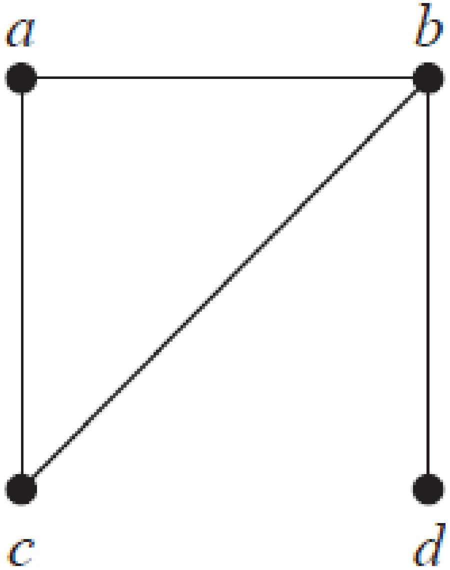
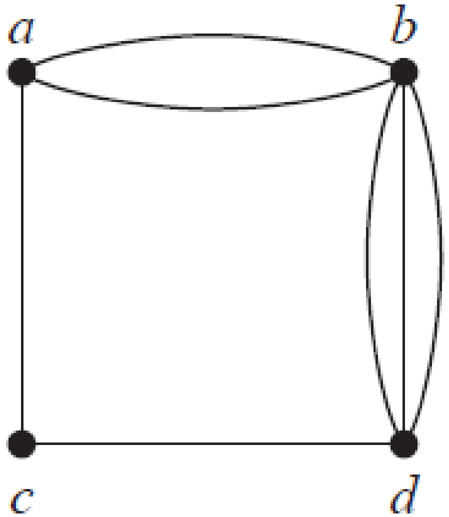
Discrete Mathematics

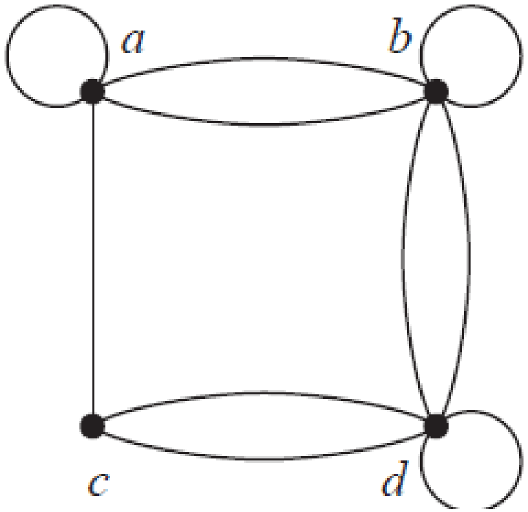
Graphs

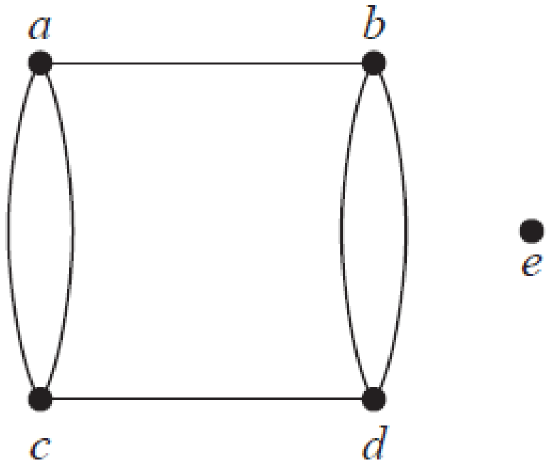
Srinivasan

A graph **G = (V ,E)** consists of **V** , a nonempty set of vertices (or nodes) and **E**, a set of edges. Each edge has two vertices associated with it, called its endpoints. An edge is said to connect its endpoints.

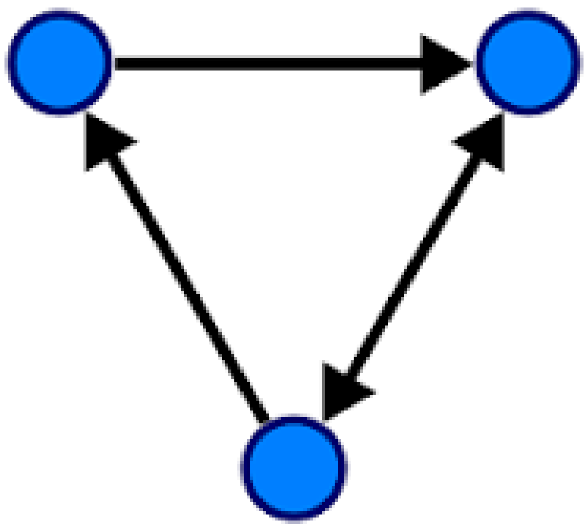


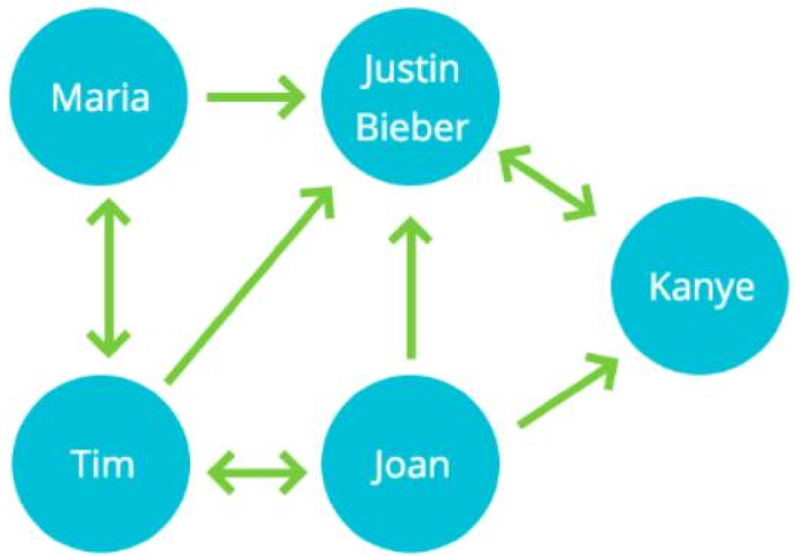






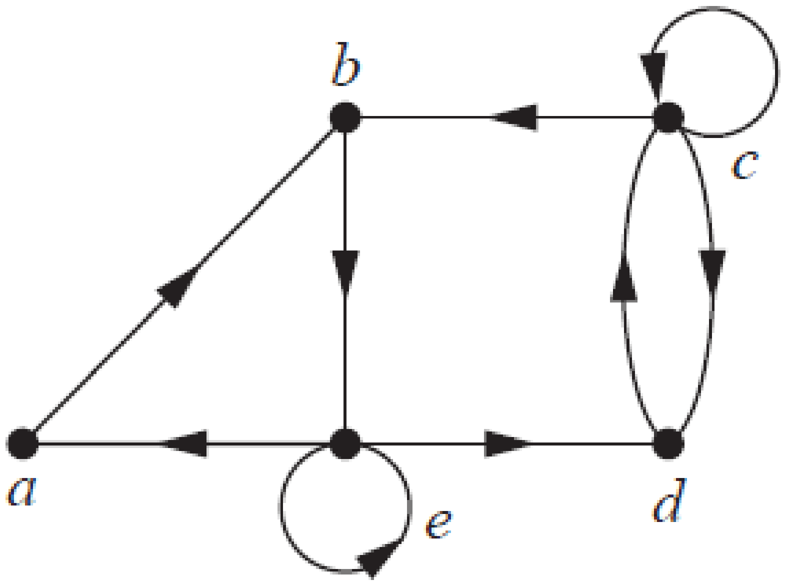
A directed graph (or digraph) (V ,E) consists of a nonempty set of vertices V and a set of directed edges (or arcs) E. Each directed edge is associated with an ordered pair of vertices. The directed edge associated with the ordered pair (u, v) is said to start at u and end at v.



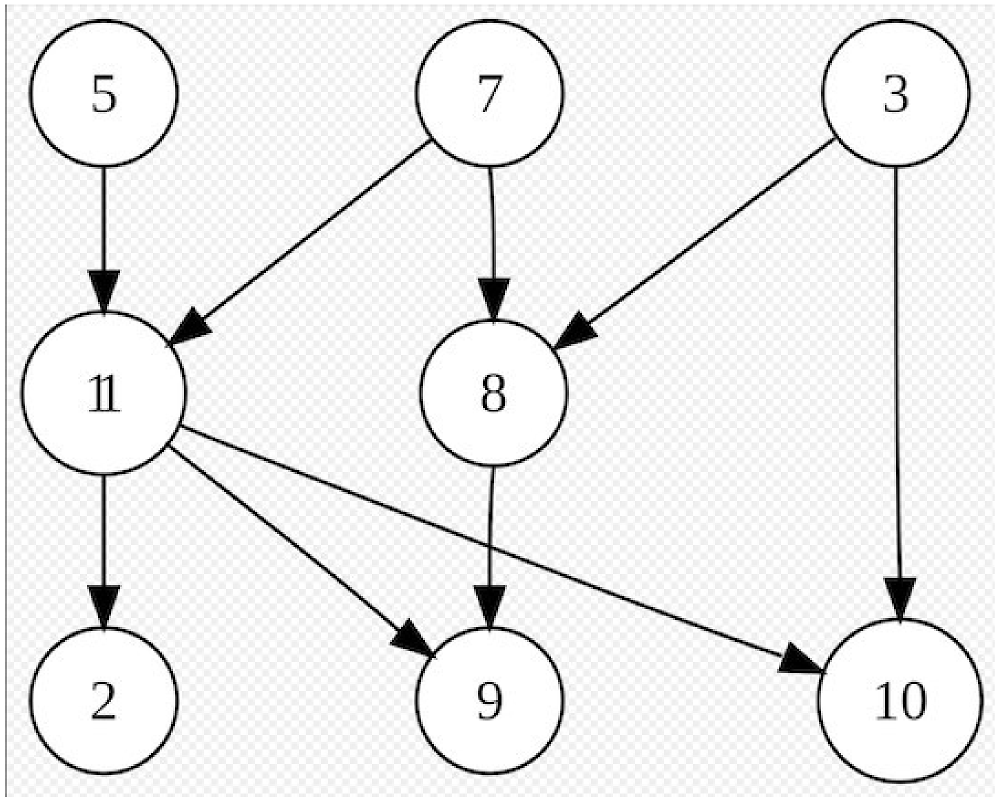


Instagram followers graph

A directed graph **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.



