Discrete Structures
CSC160

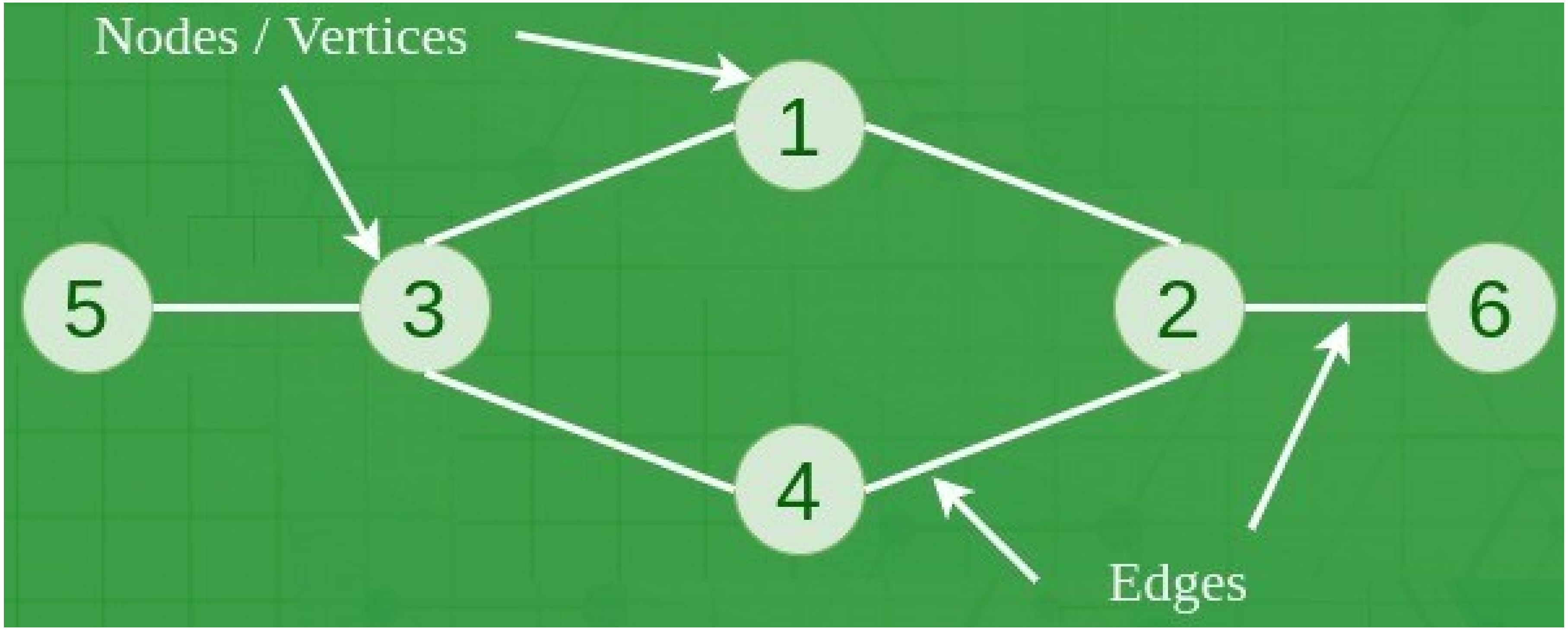
Instructor: Prakash Neupane
Godawari College
Itahari

Relations and Grapahs

- Graphs
  - A Graph is a non-linear data structure consisting of vertices and edges.
  - The vertices are sometimes also referred to as nodes and the edges are lines or arcs that connect any two nodes in the graph.
  - More formally a Graph is composed of a set of vertices(V) and a set of edges(E).
  - The graph is denoted by G(V, E).

Relations and Grapahs

- Graphs



Relations and Grapahs

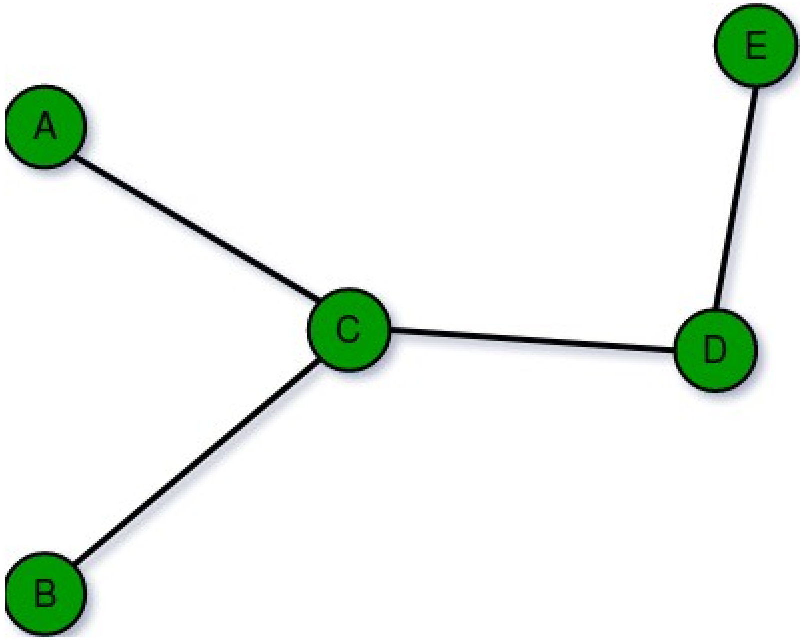
- Graphs: Components
  - **Vertices**:
  - Vertices are the fundamental units of the graph.
  - Sometimes, vertices are also known as vertex or nodes.
  - Every node/vertex can be labeled or unlabelled.
  - **Edges**:
  - Edges are drawn or used to connect two nodes of the graph.
  - It can be ordered pair of nodes in a directed graph.
  - Edges can connect any two nodes in any possible way.
  - There are no rules. Sometimes, edges are also known as arcs.
  - Every edge can be labelled/unlabelled.

Relations and Grapahs

- Graphs: Types (but not limited too)
  - Finite graphs
  - Infinite graphs
  - Trivial graphs
  - Simple graphs
  - Multi graphs
  - Null graphs
  - Complete graphs

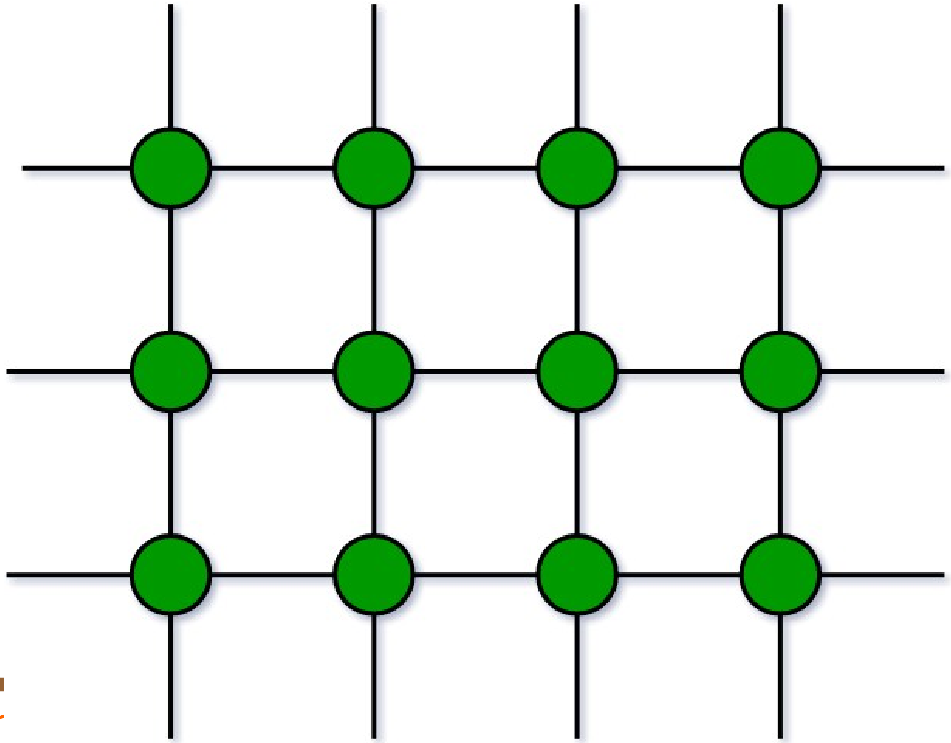
Relations and Grapahs

- Graphs: Types
  - Finite graphs
  - A graph is said to be finite if it has a finite number of vertices and a finite number of edges.



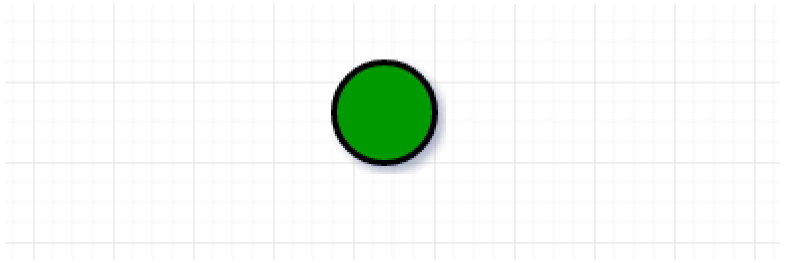
Relations and Grapahs

- Graphs: Types
  - Infinite graphs
  - A graph is said to be infinite if it has an infinite number of vertices as well as an infinite number of edges.



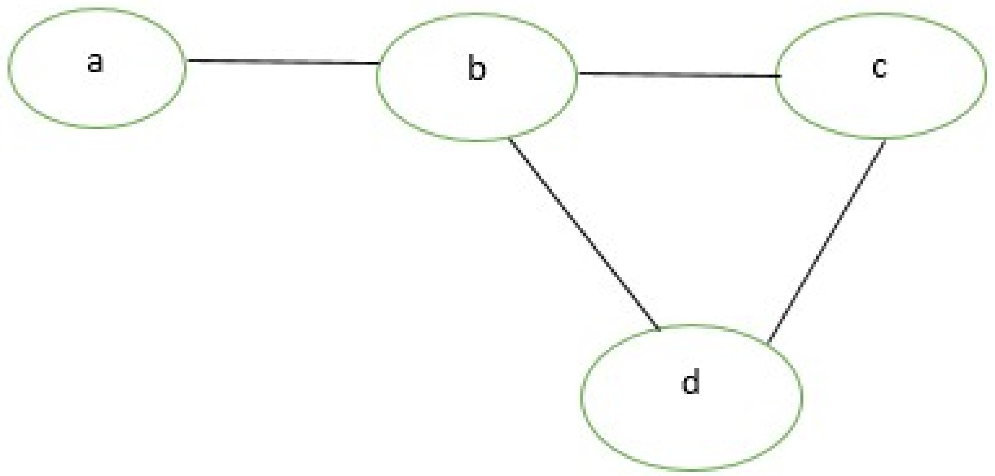
Relations and Grapahs

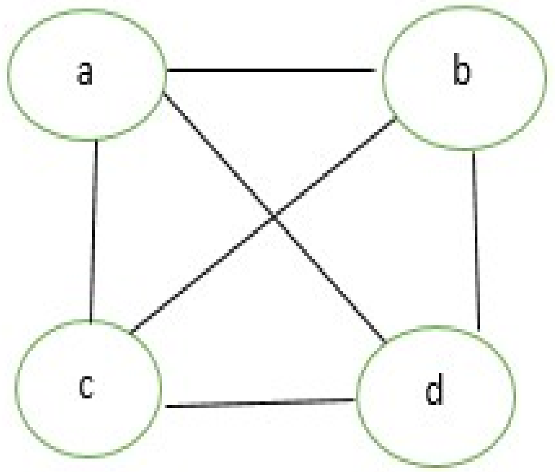
- Graphs: Types
  - Trivial graphs
  - A graph is said to be trivial if a finite graph contains only one vertex and no edge.
  - A trivial graph is a graph with only one vertex and no edges.
  - It is also known as a singleton graph or a single vertex graph.



