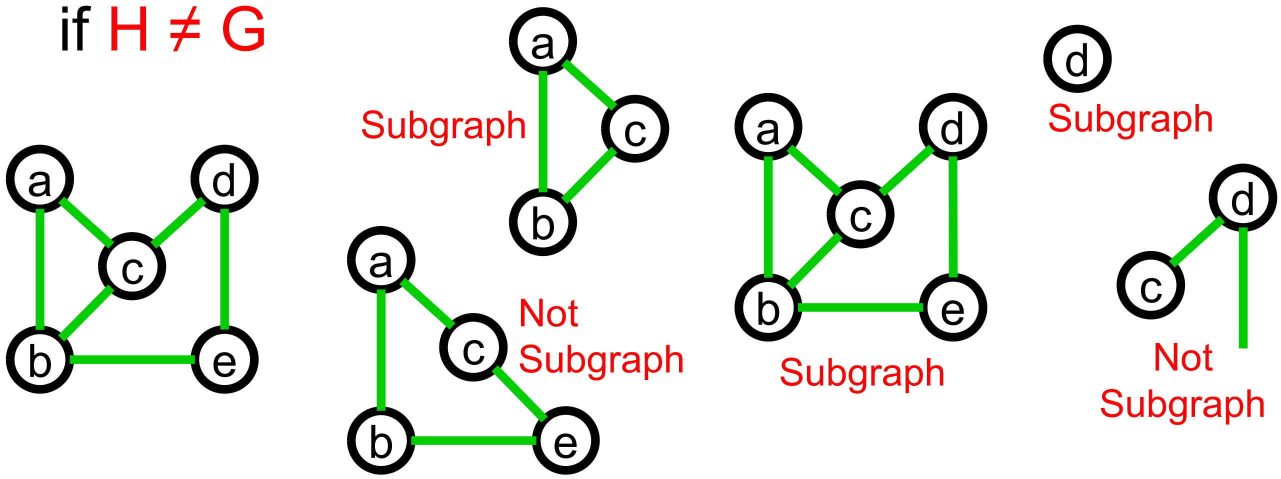
$K_{2,4}$



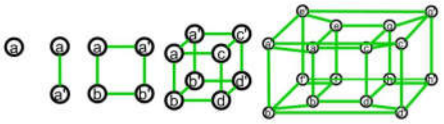
$K_{4,5}$

Subgraph

- Let G = (V,E) and H = (W,F) be graphs. H is a subgraph of G, if $W \subseteq V$ and $F \subseteq E$
  - Subgraph is a graph inside another group
- A subgraph H of G is a proper subgraph of G if $H \neq G$

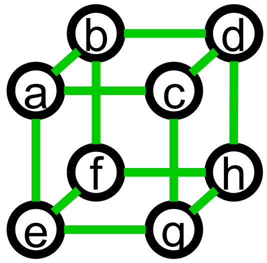


Subgraph: Example

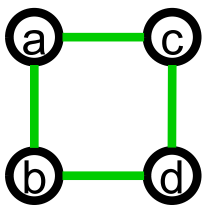


- How many different Q_2 subgraphs does Q_3 have?

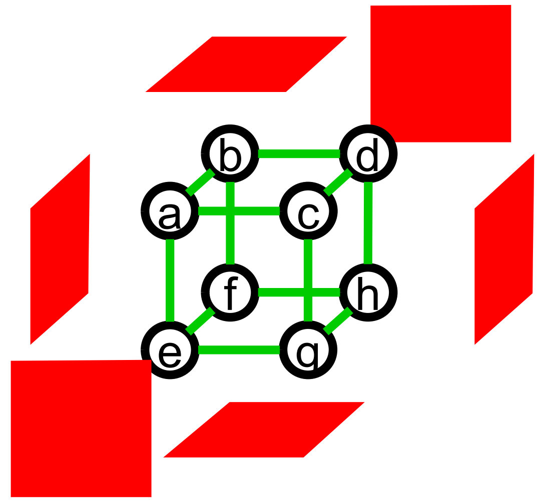
6



Q_3

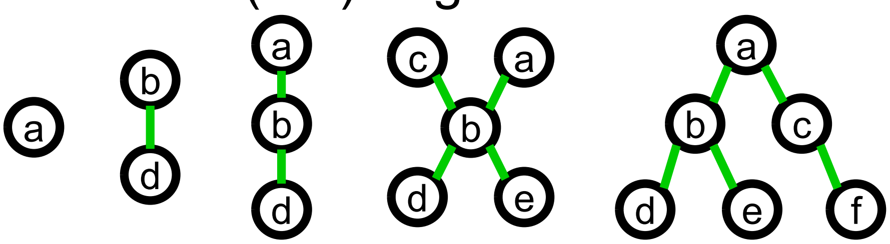


Q_2

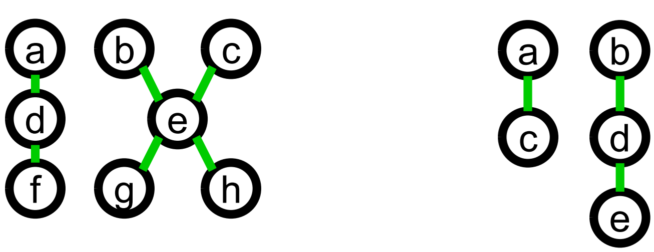


Tree

- Tree is an undirected, connected and acyclic graph
  - n vertices has (n-1) edges