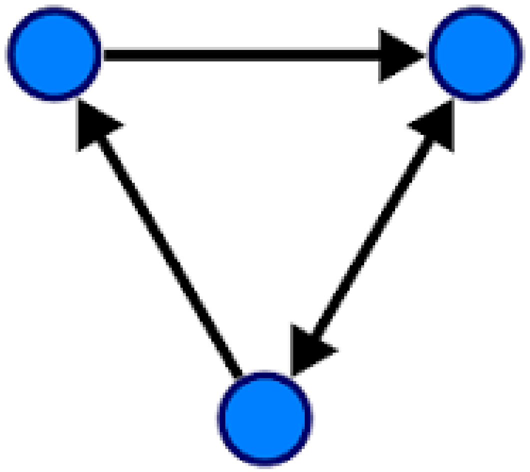
multiple edges, loops

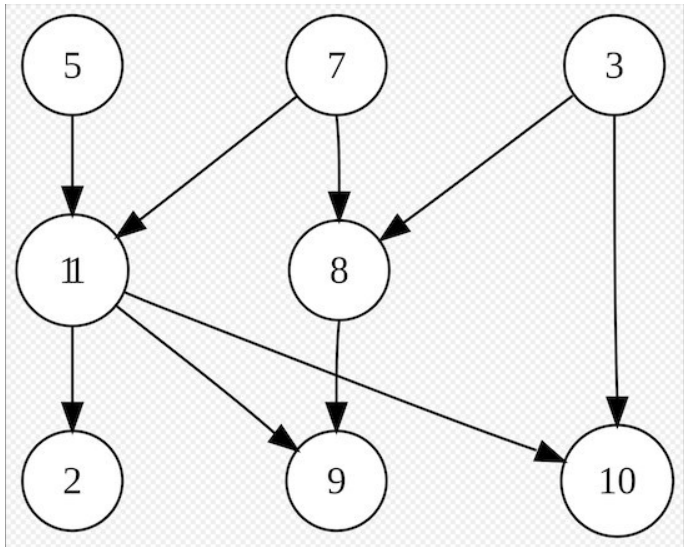


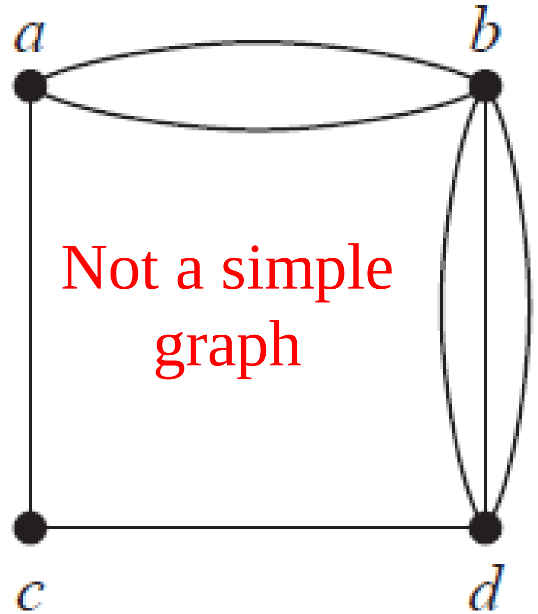
Simple directed graph

- A directed graph has no loops and has no multiple directed edges, it is called a **simple directed graph.**
- Because a simple directed graph has at most one edge associated to each ordered pair of vertices **(u, v)**,

we call **(u, v)** an edge if there is an edge associated to it in the graph.







Precedence Graphs and Concurrent Processing

- Computer programs can be executed more rapidly by executing certain statements concurrently.

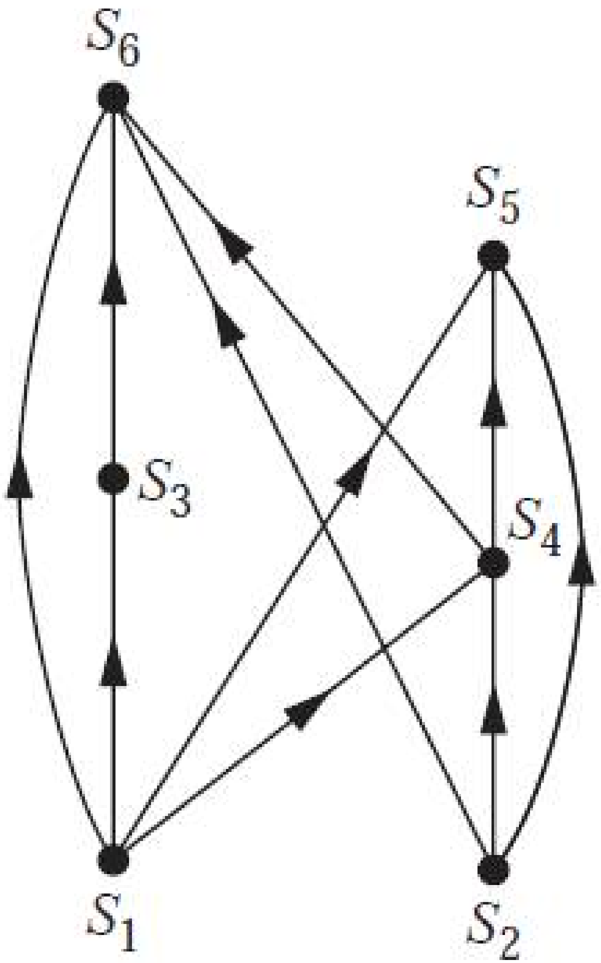
- It is important not to execute a statement that requires results of statements not yet executed.

- The dependence of statements on previous statements can be represented by a directed graph.

- Each statement is represented by a vertex, and there is an edge from one statement to a second statement if the second statement cannot be executed before the first statement.

Precedence Graphs and Concurrent Processing

S_1	$a := 0$
S_2	$b := 1$
S_3	$c := a + 1$
S_4	$d := b + a$
S_5	$e := d + 1$
S_6	$e := c + d$

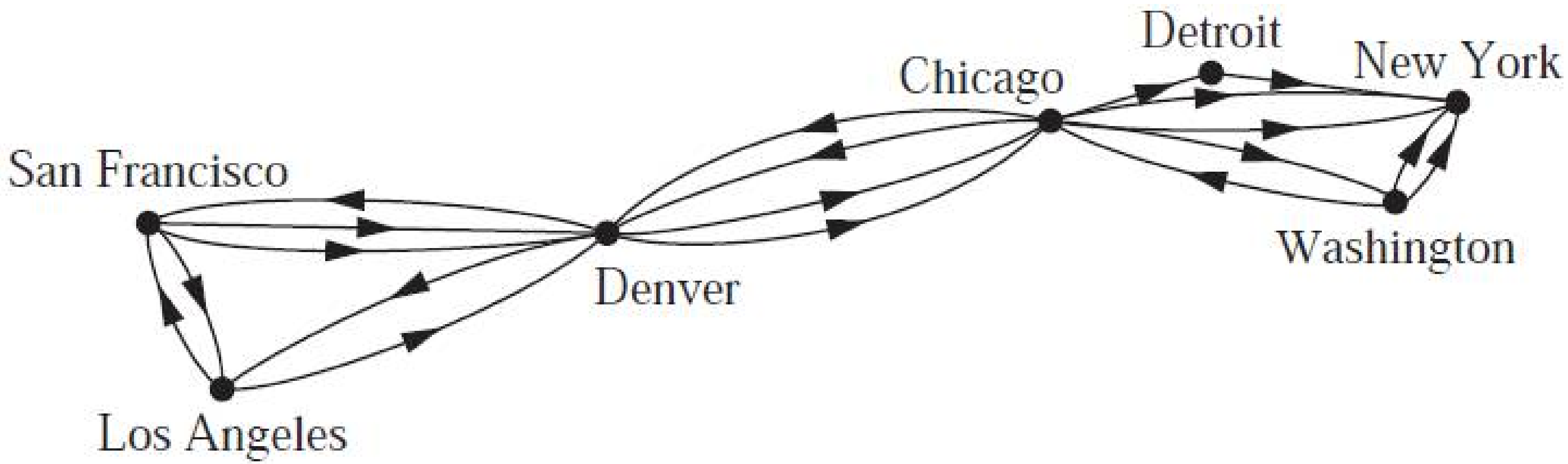


Precedence Graph

Directed multigraphs

In some computer networks, multiple communication links between two data centers may be present

multiple directed edges from a vertex to a second



A Computer Network with Multiple One-Way Links

Basic Terminology

Definition 1

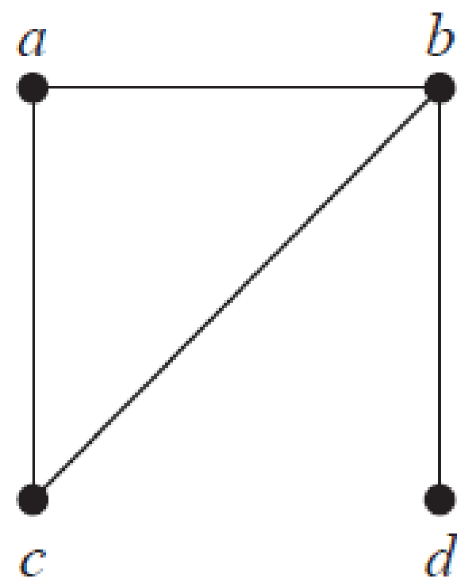
Two vertices *u* and *v* in an undirected graph *G* are called *adjacent* (or *neighbors*) in *G* if *u* and *v* are endpoints of an edge *e* of *G*.

Such an edge e is called incident with the vertices u and v and e is said to connect u and v.

Definition 2

The set of all neighbors of a vertex **v** of **G = (V, E)**, denoted by **N(v)**, is called the neighborhood of **v**.