Relations and Grapahs

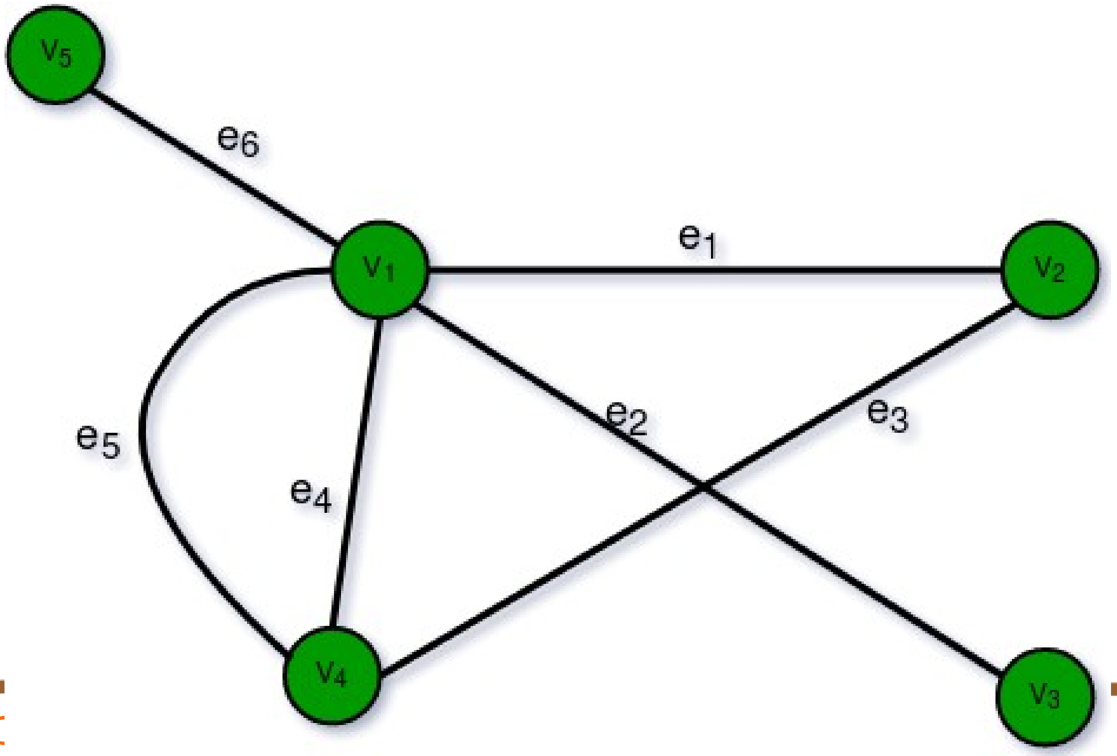
- Graphs: Types
  - Simple graphs
  - A simple graph is a graph that does not contain more than one edge between the pair of vertices.





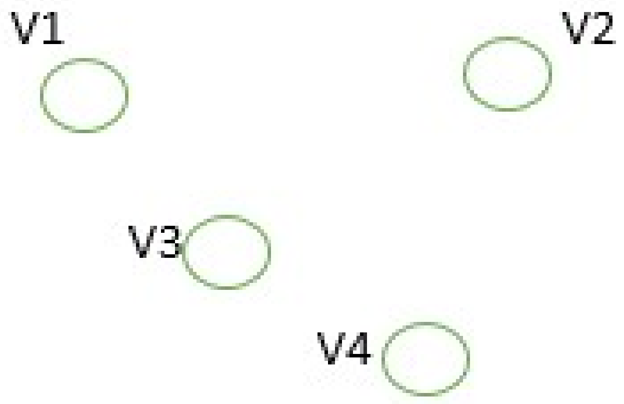
Relations and Grapahs

- Graphs: Types
  - Multi graphs
  - Any graph which contains some parallel edges but doesn't contain any self-loop is called a multigraph.



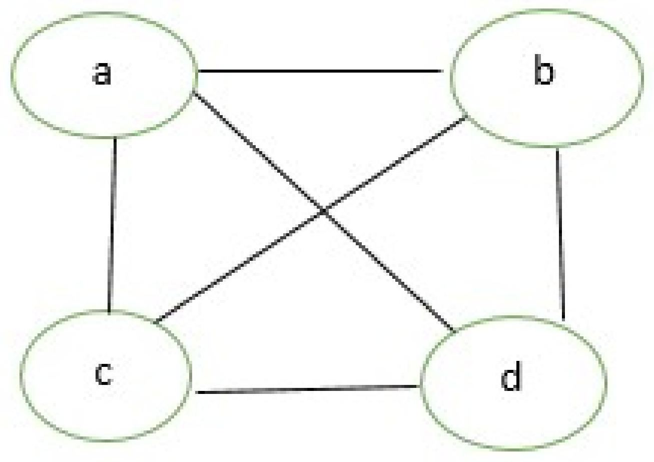
Relations and Grapahs

- Graphs: Types
  - Null graphs
  - A graph of order n and size zero is a graph where there are only isolated vertices with no edges connecting any pair of vertices.
  - A null graph is a graph with no edges. In other words, it is a graph with only vertices and no connections between them.



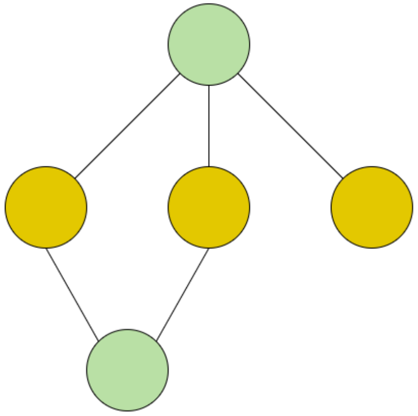
Relations and Grapahs

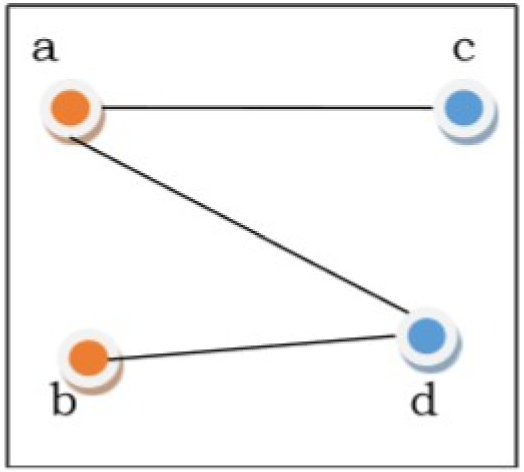
- Graphs: Types
  - Complete graphs
  - A simple graph with n vertices is called a complete graph if the degree of each vertex is n-1, that is, one vertex is attached with n-1 edges or the rest of the vertices in the graph.
  - A complete graph is also called Full Graph.

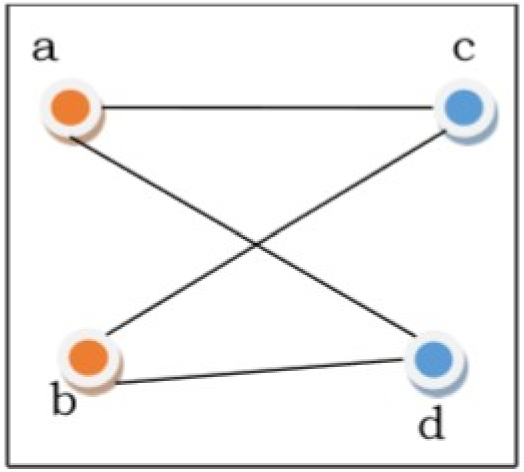


Relations and Grapahs

- Graphs: Types
  - Bipartite Graph
  - Formally, a graph G = (V, E) is bipartite if and only if its vertex set V can be partitioned into two non-empty subsets X and Y, such that every edge in E has one endpoint in X and the other endpoint in Y.
  - This partition of vertices is also known as bi-partition.


Example of Bipartite Graph





Instructor: Prakash Neupane

Graph Representation

- Graphs can be represented in various ways depending on the context and the specific requirements of the problem you are trying to solve.
- There are mainly two ways to represent a graph −
  - Adjacency Matrix
  - Adjacency List

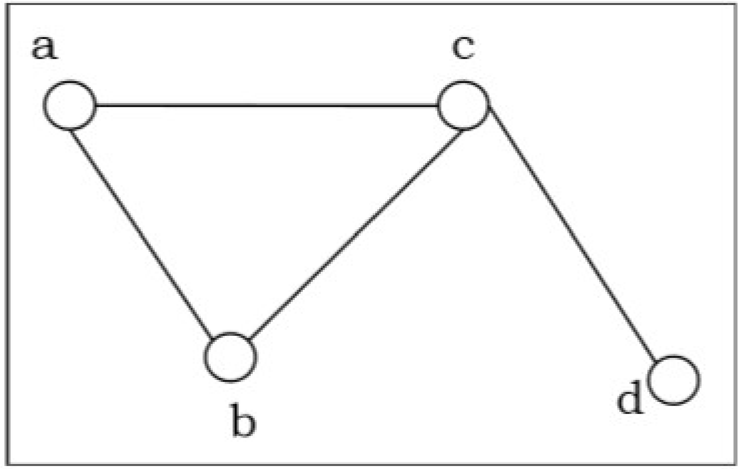
Relations and Grapahs

- **Graph Representation: Adjacency Matrix**
- An Adjacency Matrix A[V][V] is a 2D array of size V×V where V is the number of vertices in a undirected graph.
- If there is an edge between Vx to Vy then
  - the value of A[Vx][Vy]=1 and A[Vy][Vx]=1,
- otherwise
  - the value will be zero.
- And for a directed graph, if there is an edge between Vx to Vy, then
  - the value of A[Vx][Vy]=1,
- otherwise
  - the value will be zero.

Relations and Grapahs

- **Graph Representation: Adjacency Matrix**
  - Adjacency matrix of the above undirected graph will be

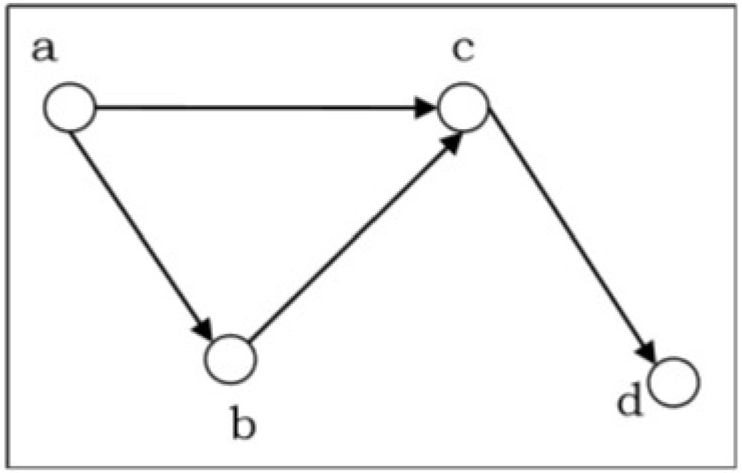
	a	b	c	d
a	0	1	1	0
b	1	0	1	0
c	1	1	0	1
d	0	0	1	0



Relations and Grapahs

- **Graph Representation: Adjacency Matrix**
  - Adjacency matrix of the above undirected graph will be

	a	**b**	**c**	**d**
a	0	1	1	0
b	0	0	1	0
c	0	0	0	1
d	0	0	0	0



Relations and Grapahs

- **Graph Representation: Adjacency Matrix**
  - Another way to represent a graph with no multiple edges is to use adjacency lists, which specify the vertices that are adjacent to each vertex of the graph.