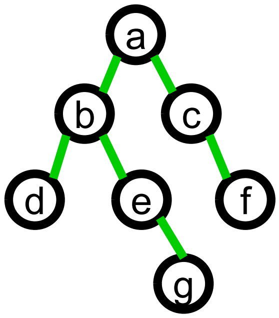


- Forest is an undirected, disconnected, acyclic graph
  - Disjoint collection of trees



Theorem 1

- A tree with at least two vertices has at least two leaves



- Assume P is a longest path in a tree T
- Prove its endpoints are leaves
- Suppose v is not a leaf,
  then v has at least two neighbors, x and y
- One of them (say x) must not in P, otherwise a cycle
- Let P' be the path that begins at x followed by P
- This is a longer path than P which is contradict to the assumption

Theorem 2

- A tree on n vertices has n − 1 edges

- For N(1)
  - If n = 1, then T has no edges
- Assume N(k) is true
  - T with n vertices has exactly n − 1
- Show N(k+1)

Theorem 2

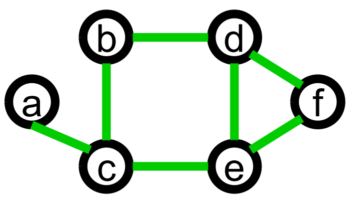
- Show T with n+1 vertices has exactly n − 1
- Since T is a tree, T has at least two leaves (Theorem 1)
- Let T' be the graph created by deleting a leaf in T
- Note that T' is a tree with n vertices, since:
  - T' is connected and acyclic
  - T' has one less vertex than T
- According to N(k), T' has n − 1 edges
- Since T' has one less edge than T, T has K edges

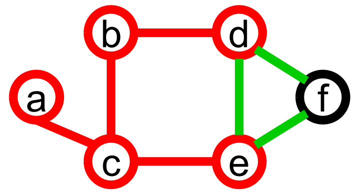
Theorem 3

- Let G be a graph with n vertices. Then the following are equivalent:
  1. G is a tree
  2. G is a maximal acyclic graph
  3. G is a minimal connected graph
  4. G is acyclic and it has n − 1 edges
  5. G is connected and it has n − 1 edges
  6. Between any two distinct vertices of G there exists a unique path

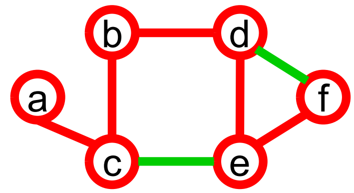
Spanning Tree

- Spanning Tree in a connected graph G is a sub-graph H of G that includes all the vertices of G and is also a tree

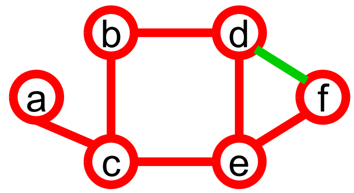




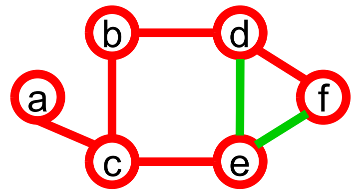
Not Spanning Tree
(not all vertices)



Spanning Tree



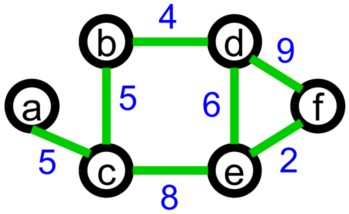
Not Spanning Tree
(not a tree)

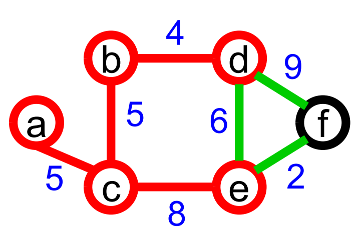


Spanning Tree

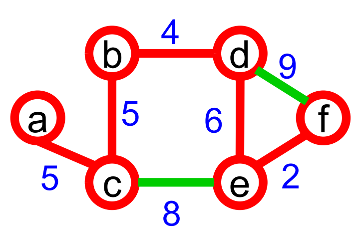
Minimum Spanning Tree

- Minimum Spanning Trees (MST) is a spanning tree with the minimal cost to call all the vertices