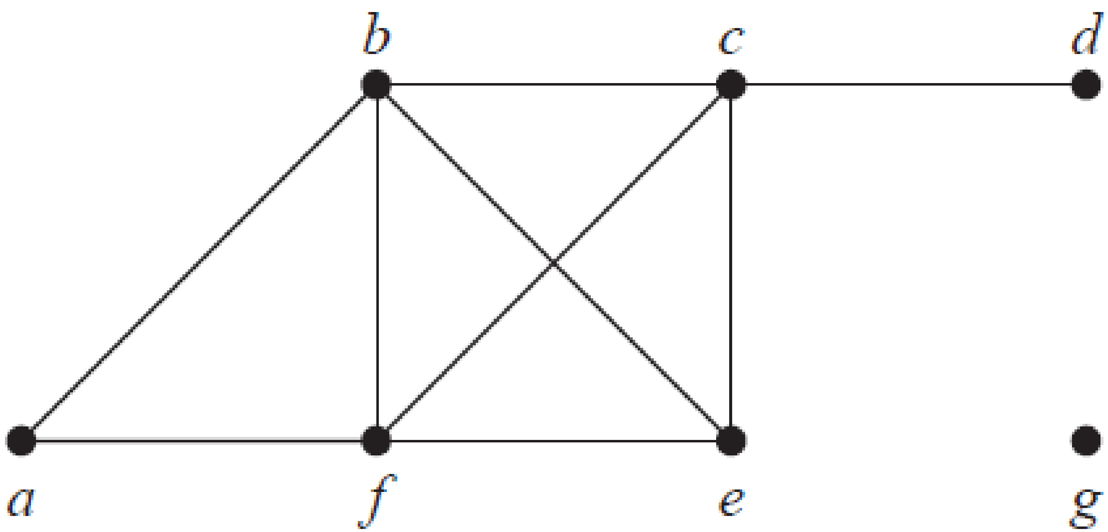
If **A** is a subset of **V**, we denote by **N(A)** the set of all vertices in **G** that are adjacent to at least one vertex in **A**

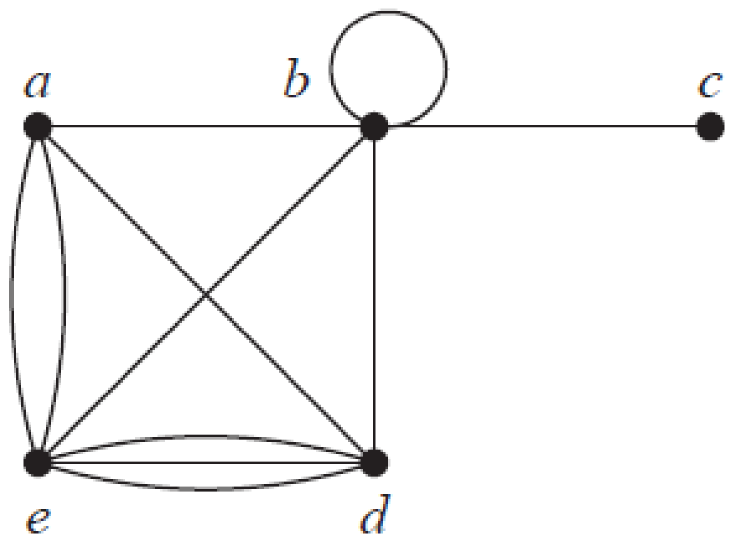


$$N(A) = \bigcup_{v \in A} N(v)$$

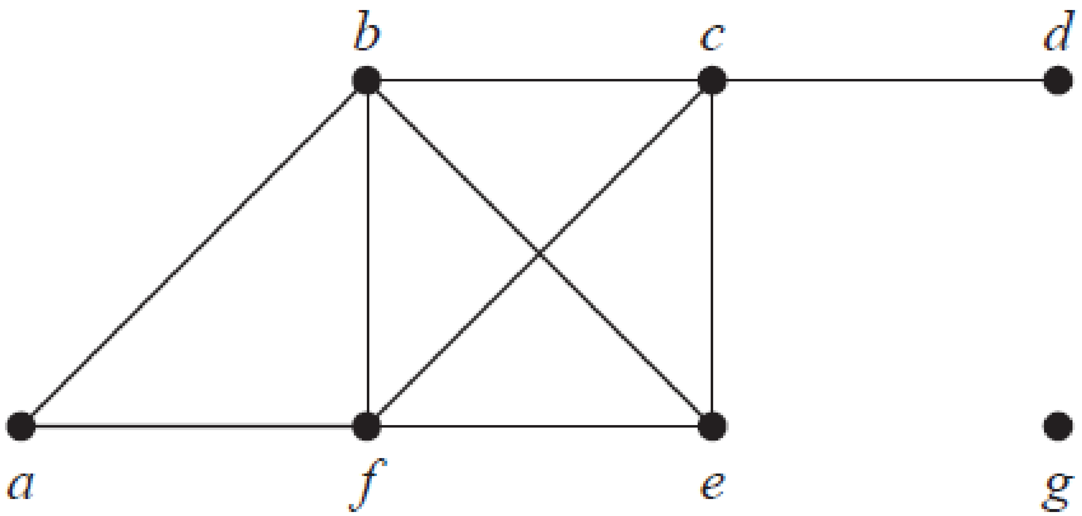
The ***degree of a vertex in an undirected graph*** is the number of edges incident with it, except that a loop at a vertex contributes twice to the degree of that vertex. The degree of the vertex ***v*** is denoted by **deg(v)**.

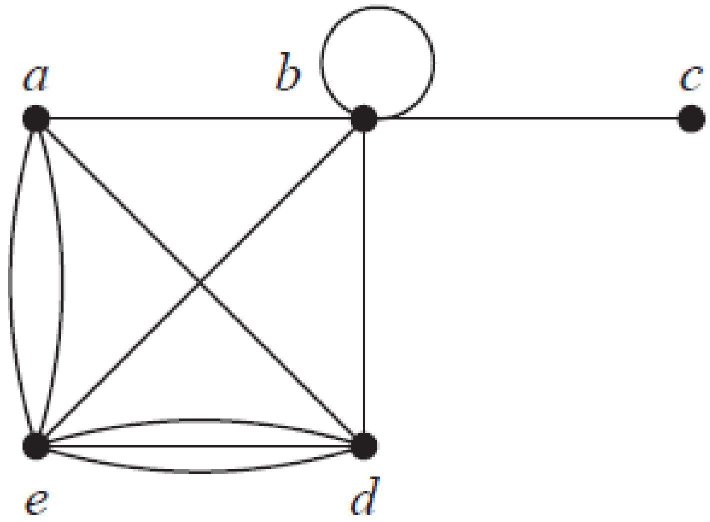
Problem: What are the degrees and what are the neighborhoods of the vertices in the graphs G and H displayed





Problem: **What are the degrees and what are the neighborhoods of the vertices in the graphs G and H displayed**





- ❑ **A vertex of degree zero is called isolated**
- ❑ **A vertex is pendant if and only if it has degree one**

THE HANDSHAKING THEOREM

Let G = (V, E) be an undirected graph with *m* edges. Then

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

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

Note: the sum of the degrees of the vertices of an undirected graph is even.

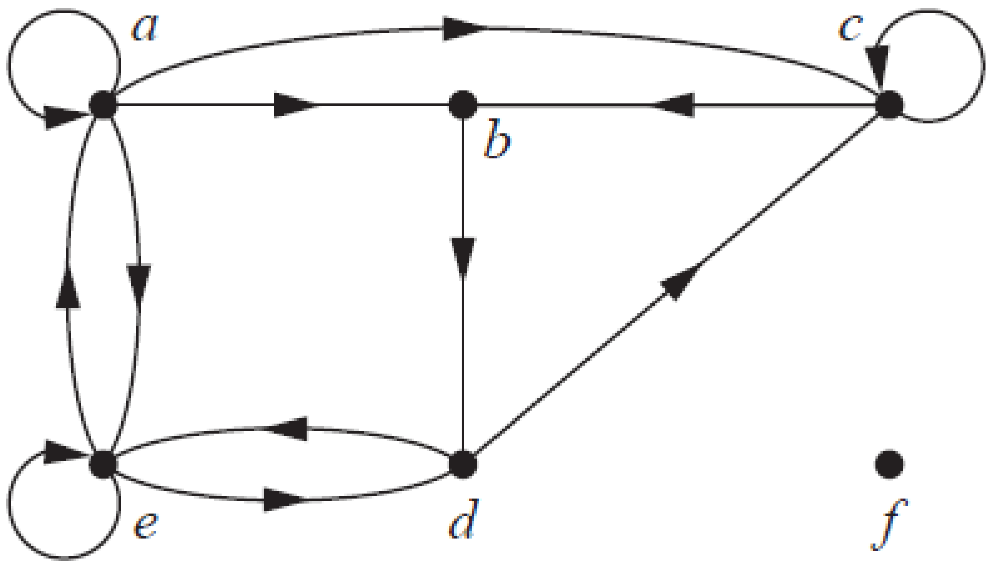
In-degree and out-degree of a vertex **Directed Graph**

In a graph with directed edges the *in-degree of a vertex* ***v***, denoted by **deg−(*v*),** is the number of edges with *v* as their terminal vertex.

The out-degree of ***v***, denoted by **deg+(v)**, is the number of edges with v as their initial vertex.

Problem:

Find the in-degree and out-degree of each vertex in the graph *G*



In-degree and out-degree of a vertex

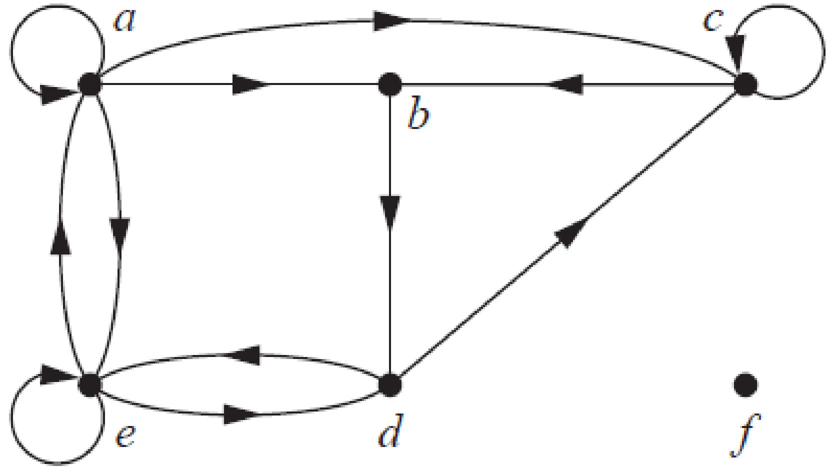
Directed Graph

THEOREM 3

Let $G = (V, E)$ be a graph with directed edges. Then

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

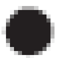
Verify the above theorem for this graph



Some Special Simple Graphs

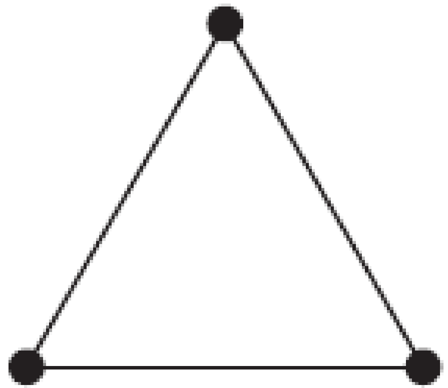
Complete Graphs: A complete graph on n vertices, denoted by K_n, is a simple graph that contains exactly one edge between each pair of distinct vertices.