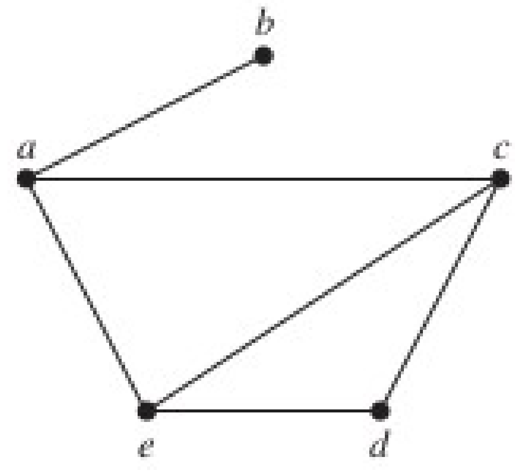


FIGURE 1 A Simple Graph.

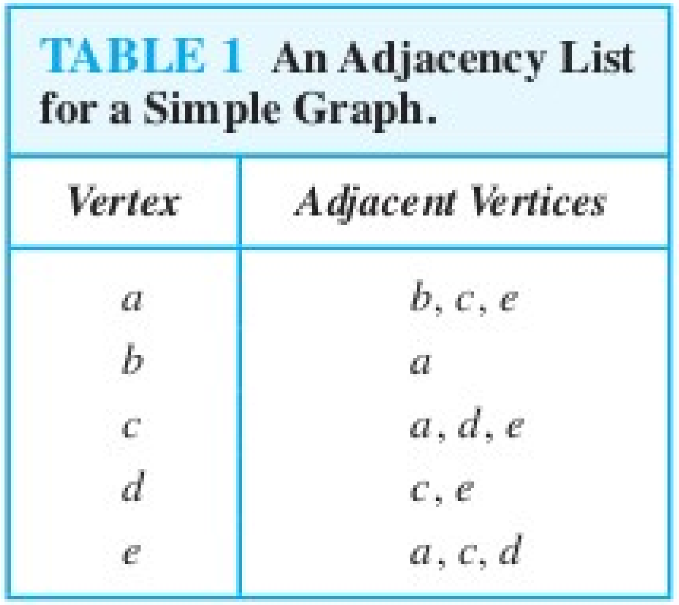


TABLE 1 An Adjacency List for a Simple Graph.

Vertex	*Adjacent Vertices*
a	*b, c, e*
b	*a*
c	*a, d, e*
d	*c, e*
e	*a, c, d*

Relations and Grapahs

- **Graph Representation: Adjacency Matrix**
  - Another way to represent a graph with no multiple edges is to use adjacency lists, which specify the vertices that are adjacent to each vertex of the graph.

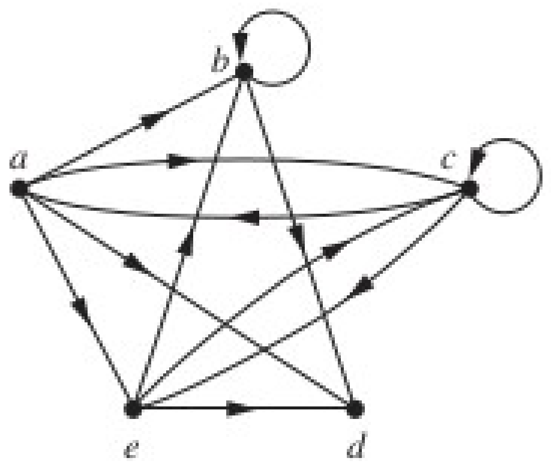


FIGURE 2 A Directed Graph.

TABLE 2 An Adjacency List for a Directed Graph.

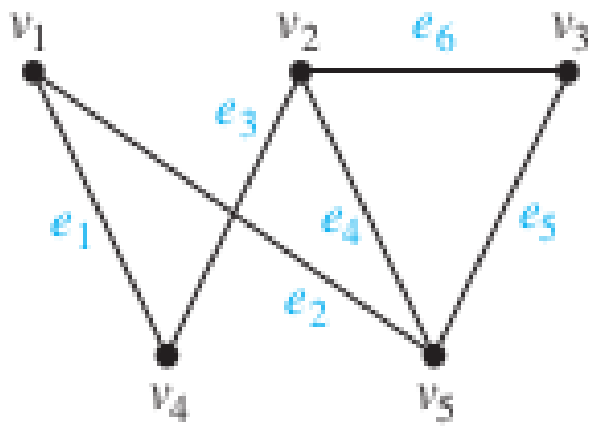
Initial Vertex	*Terminal Vertices*
a	*b*, *c*, *d*, *e*
b	*b*, *d*
c	*a*, *c*, *e*
d	
e	*b*, *c*, *d*

Relations and Grapahs

- **Graph Representation: Incidence Matrices**

Another common way to represent graphs is to use **incidence matrices**. Let $G = (V, E)$ be an undirected graph. Suppose that $v_1, v_2, \ldots, v_n$ are the vertices and $e_1, e_2, \ldots, e_m$ are the edges of G. Then the incidence matrix with respect to this ordering of V and E is the $n \times m$ matrix $\mathbf{M} = [m_{ij}]$, where

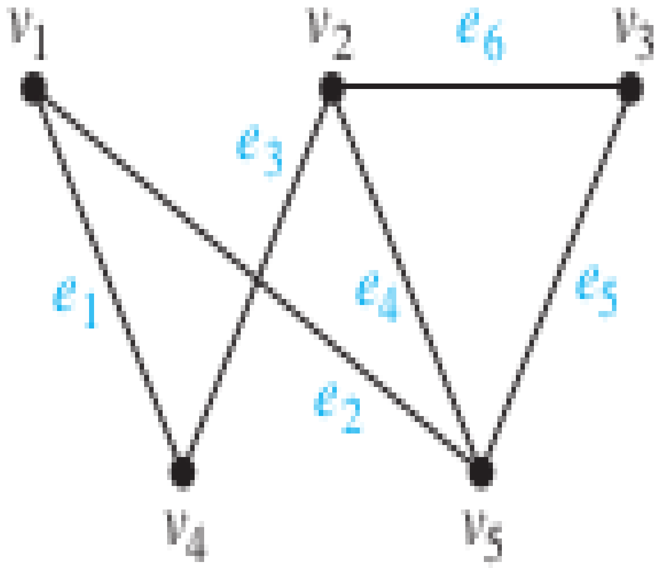
$$m_{ij} = \begin{cases} 1 & \text{when edge } e_j \text{ is incident with } v_i, \\ 0 & \text{otherwise.} \end{cases}$$



$$\begin{array}{c|cccccc} & e_1 & e_2 & e_3 & e_4 & e_5 & e_6 \\ \hline v_1 & 1 & 1 & 0 & 0 & 0 & 0 \\ v_2 & 0 & 0 & 1 & 1 & 0 & 1 \\ v_3 & 0 & 0 & 0 & 0 & 1 & 1 \\ v_4 & 1 & 0 & 1 & 0 & 0 & 0 \\ v_5 & 0 & 1 & 0 & 1 & 1 & 0 \end{array}.$$

Relations and Grapahs

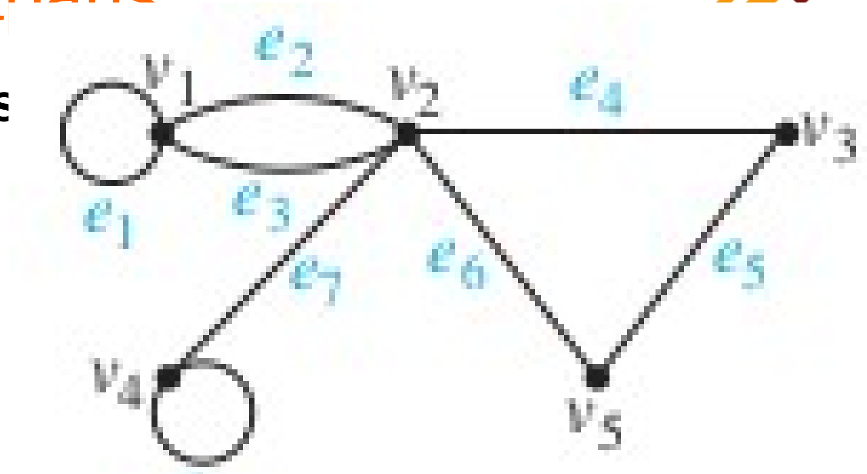
- **Graph Representation: Incidence Matrices**



$$\begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \end{array} \begin{array}{c} \begin{array}{cccccc} e_1 & e_2 & e_3 & e_4 & e_5 & e_6 \end{array} \\ \left[\begin{array}{cccccc} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 \end{array}\right] \end{array}.$$

Relations and Graphs

- **Graph Representation: Incidence Matrices**
  - .



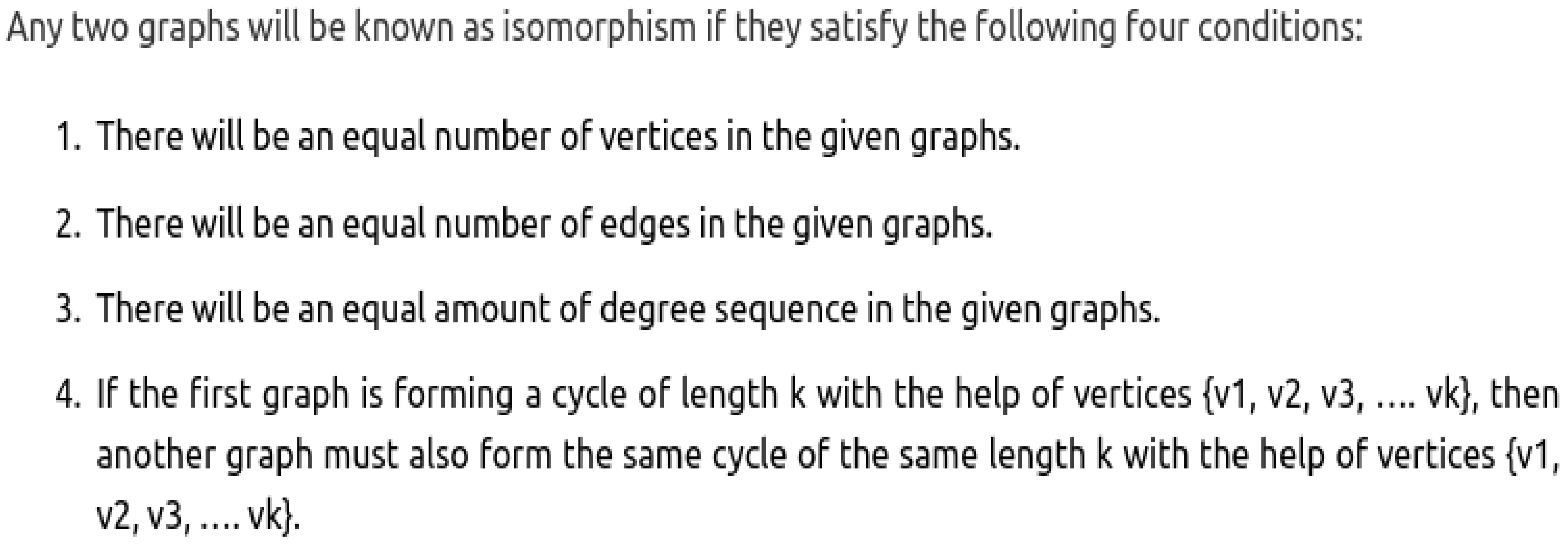
$$\begin{array}{c} & e_1 & e_2 & e_3 & e_4 & e_5 & e_6 & e_7 & e_8 \\ \begin{array}{c} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \end{array} & \left[\begin{array}{cccccccc} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \end{array}\right] \end{array}.$$