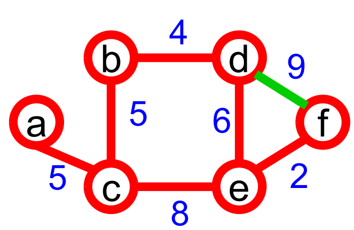




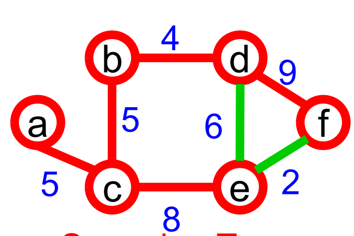
Not Spanning Tree
(not all vertices)



Spanning Tree
MST (22)



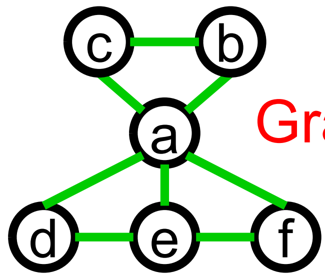
Not Spanning Tree
(not a tree)



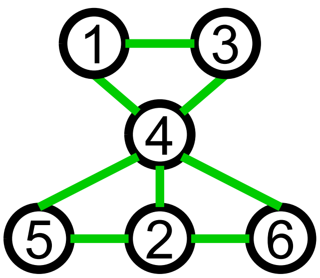
Spanning Tree
Not MST (31)

Graph Isomorphism

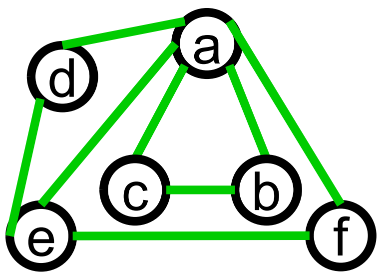
- Is the following graphs the same as Graph A?



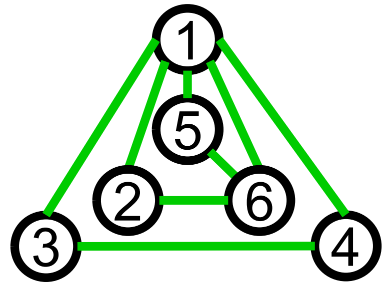
Graph A



Yes
Different Labels



Yes
Different Positions



Yes
Different Label
and Positions

Graph Isomorphism

- Two graphs, A and B, which contain the same number of graph vertices connected in the same way are said to be **isomorphic, A ≅ B**
- Applications
  - Checking fingerprint
  - Testing molecules

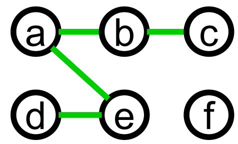
Graph Isomorphism

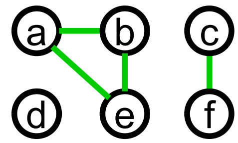
- If $G_1 \cong G_2$, do they have
  - the same number of vertices? Yes
  - the same number of edges? Yes
  - the same degree sequence? Yes
- Are $G_1 \cong G_2$, if they have
  - the same number of vertices? No
  - the same number of edges? No
  - the same degree sequence? No

$|V| = 6$

$|E| = 4$

2, 2, 2, 1, 1, 0





$|V| = 6$

$|E| = 4$

2, 2, 2, 1, 1, 0

Graph Isomorphism

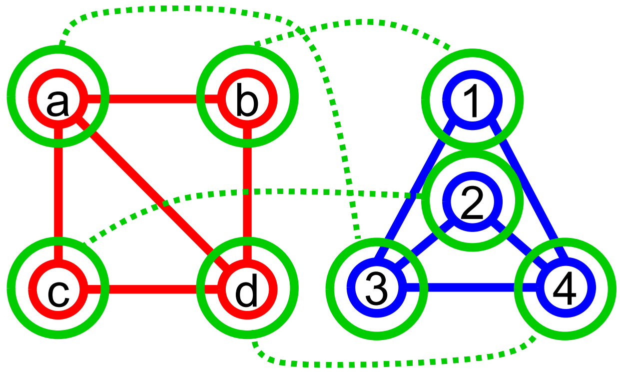
- $G_1 = (V_1,E_1) \cong G_2 = (V_2,E_2)$ if there is a bijective function $f: V_1 \rightarrow V_2$ such that for all $(u, v) \in E_1$:

$$(u, v) \in E_1 \text{ iff } (f(u), f(v)) \in E_2$$

- It is edge-preserving vertex matching
  - If there is an edge in the original graph, there is an edge after the mapping; vice versa.