The graphs K_n, for $n = 1, 2, 3, 4, 5, 6$ are displayed

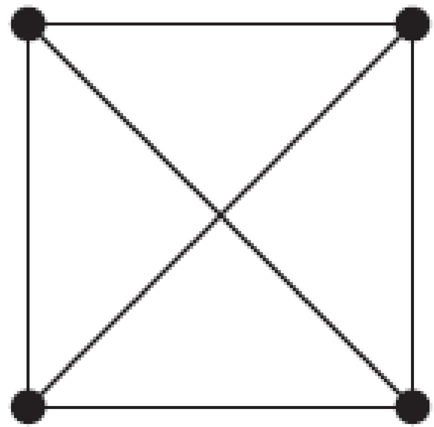


K_1

K_2



K_3

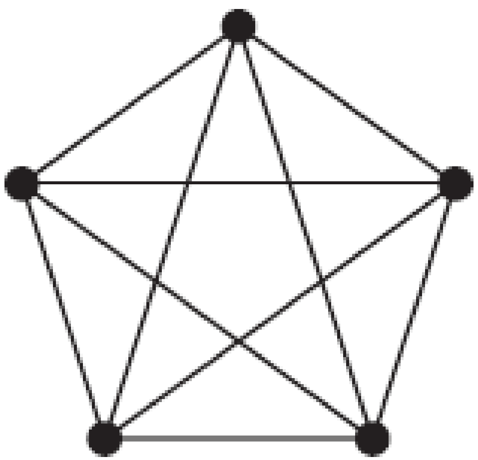


K_4

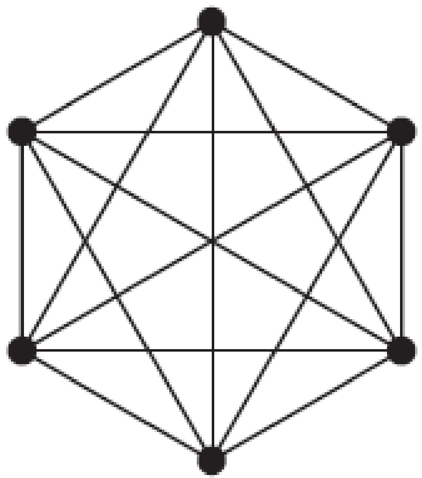
Some Special Simple Graphs

Complete Graphs: A complete graph on n vertices, denoted by K_n, is a simple graph that contains exactly one edge between each pair of distinct vertices.

The graphs K_n, for n = 1, 2, 3, 4, 5, 6 are displayed



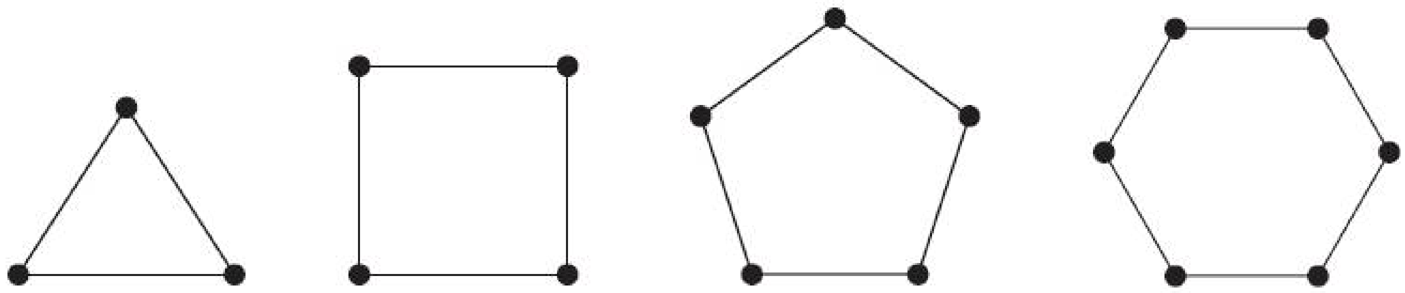
K_5



K_6

Cycles in Graph

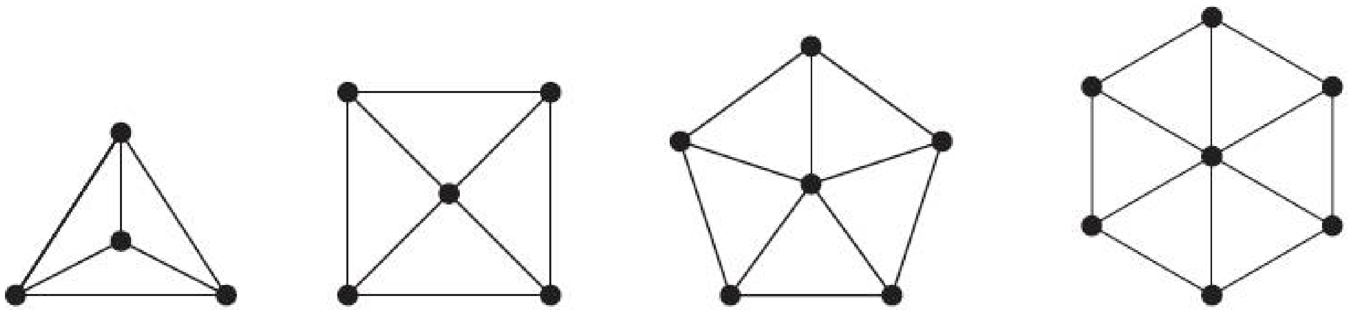
Cycles A **cycle** C_n, $n \geq 3$, consists of n vertices $v_1, v_2, \ldots, v_n$ and edges $\{v_1, v_2\}$, $\{v_2, v_3\}, \ldots, \{v_{n-1}, v_n\}$, and $\{v_n, v_1\}$. The cycles C_3, C_4, C_5, and C_6 are displayed in Figure 4. ◀



C_3 C_4 C_5 C_6

Wheels in Graph

Wheels We obtain a **wheel** W_n when we add an additional vertex to a cycle C_n, for $n \geq 3$, and connect this new vertex to each of the n vertices in C_n, by new edges. The wheels W_3, W_4, W_5, and W_6 are displayed in Figure 5.

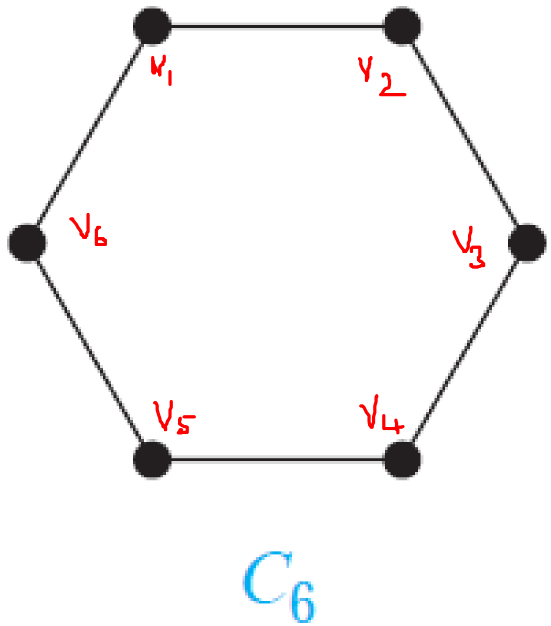


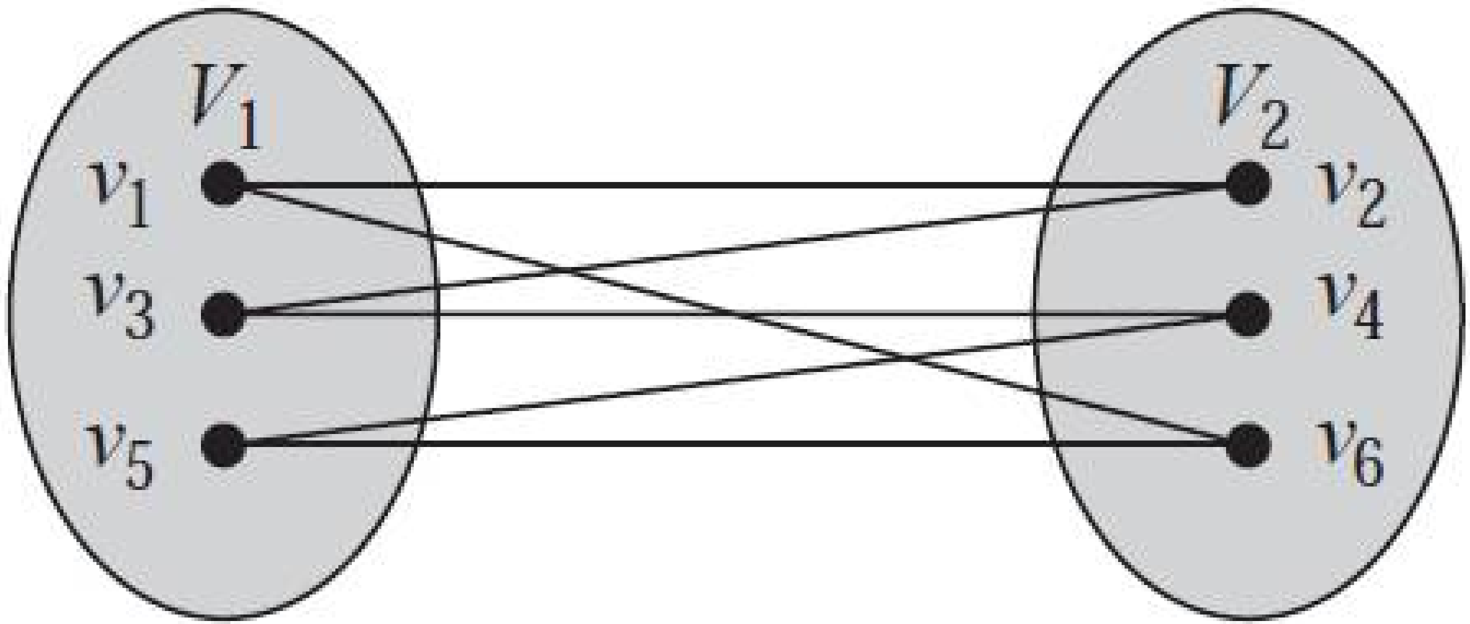
W_3 W_4 W_5 W_6

Bipartite Graphs

A simple graph **G** is called bipartite if its vertex set **V** can be partitioned into two disjoint sets $\mathbf{V_1}$ and $\mathbf{V_2}$ such that every edge in the graph connects a vertex in $\mathbf{V_1}$ and a vertex in $\mathbf{V_2}$