Relations and Grapahs

- **Isomorphism of Graphs**
- **Defn:**
  - The simple graphs G1 = (V1 , E1) and G2 = (V2 , E2) are isomorphic if there exists a one-to-one and onto function f from V1 to V2 with the property that a and b are adjacent in G1 if and only if f (a) and f (b) are adjacent in G2 , for all a and b in V1 . Such a function f is called an isomorphism. * Two simple graphs that are not isomorphic are called nonisomorphic.
  - **In other words, when two simple graphs are isomorphic, there is a one-to-one correspondence between vertices of the two graphs that preserves the adjacency relationship. Isomorphism of simple graphs is an equivalence relation.**

Relations and Grapahs

- **Isomorphism of Graphs**
- **Conditions for graph isomorphism**

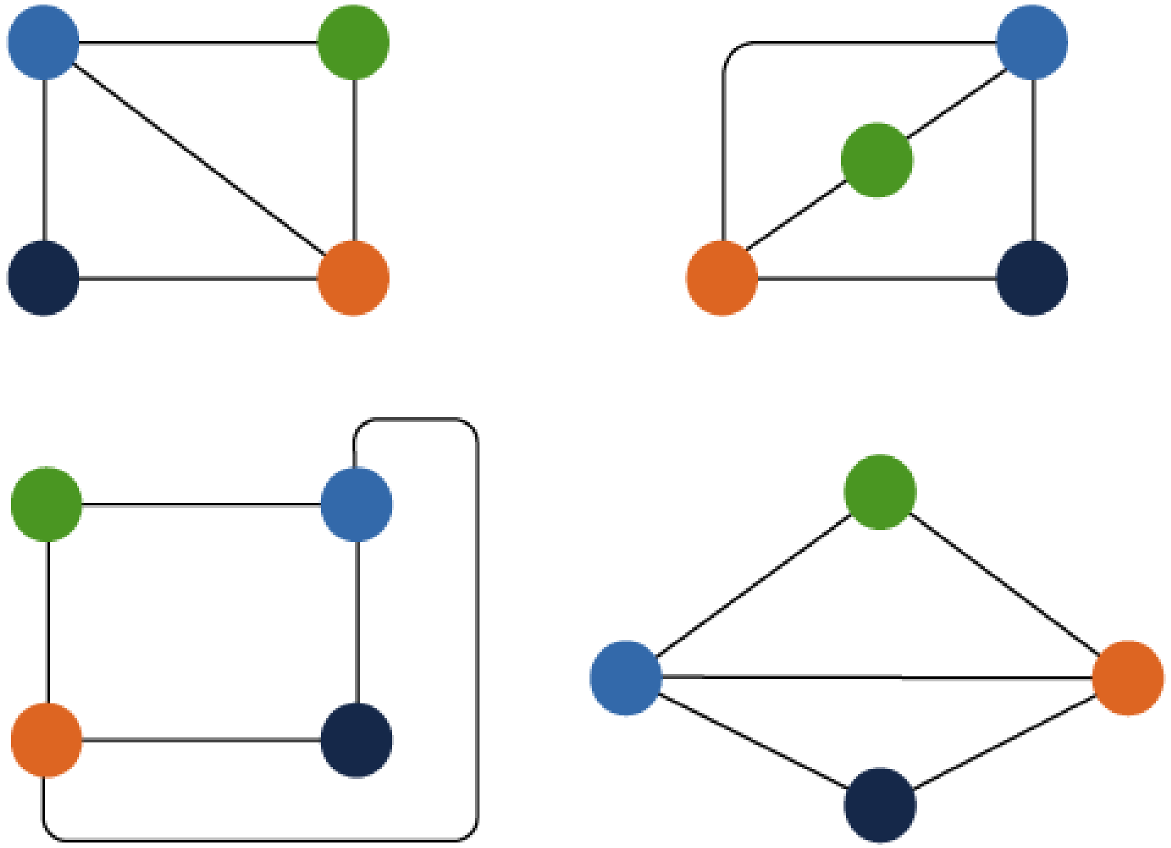


Any two graphs will be known as isomorphism if they satisfy the following four conditions:

1. There will be an equal number of vertices in the given graphs.
2. There will be an equal number of edges in the given graphs.
3. There will be an equal amount of degree sequence in the given graphs.
4. If the first graph is forming a cycle of length k with the help of vertices {v1, v2, v3, ... vk}, then another graph must also form the same cycle of the same length k with the help of vertices {v1, v2, v3, ... vk}.