Graph Isomorphism
Example

$(u, v) \in E_1$ iff $(f(u), f(v)) \in E_2$



$G_1 = (V_1, E_1)$

$V_1 = \{a, b, c, d\}$

$E_1 = \{(a,b), (a,c), (a,d), (b,d), (c,d)\}$

$G_2 = (V_2, E_2)$

$V_2 = \{1, 2, 3, 4\}$

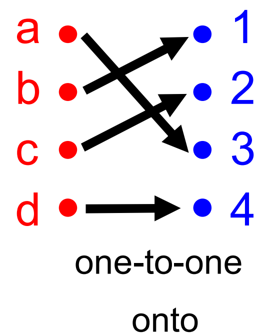
$E_2 = \{(1,3), (1,4), (2,3), (2,4), (3,4)\}$

$f(a) = 3$

$f(b) = 1$

$f(c) = 2$

$f(d) = 4$



$(a,b) \Leftrightarrow (f(a), f(b)) = (3,1)$

$(a,c) \Leftrightarrow (f(a), f(c)) = (3,2)$

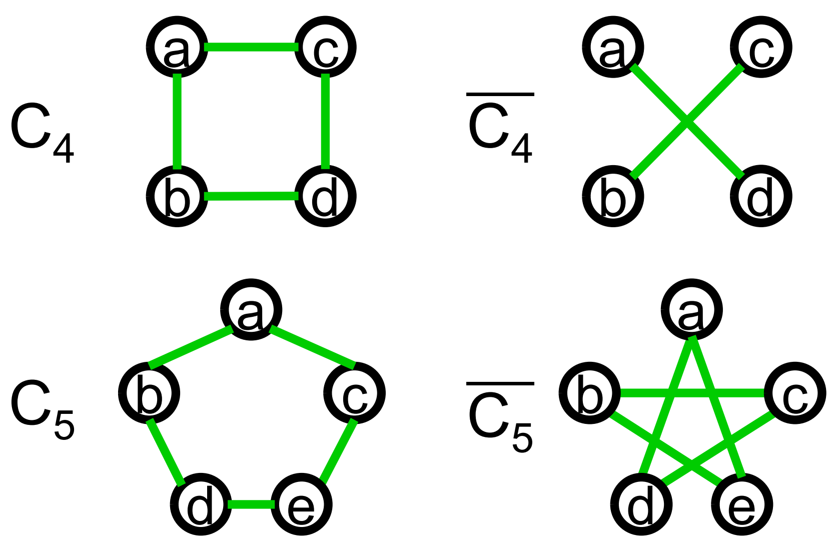
$(a,d) \Leftrightarrow (f(a), f(d)) = (3,4)$

$(b,d) \Leftrightarrow (f(b), f(d)) = (1,4)$

$(c,d) \Leftrightarrow (f(c), f(d)) = (2,4)$

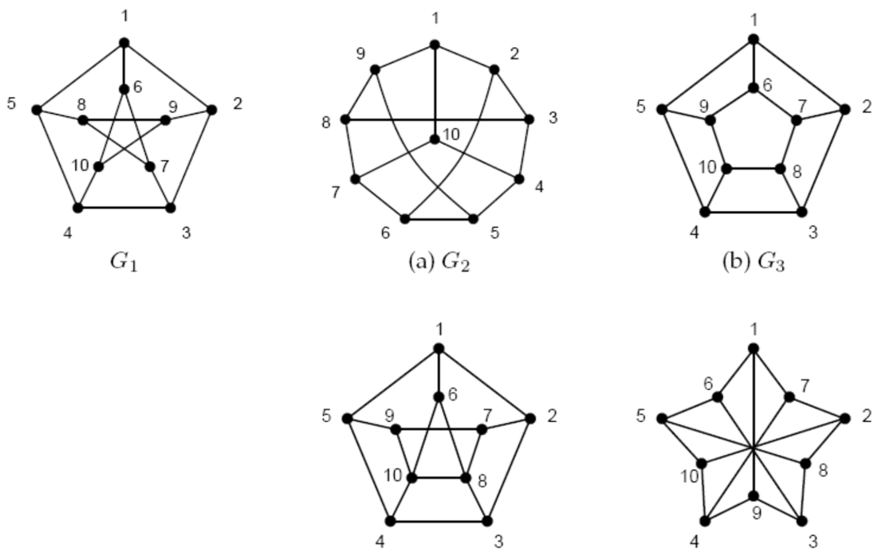
Graph Isomorphism
Self-complementary

- A graph G is called **self-complementary** if $G \cong \overline{G}$
  - C_5 and $\overline{C_5}$ are self-complementary ($C_5 \cong \overline{C_5}$)



Graph Isomorphism

- Showing Isomorphism is not easy
  - No general method which is more efficient than trying all possibilities



G_1 (a) G_2 (b) G_3 (c) G_4 (d) G_5

Graph Isomorphism

- Showing non-isomorphic is simpler
  - Violate any isomorphic-preserving property
  - Example: Are they isomorphism? NO