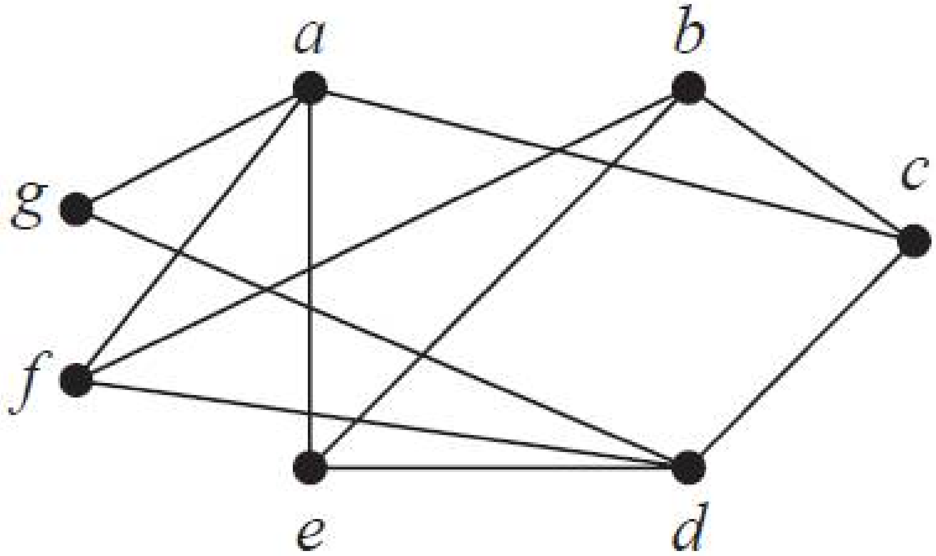
We call the pair $\mathbf{(V_1, V_2)}$ a bipartition of the vertex set **V** of **G**

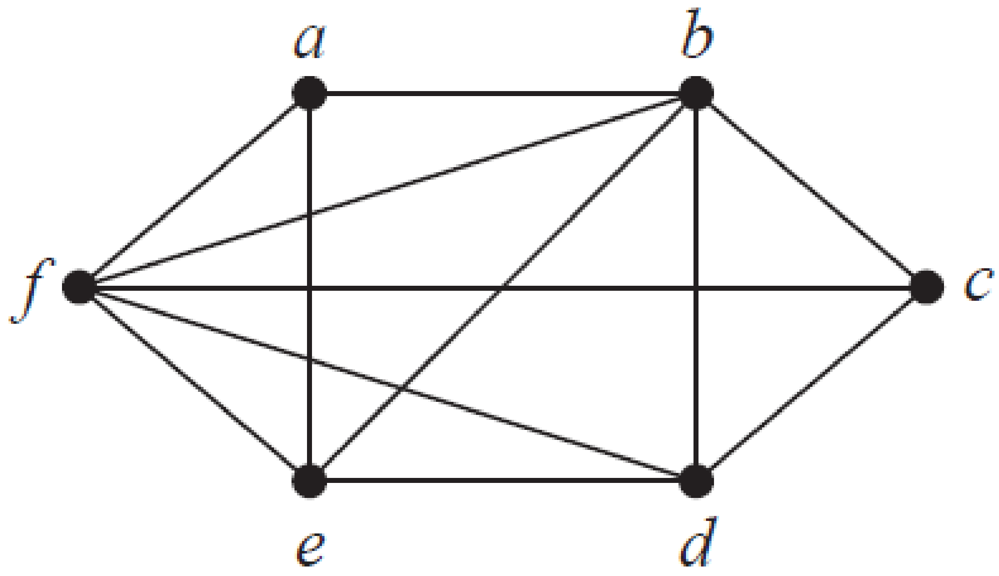




Bipartite Graphs

Problem: Are the graphs below bipartite?



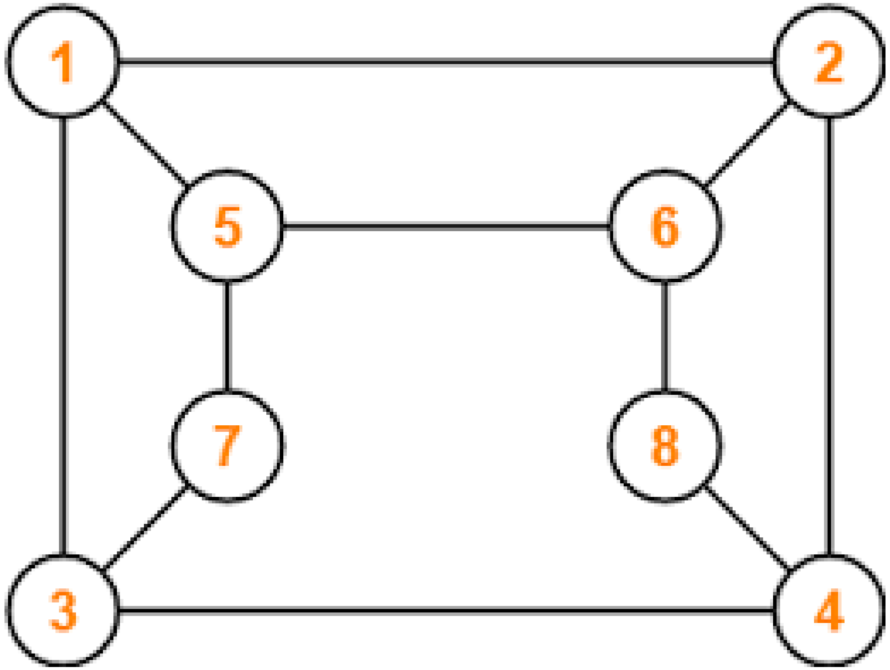


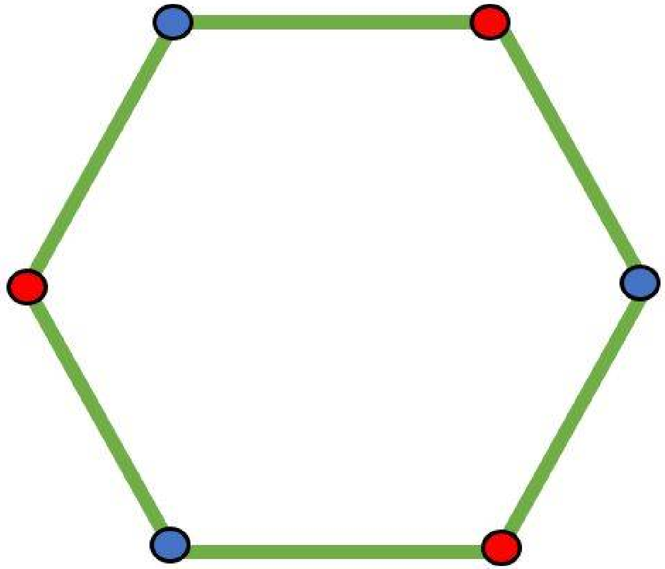
Disjoint sets
{a, b, d} and {c, e, f, g}

Graph *H* is not bipartite

Theorem 4 (*text page number* 657)

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.

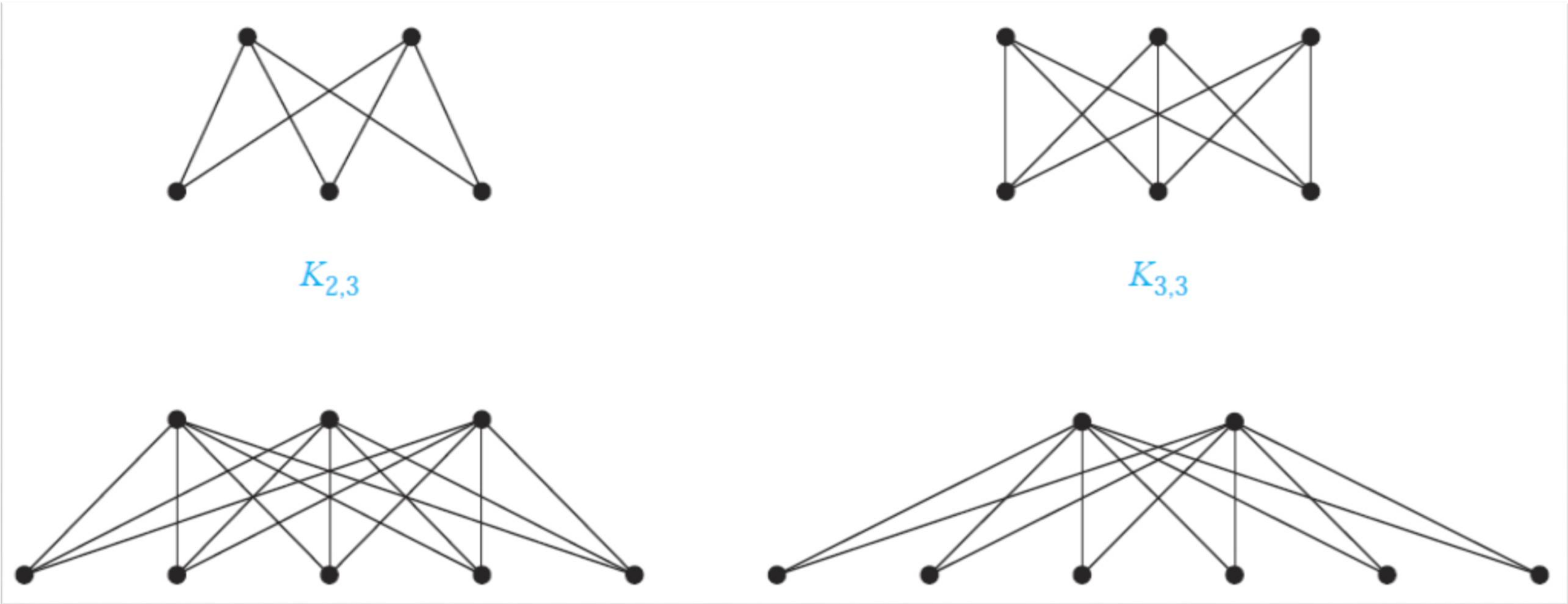




Cycle graph of length 6

Complete Bipartite Graphs

A complete bipartite graph $K_{m,n}$



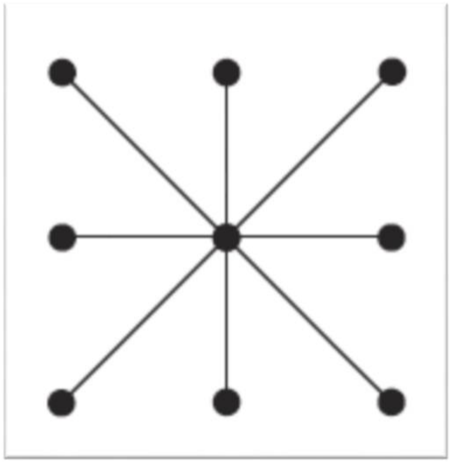
Some Applications of Special Types of Graphs