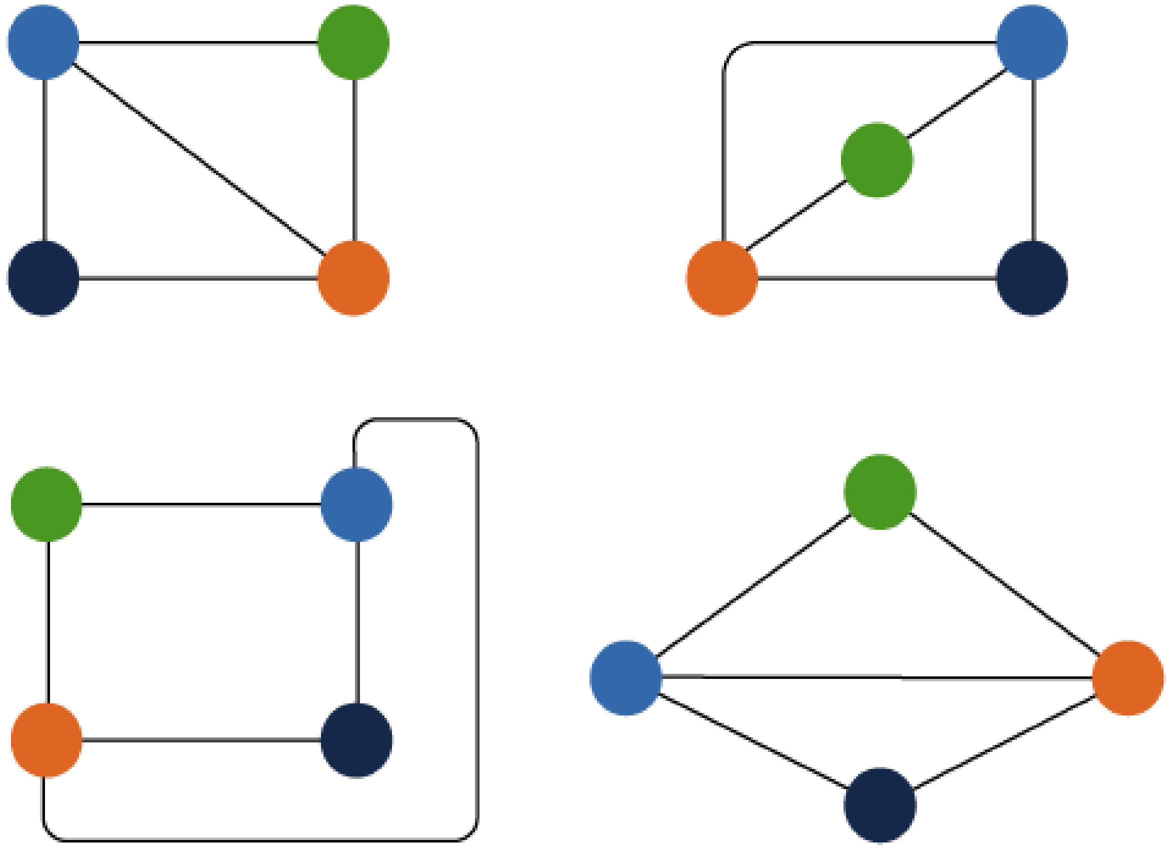
Relations and Grapahs

- **Isomorphism of Graphs**



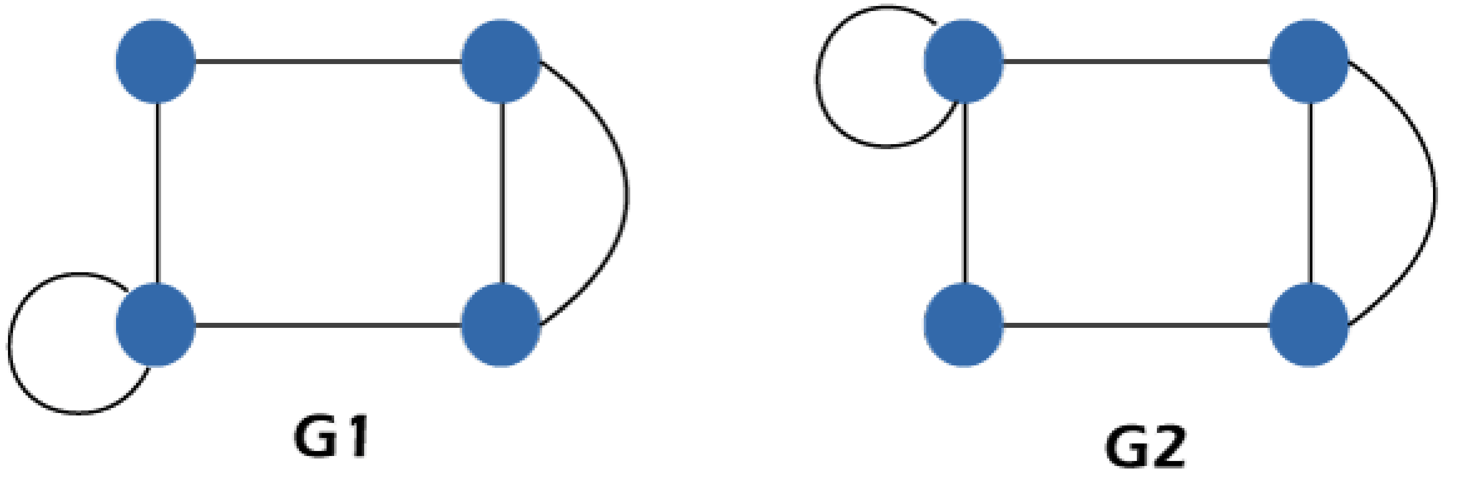
Relations and Grapahs

- **Isomorphism of Graphs**
-



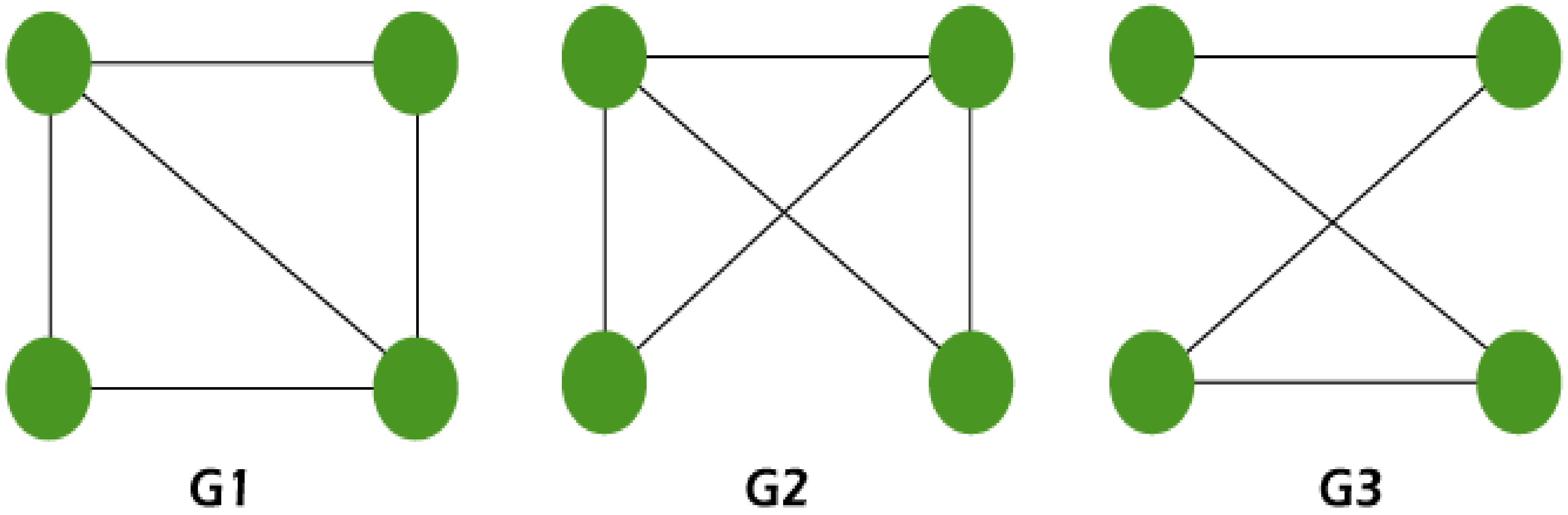
Relations and Grapahs

- **Isomorphism of Graphs**
-



Relations and Grapahs

- **Isomorphism of Graphs**
-



Relations and Grapahs

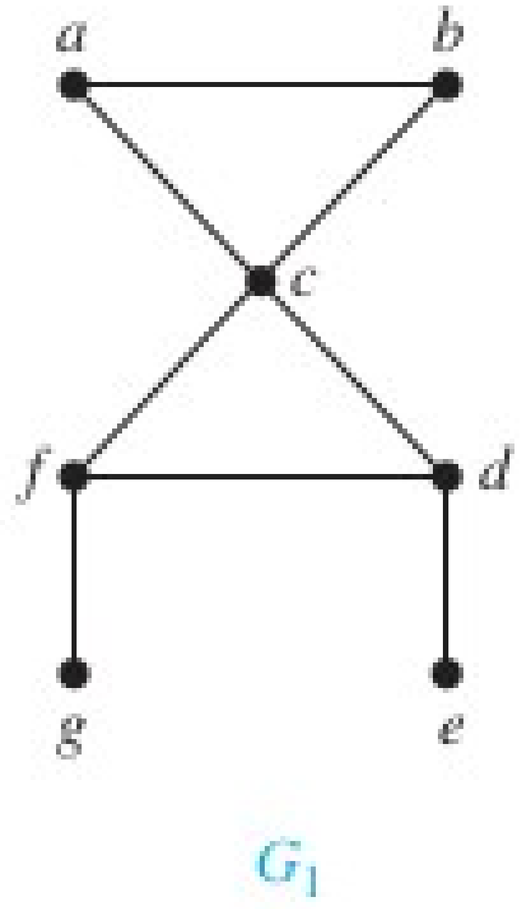
- **Connectivity**
  - A **path** is a sequence of edges that begins at a vertex of a graph and travels from vertex to vertex along edges of the graph.
  - As the path travels along its edges, it visits the vertices along this path, that is, the endpoints of these edges.
-

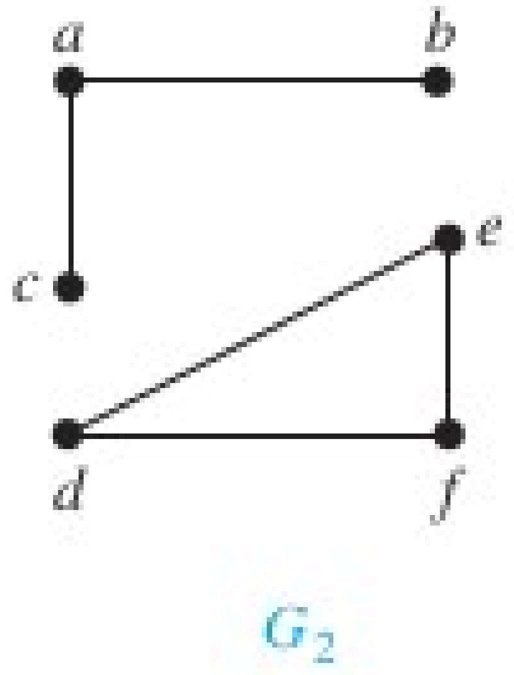
Relations and Grapahs

- **Connectivity**
  - An undirected graph is called connected if there is a path between every pair of distinct vertices of the graph.
  - An undirected graph that is not connected is called disconnected.
  - We say that we disconnect a graph when we remove vertices or edges, or both, to produce a disconnected subgraph.

Relations and Grapahs

- **Connectivity**
  –



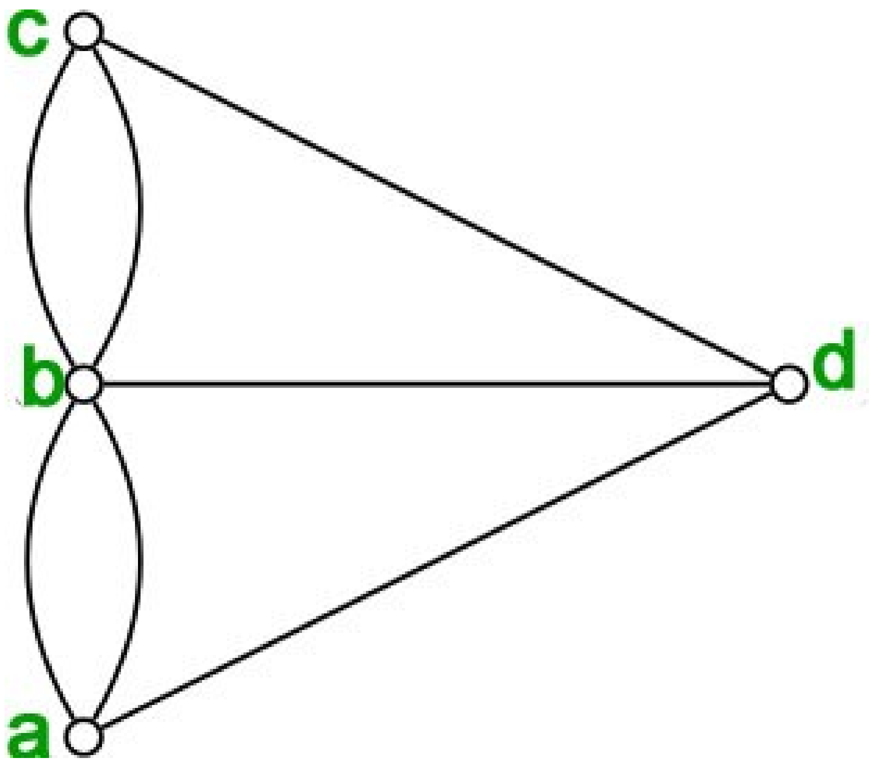


Relations and Graphs

- Euler Path and Circuits
  - An Euler path is a path that uses every edge of a graph exactly once.
  - An Euler circuit is a circuit that uses every edge of a graph exactly once.
  - An Euler path starts and ends at different vertices.
  - An Euler circuit starts and ends at the same vertex.

Relations and Graphs

- Euler Path and Circuits
  - The Konigsberg bridge problem’s graphical representation :

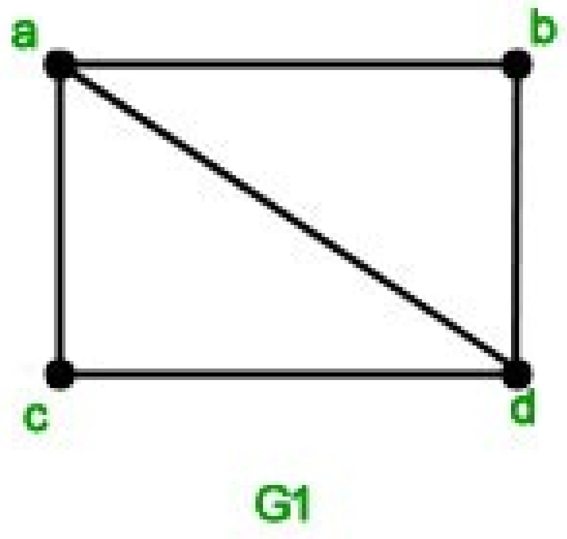


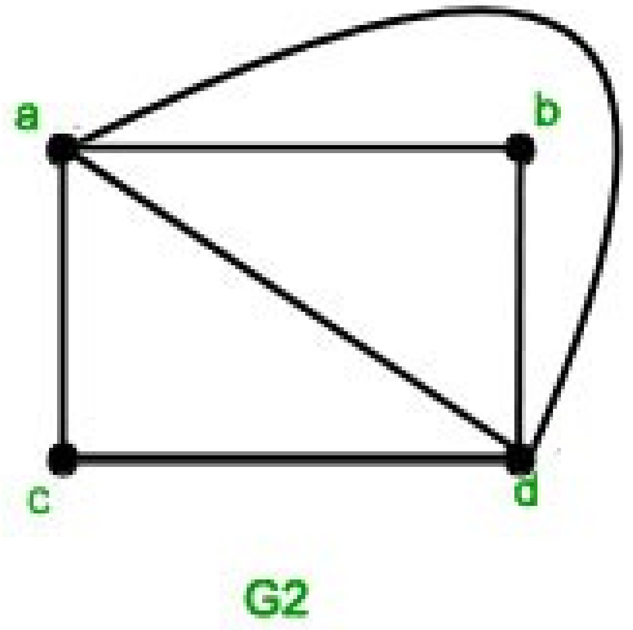
Relations and Graphs

- Euler Path and Circuits
  - There are simple criteria for determining whether a multigraph has a Euler path or a Euler circuit.
  - If the graph has exactly zero vertices with an odd degree, it has an Euler path.
  - If the graph has exactly two vertices with an odd degree (and all other vertices have even degrees), it has an Euler path that starts at one of the vertices with an odd degree and ends at the other.
  - For any multigraph to have a Euler circuit, all the degrees of the vertices must be even.

Relations and Graphs

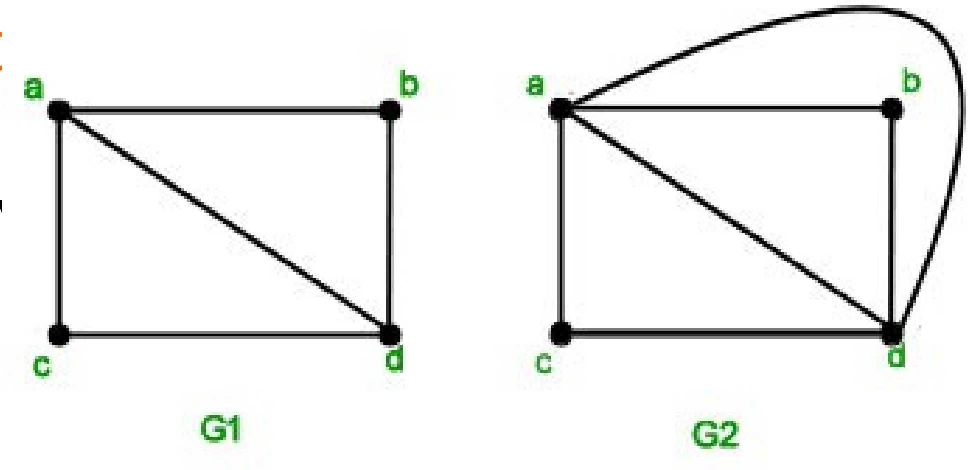
- Euler Path and Circuits
  - Which graphs shown below have an Euler path or Euler circuit?





Relations anc

- Euler Path and Circuits
  - Which graphs shown below ha



Solution – G_1 has two vertices of odd degree a and d and the rest of them have even degree. So this graph has an Euler path but not an Euler circuit. The path starts and ends at the vertices of odd degree. The path is- a, c, d, a, b, d.
G_2 has four vertices all of even degree, so it has a Euler circuit. The circuit is – a, d, b, a, c, d, a.

Relations and Graphs

- Hamiltonian Path and Circuits
- Hamiltonian Path –
  - A simple path in a graph G that passes through every vertex exactly once is called a Hamiltonian path.
- Hamiltonian Circuit –
  - A simple circuit in a graph G that passes through every vertex exactly once is called a Hamiltonian circuit.

Relations and Graphs

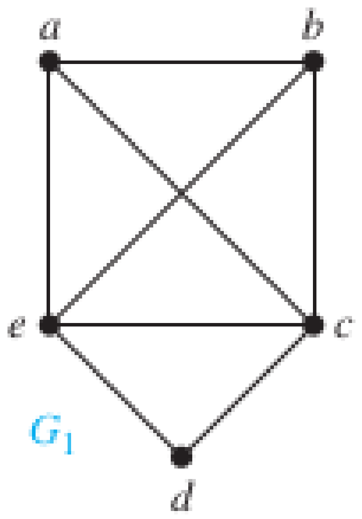
- Hamiltonian Path Criteria
- Dirac's Theorem:
  - In a simple graph with n vertices (i.e. n>=3), if each vertex has a degree of at least n/2, there is a Hamiltonian path.
- Ore's Theorem:
  - In a simple graph with n vertices, if for every non-adjacent pair of vertices u and v, deg(u) + deg(v) ≥ n, there is a Hamiltonian path.

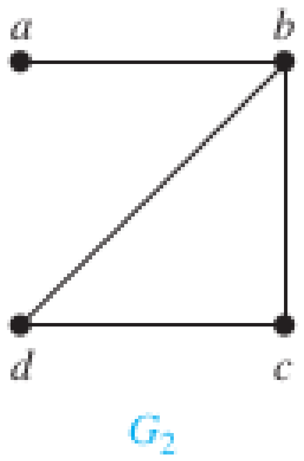
Relations and Graphs

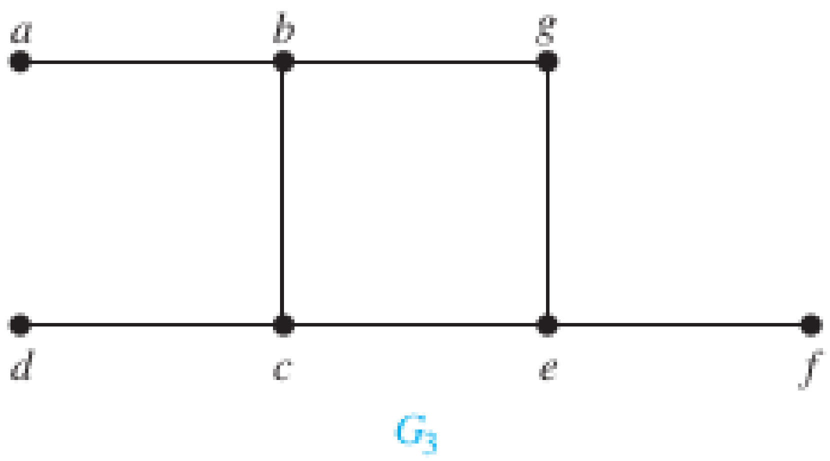
- Hamiltonian Circuit Criteria
- Dirac's Theorem:
  - In a simple graph with n vertices (i.e. n>=3), if each vertex has a degree of at least n/2, there is a Hamiltonian circuit.
- Ore's Theorem:
  - In a simple graph with n vertices, if for every non-adjacent pair of vertices u and v, deg(u) + deg(v) $\geq$ n, there is a Hamiltonian circuit.

Relations and Graphs

- Hamiltonian Path and Circuits
- Which of the simple graphs in Figure have a Hamilton circuit or, if not, a Hamilton path?





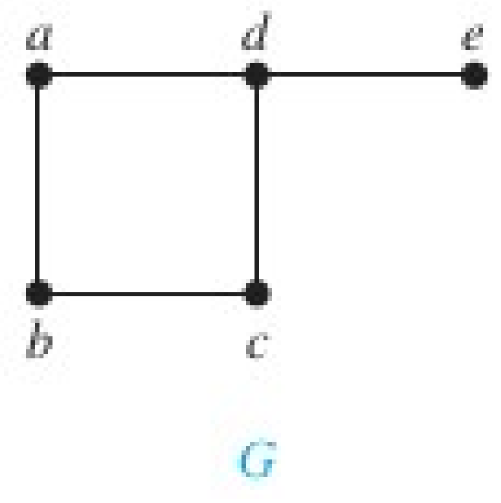


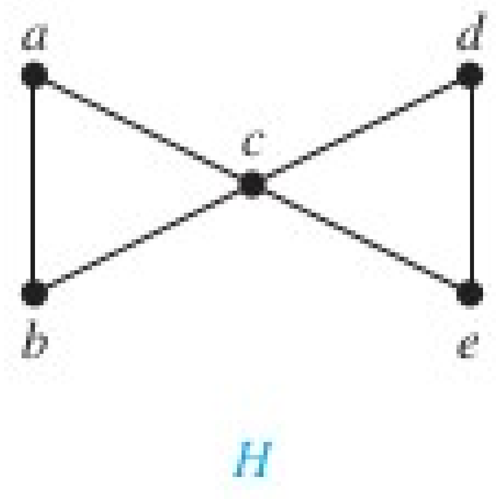
Relations and Graphs

- Hamiltonian Path and Circuits
- Which of the simple graphs in Figure have a Hamilton circuit or, if not, a Hamilton path?
  - G1 has a Hamilton circuit: a, b, c, d, e, a.
  - There is no Hamilton circuit in G2 (this can be seen by noting that any circuit containing every vertex must contain the edge {a, b} twice), but G2 does have a Hamilton path, namely, a, b, c, d.
  - G3 has neither a Hamilton circuit nor a Hamilton path, because any path containing all vertices must contain one of the edges {a, b}, {e, f }, and {c, d} more than once.

Relations and Graphs

- Hamiltonian Path and Circuits
- Which of the simple graphs in Figure have a Hamilton circuit or, if not, a Hamilton path?
  -





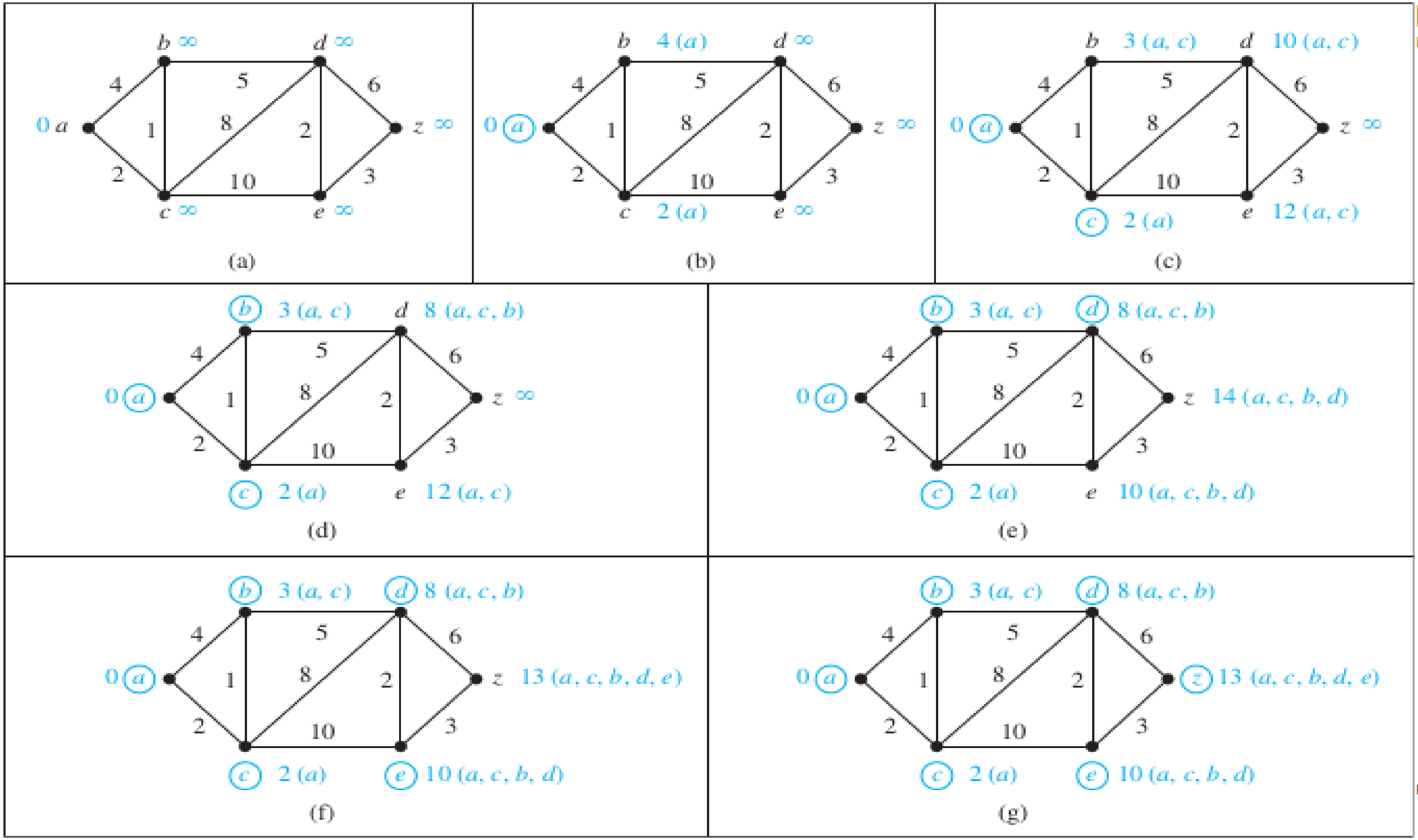
Relations and Graphs

- **A Shortest Path Algorithm**
- **It** is a computational method used to find the most efficient path or route between two points (nodes) in a graph.
- These algorithms are essential in various applications, including navigation systems, network routing, transportation planning, and optimization problems.
- some of the most common shortest path algorithms:
  - Dijkstra's Algorithm
  - Bellman-Ford Algorithm
  - Floyd-Warshall Algorithm

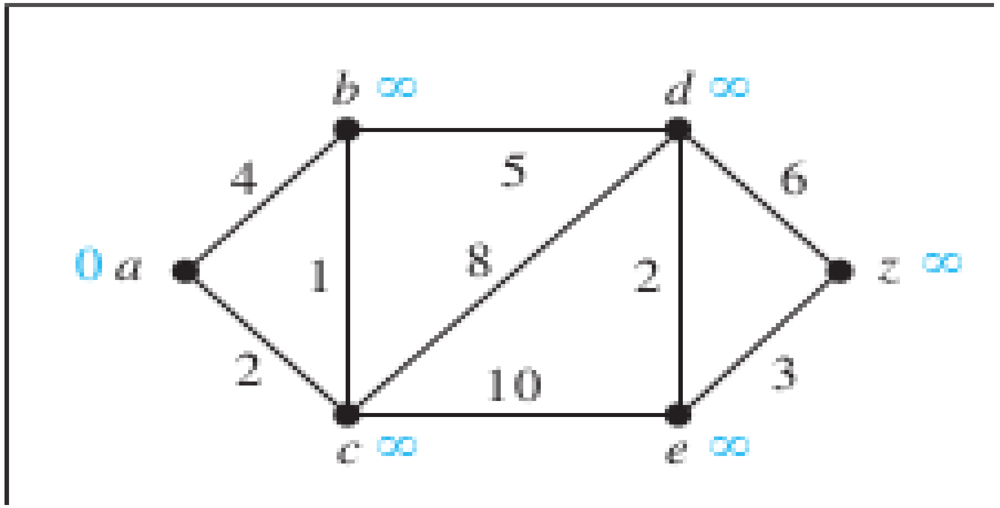
Relations and Graphs

- Goal: Find the shortest path from a source vertex to all other vertices in a weighted graph.
  - Initialization:
    - Set the source vertex's distance to 0 and all other vertices' distances to infinity.
    - Create a set of unvisited vertices.
  - Iteration:
    - Choose the unvisited vertex with the smallest distance as the current vertex.
    - For each neighbor of the current vertex, calculate tentative distances and update if shorter.
  - Repeat:
    - Continue iterating until all vertices have been visited or the smallest tentative distance is infinity.
  - Path Reconstruction:
    - After completion, you can reconstruct the shortest path from source to any vertex using recorded distances.
  - Constraints: Works for graphs with non-negative edge weights.
- Dijkstra's Algorithm efficiently finds the shortest path in various applications, such as navigation systems and network routing.