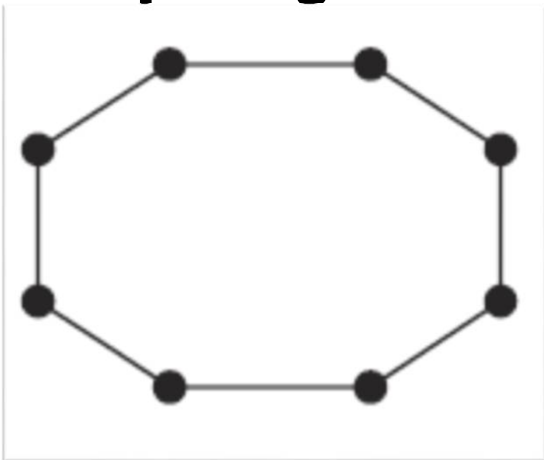
Local Area Networks

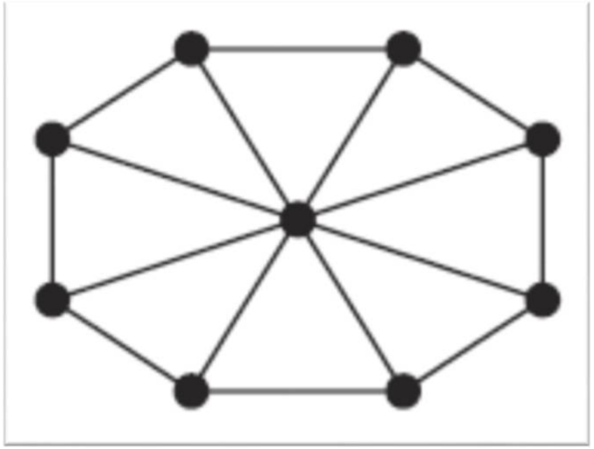
Minicomputers, personal computers, printers, routers, can be connected using a *local area network*.

Some of these networks are based on a

Star, Ring, and Hybrid Topologies



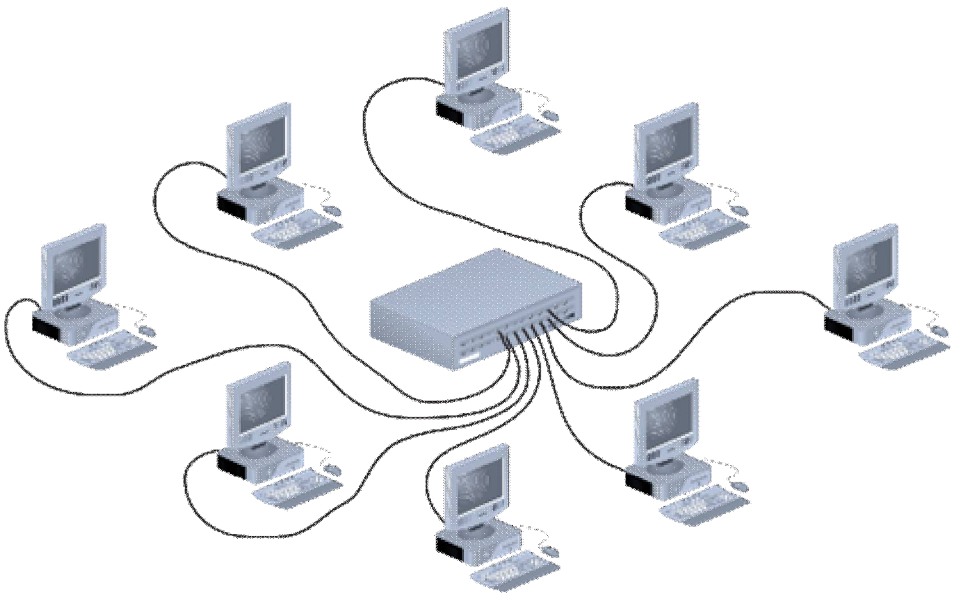


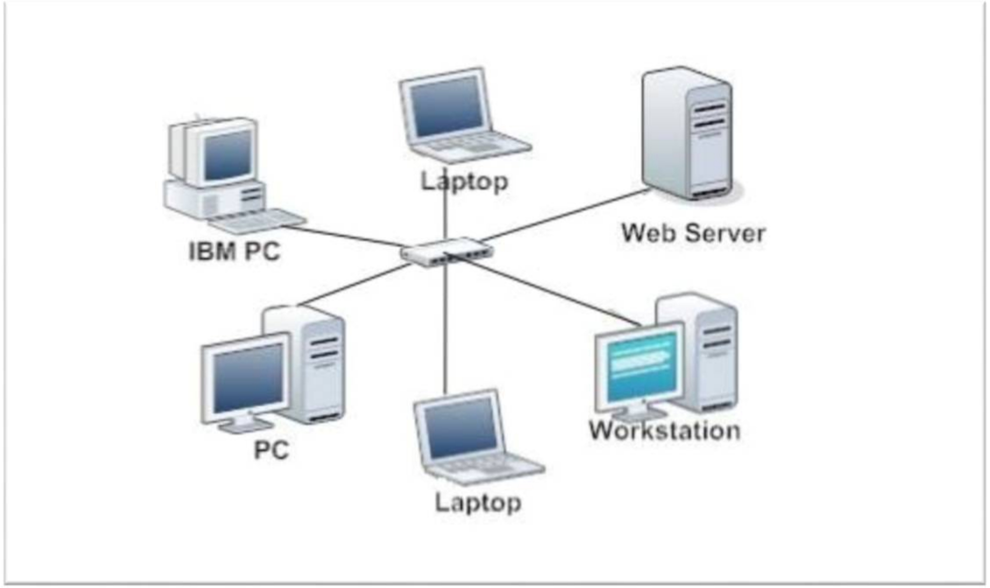


Some Applications of Special Types of Graphs

Local Area Networks

Star topology

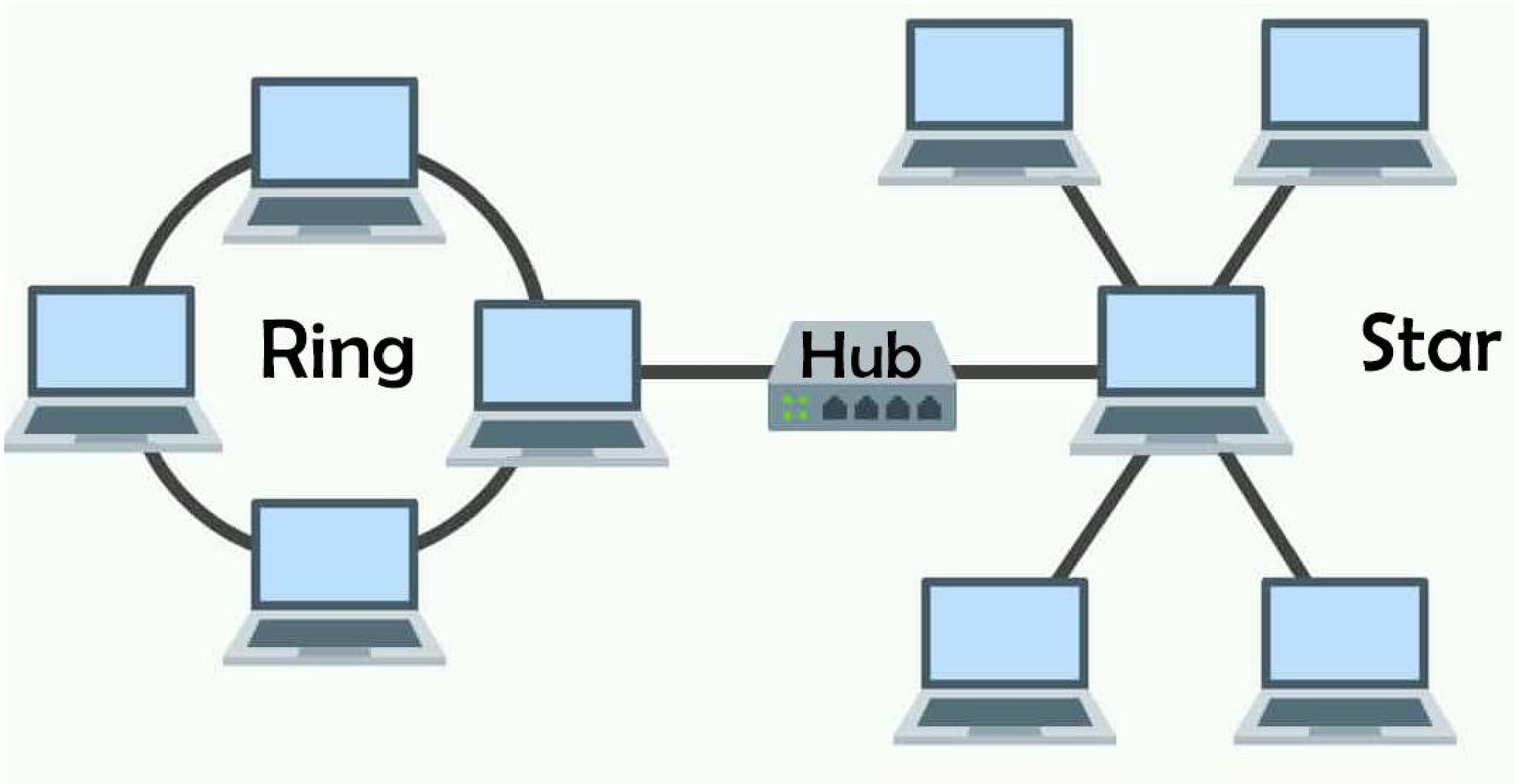




Some Applications of Special Types of Graphs

Local Area Networks

Hybrid topology



Representing Graphs - There are many useful ways to represent graphs

- represent a graph without multiple edges is to list all the edges
- represent a graph with no multiple edges is to use **adjacency lists**