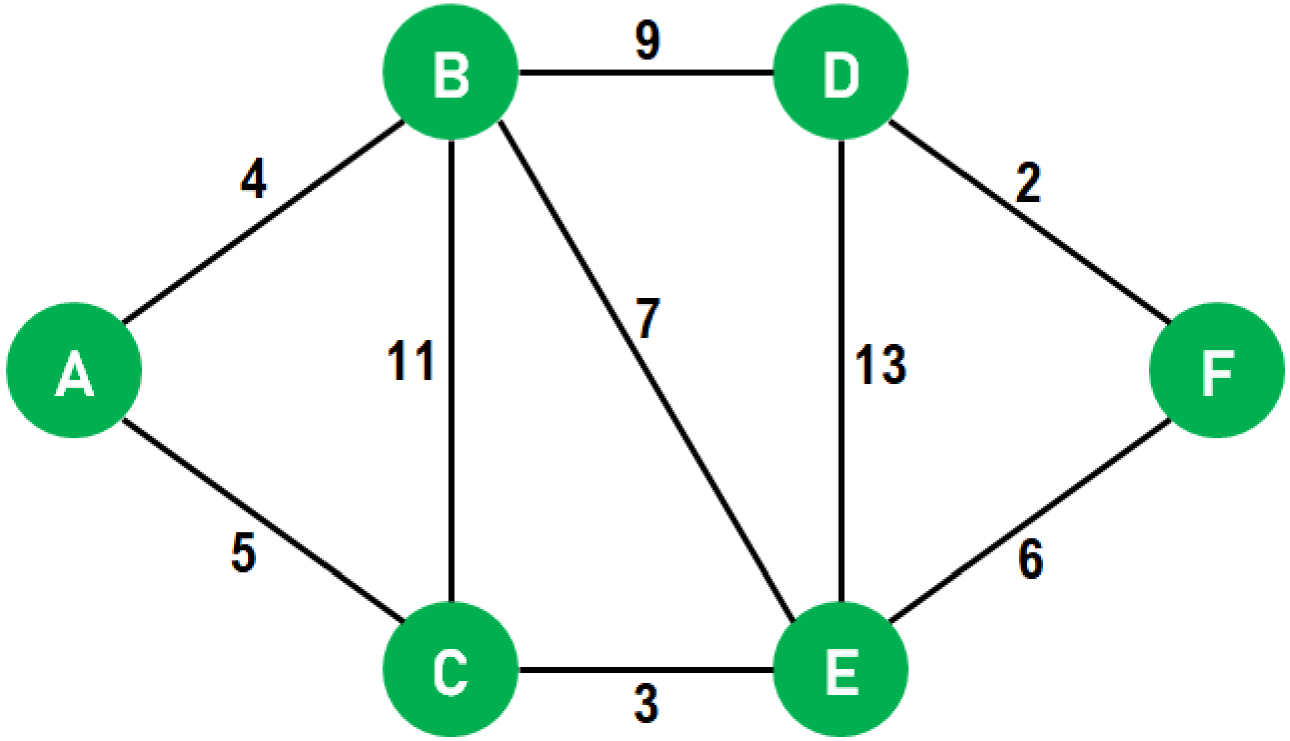
Dijkstra's Algorithm: Example



Dijkstra's Algorithm: Example

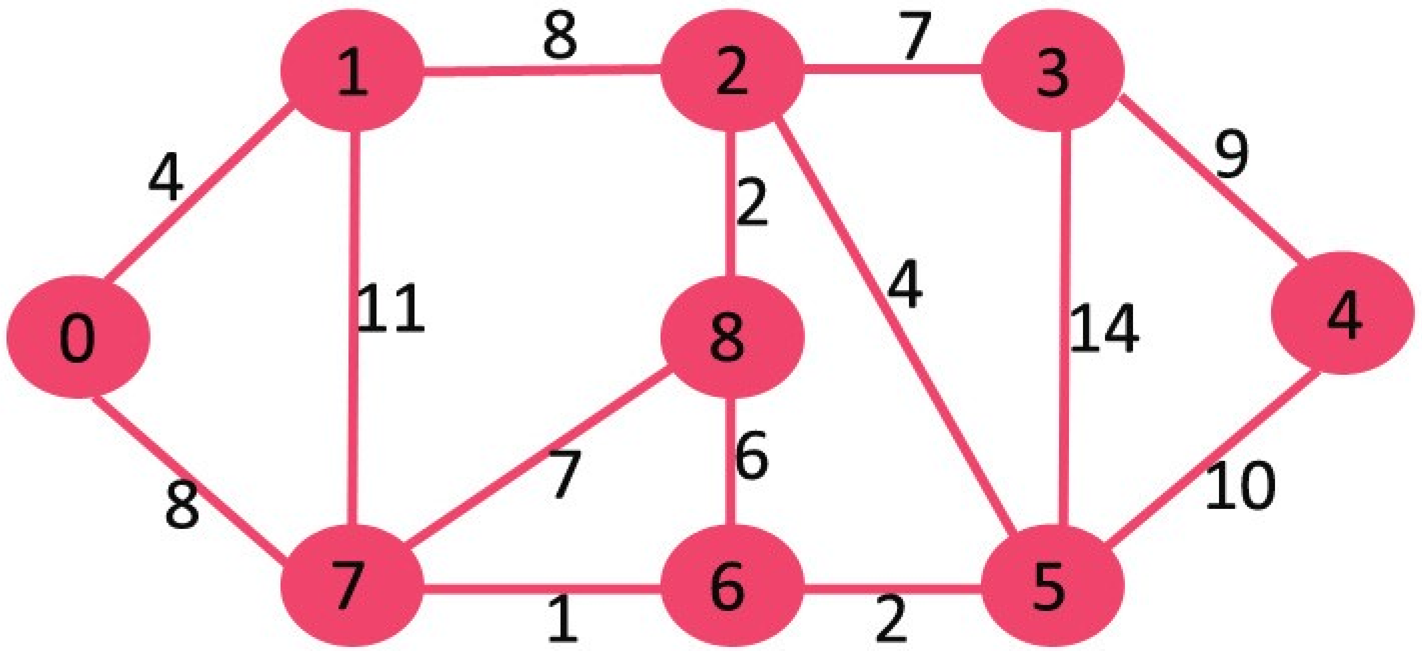


Dijkstra's Algorithm: Find the shortest path from src to goal node i.e. A to F.



Dijkstra's Algorithm: Find the shortest path from src to goal node i.e. A to F.

Dijkstra's Algorithm: Find the shortest path from src to goal node i.e. 0 to 8.



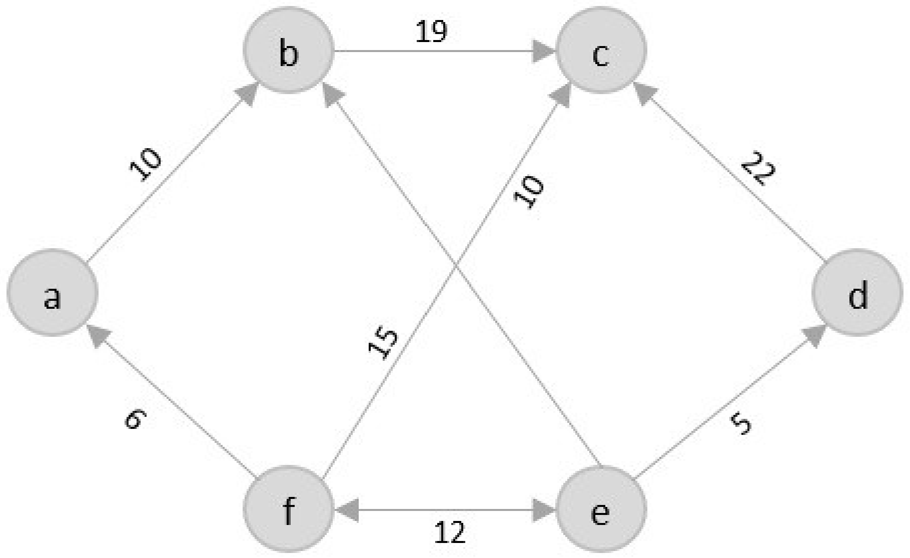
Dijkstra's Algorithm: Find the shortest path from src to goal node i.e. 0 to 8.

The travelling salesman problem

- The travelling salesman problem is a graph computational problem where the salesman needs to visit all cities (represented using nodes in a graph) in a list just once and the distances (represented using edges in the graph) between all these cities are known.
- The solution that is needed to be found for this problem is the shortest possible route in which the salesman visits all the cities and returns to the origin city.

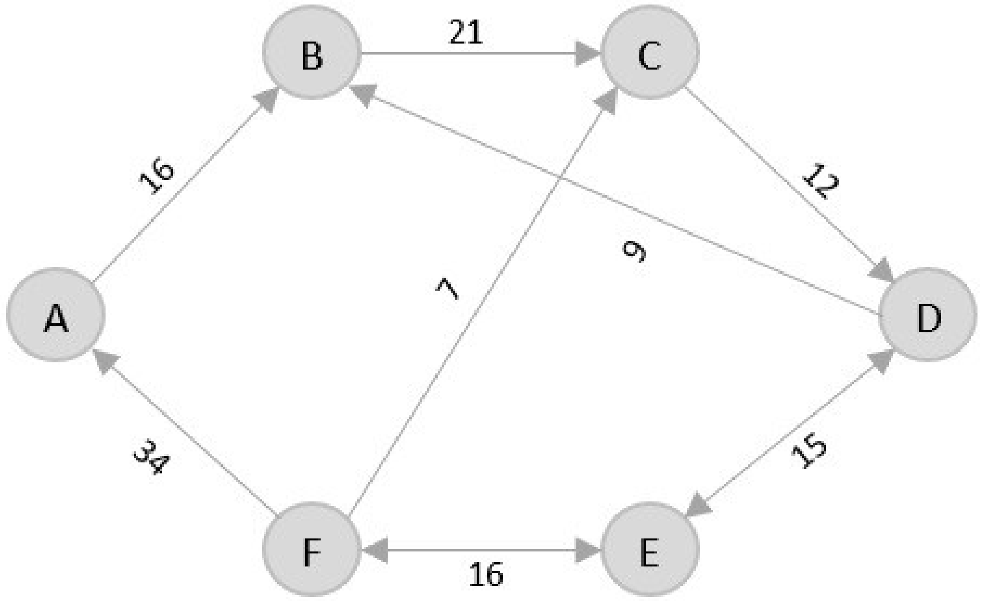
The travelling salesman problem

- If you look at the graph below, considering that the salesman starts from the vertex ‘a’, they need to travel through all the remaining vertices b, c, d, e, f and get back to ‘a’ while making sure that the cost taken is minimum.