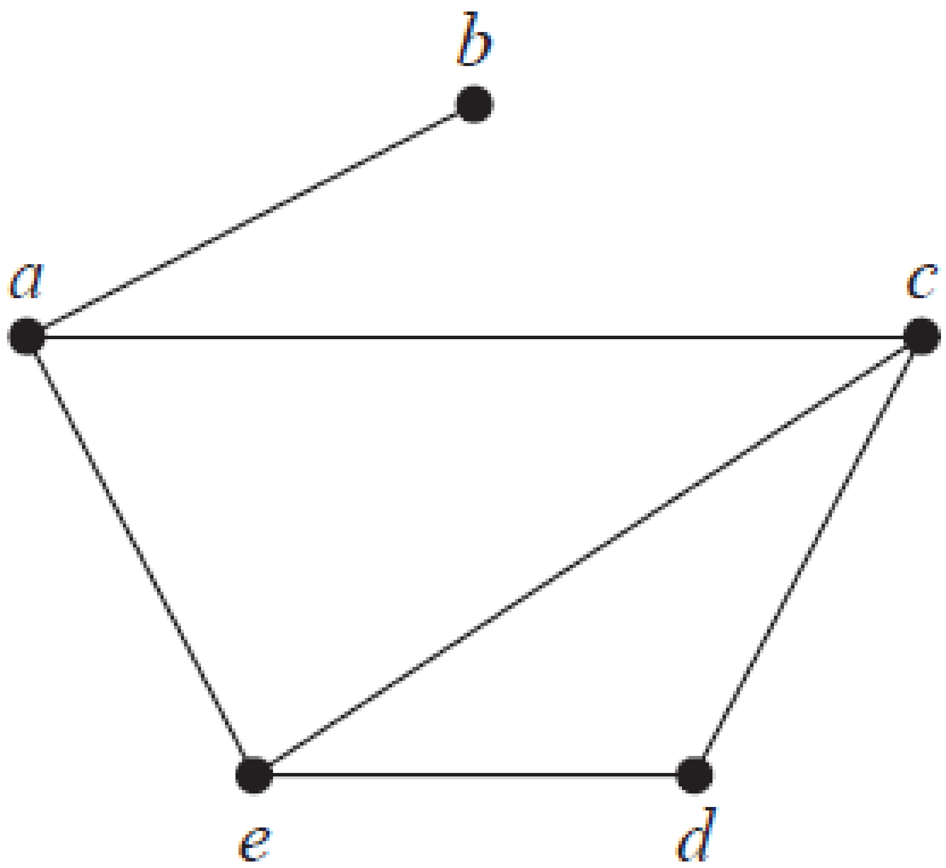


TABLE 1 An Adjacency List for a Simple Graph.

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

Represent a graph with no multiple edges is to use adjacency lists

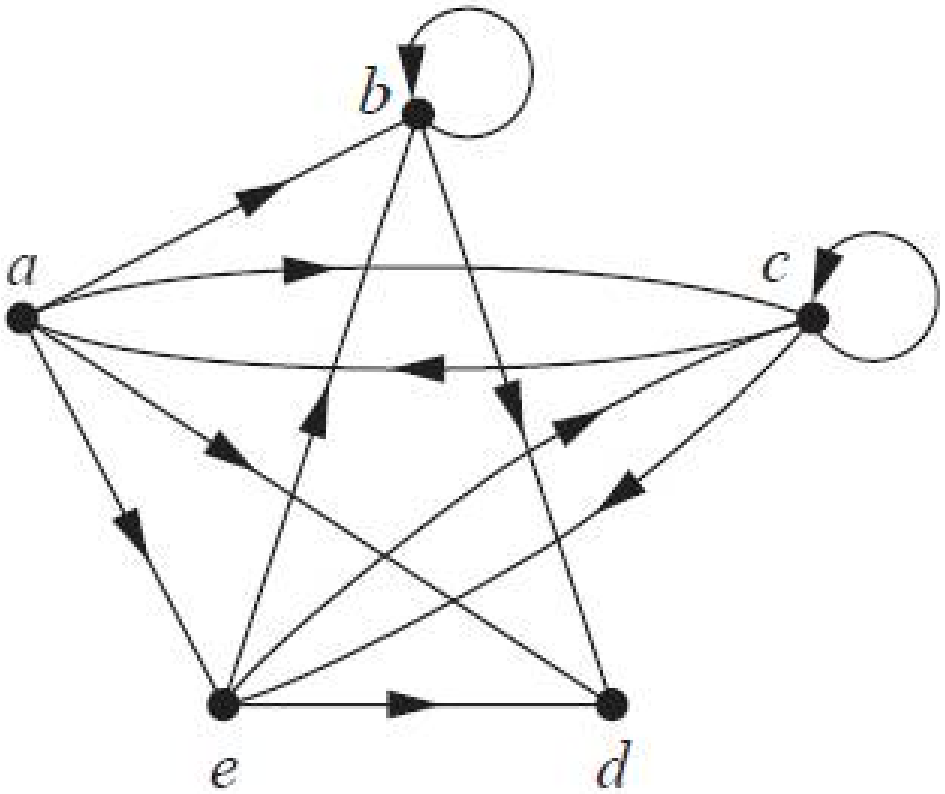


TABLE 2 An Adjacency List for a Directed Graph.

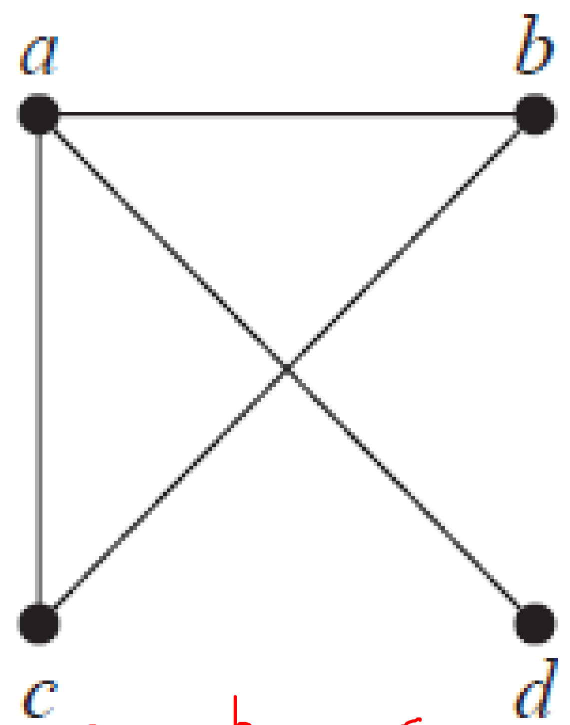
Initial Vertex	*Terminal Vertices*
a	b, c, d, e
b	b, d
c	a, c, e
d	
e	b, c, d

Adjacency Matrices

G = (V ,E) is a simple graph where |V| = n

Adjacency matrix is **A** = [$\boldsymbol{a_{ij}}$] defined as

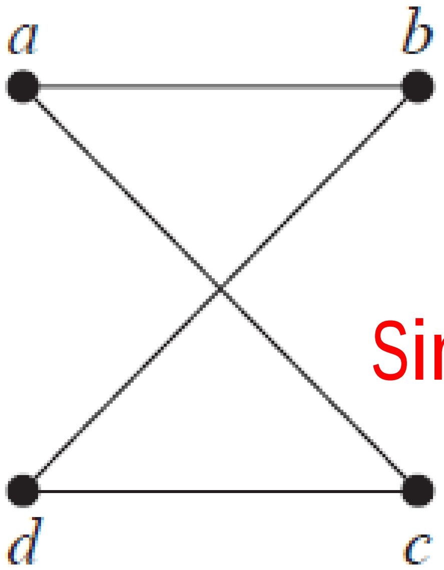
$$a_{ij} = \begin{cases} 1 & \text{if } \{v_i, v_j\} \text{ is an edge of } G, \\ 0 & \text{otherwise.} \end{cases}$$



$$\begin{array}{c|c} & \begin{matrix} a & b & c & d \end{matrix} \\ \begin{matrix} a \\ b \\ c \\ d \end{matrix} & \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

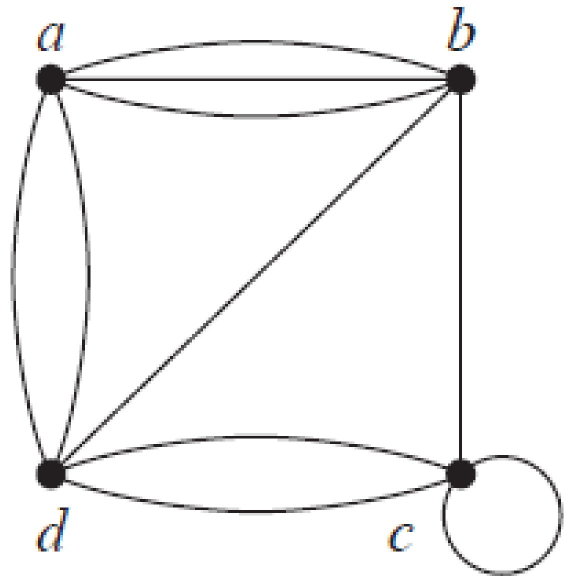
Symmetric matrix

EXAMPLE Draw a graph with the adjacency matrix



Simple graph

Multi graph



$$\begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 0 & 3 & 0 & 2 \\ 3 & 0 & 1 & 1 \\ 0 & 1 & 1 & 2 \\ 2 & 1 & 2 & 0 \end{bmatrix}$$

Incidence Matrices

represent graphs is to use incidence matrices

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

$v_1, v_2, \ldots, v_n$ are the vertices and $e_1, e_2, \ldots, e_m$ are the edges

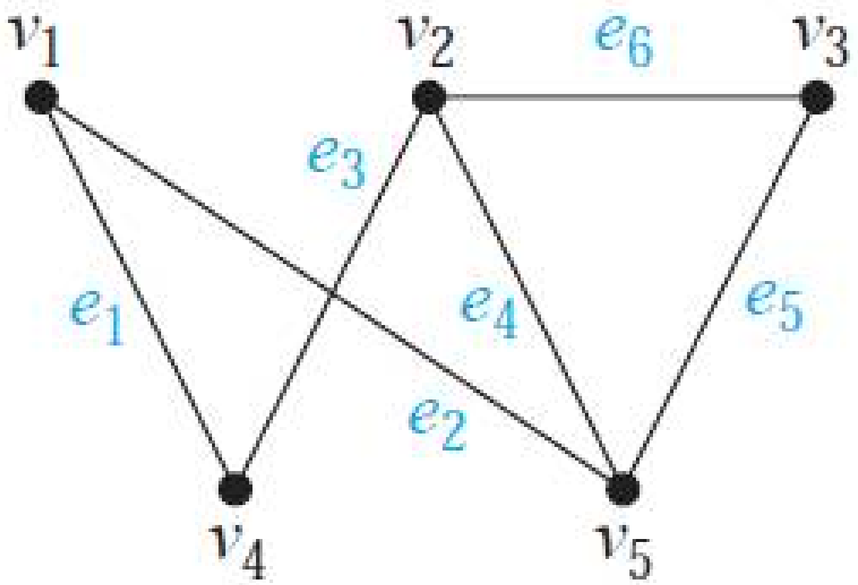
The incidence matrix M = [m_{ij}], is an $n \times m$ matrix