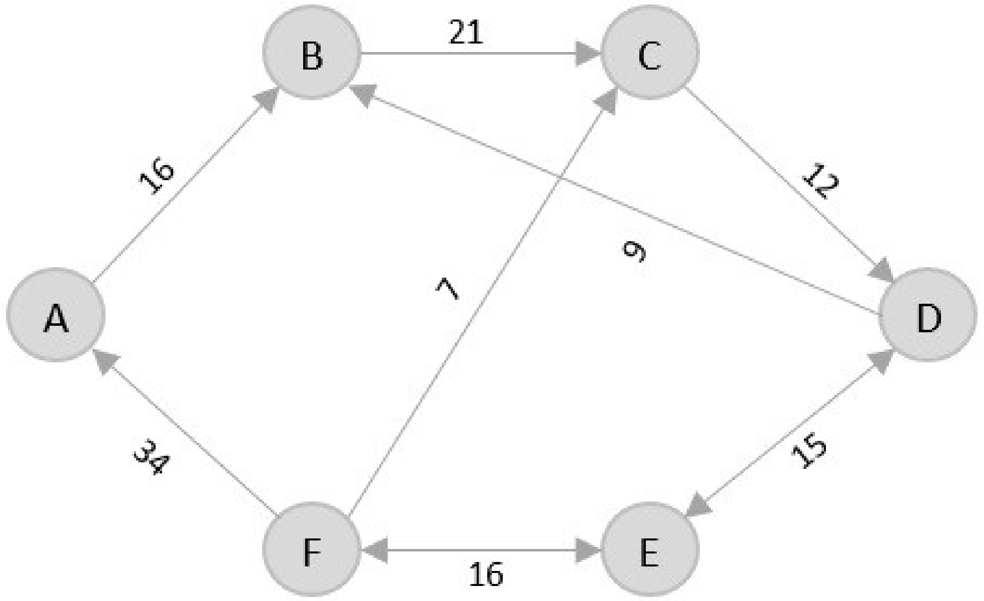
The travelling salesman problem

- Consider the following graph with six cities and the distances between them –



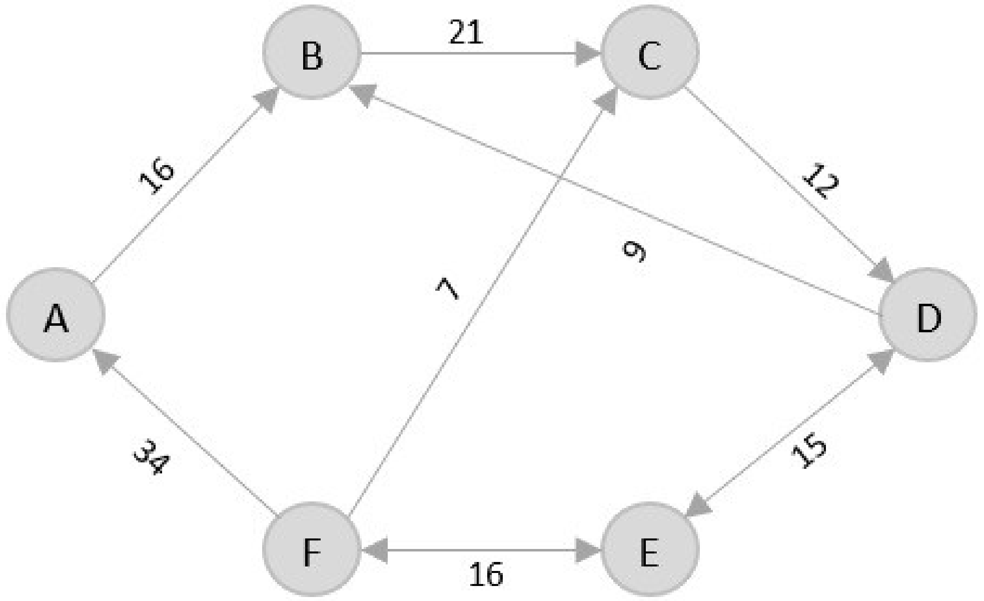
The travelling salesman problem

- Consider the following graph with six cities and the distances between them −



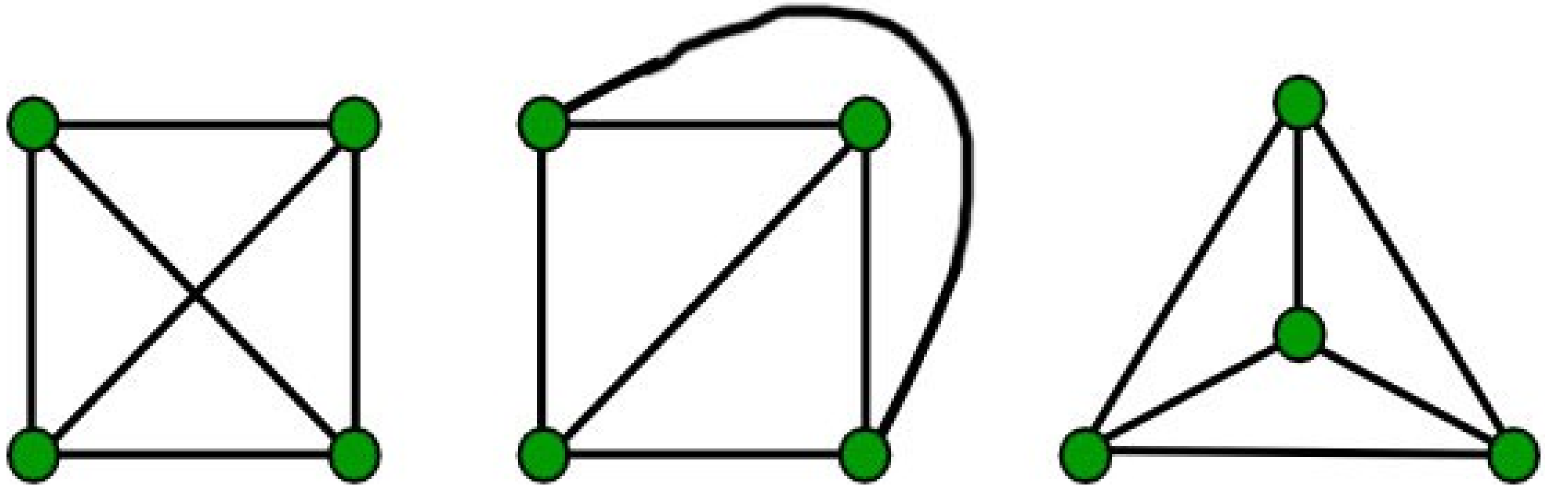
The travelling salesman problem

- Consider the following graph with six cities and the distances between them −



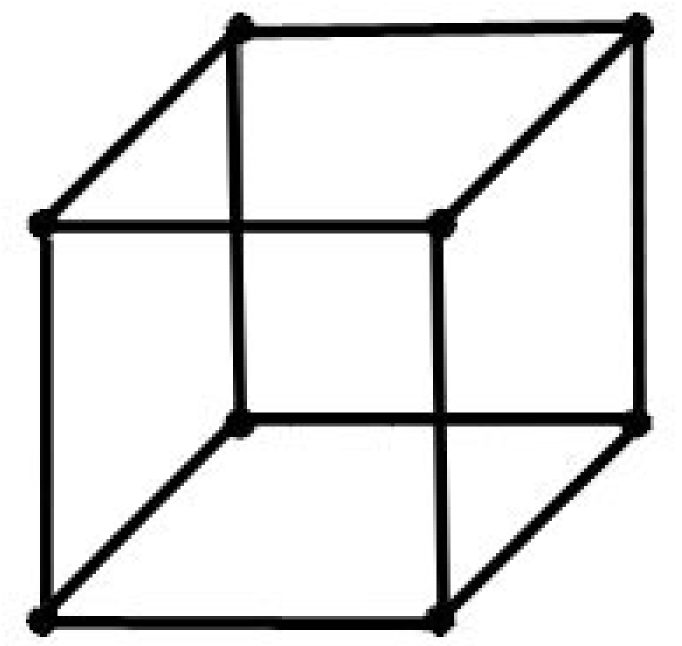
Graphs Coloring

- Planarity – "
- A graph is said to be planar if it can be drawn on a plane without any edges crossing.
- Such a drawing is called a planar representation of the graph."



Graphs Coloring

- Is the following graphs planner?



Graphs Coloring

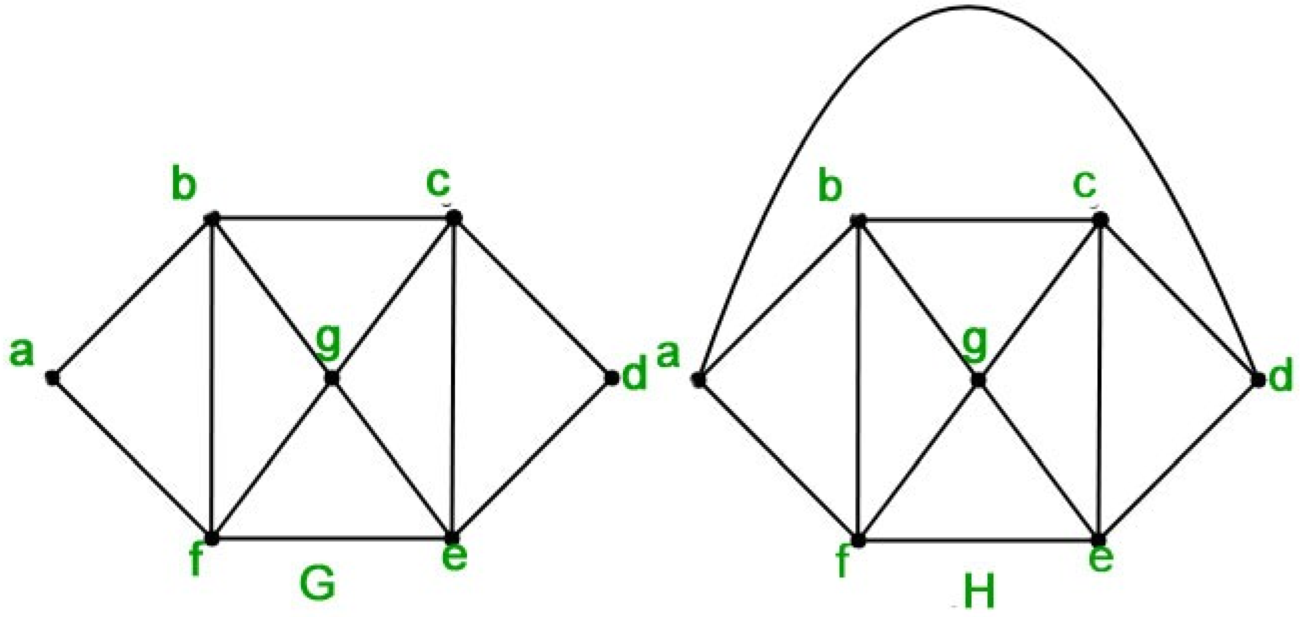
- Graph coloring
  - “A coloring of a simple graph is the assignment of a color to each vertex of the graph such that no two adjacent vertices are assigned the same color.”
  - The primary objective is to use as few colors as possible while satisfying this constraint.
  - This concept has various applications in scheduling, map coloring, register allocation in compilers, and more.

Graphs Coloring

- Graph coloring
  - chromatic number : The least number of colors required to color a graph is called its chromatic number.
  - It is denoted by X(G).
  - **4 color Theorem:**
    - “The chromatic number of a planar graph is no greater than 4.”

Graphs Coloring

- Graph coloring
  - What is the chromatic number of the following graphs and also color



Relations and Graphs

- References:
  - *Kenneth H. Rosen, Discrete mathematics and its applications, Seventh Edition McGraw Hill Publication, 2012.*