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

Incidence Matrices

Represent graphs using incidence matrices

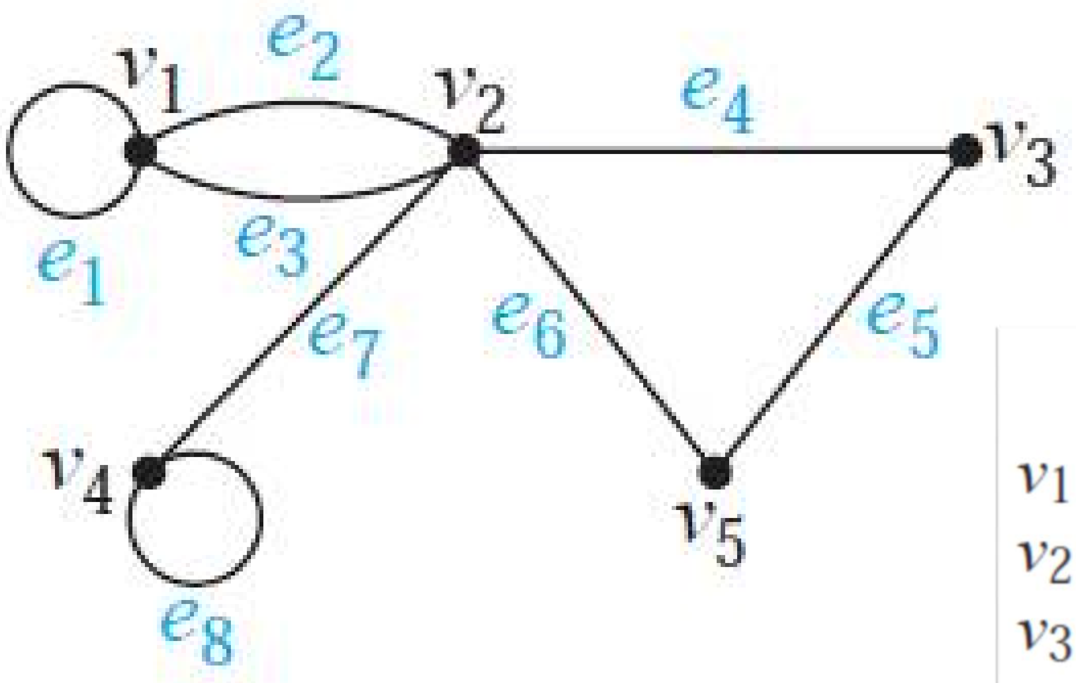
$$m_{ij} = \begin{cases} 1 & \text{when edge } e_j \text{ is incident with } v_i, \\ 0 & \text{otherwise.} \end{cases}$$



$$\begin{array}{c c} & \begin{matrix} e_1 & e_2 & e_3 & e_4 & e_5 & e_6 \end{matrix} \\ \begin{matrix} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \end{matrix} & \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 \end{bmatrix} \end{array}$$

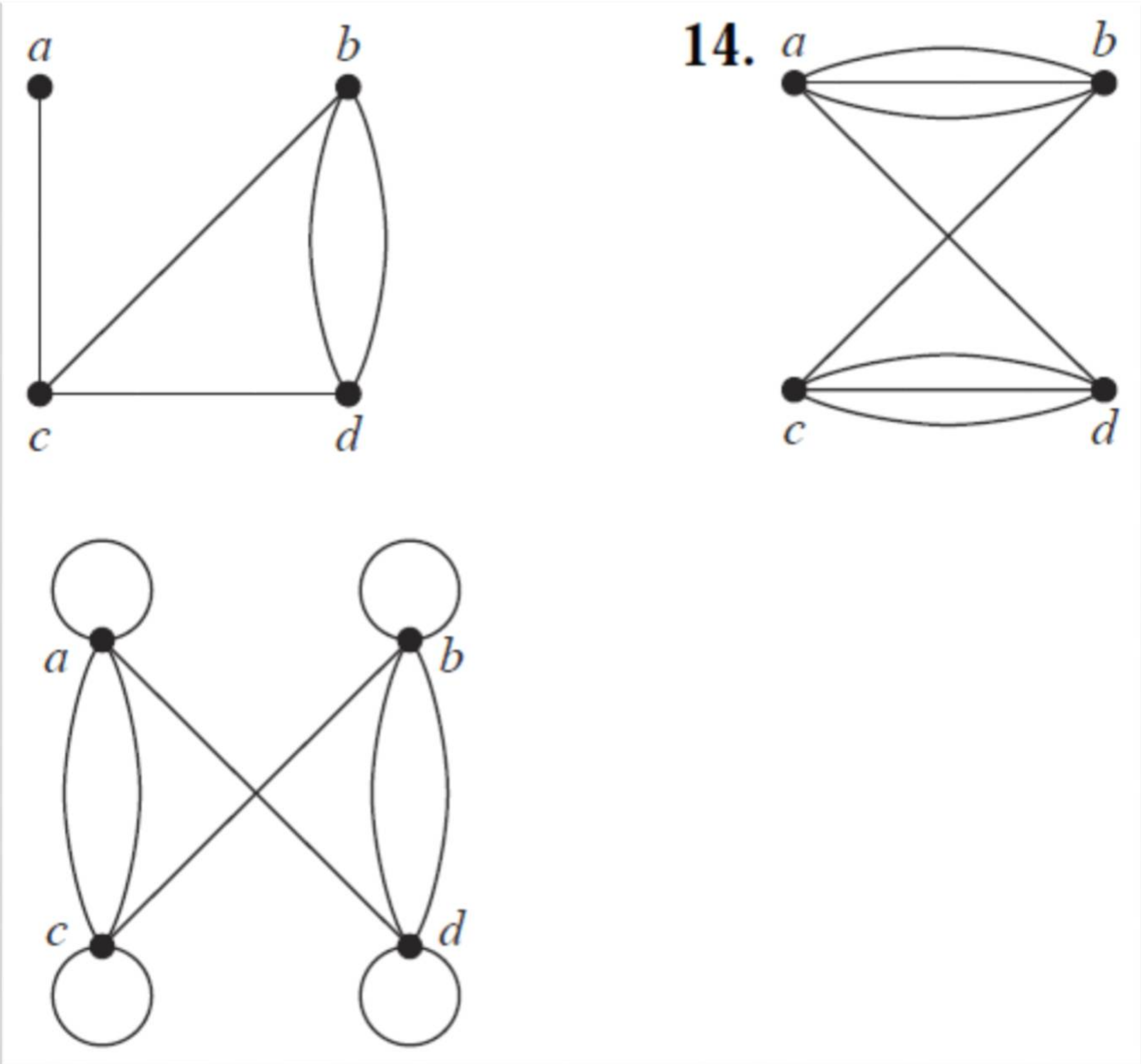
Incidence Matrices

Represent the pseudograph (or multigraph) using an incidence matrix.



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

Incidence Matrices - Find the incidence matrix of

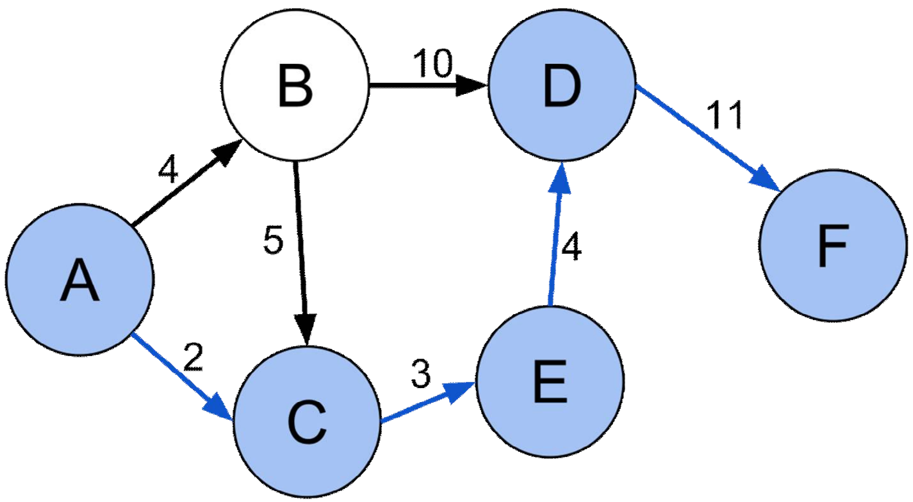


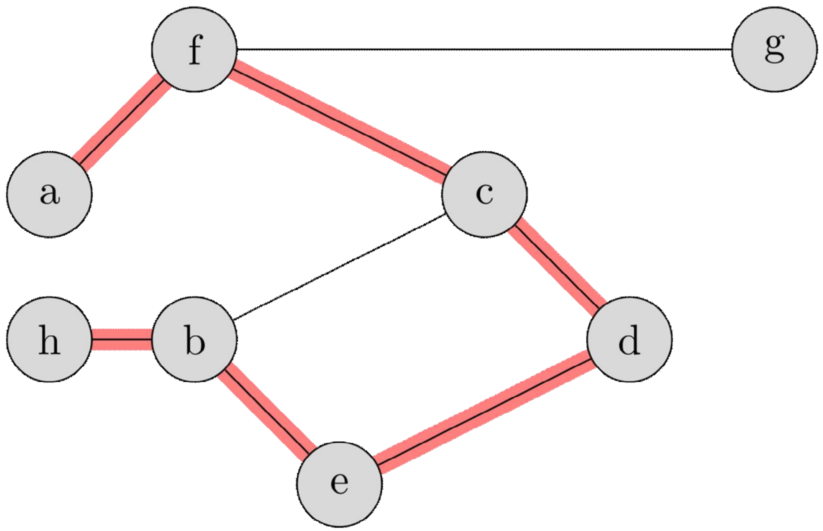
Connectivity

Many problems can be modeled with paths formed by traveling along the edges of graphs

Paths

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.





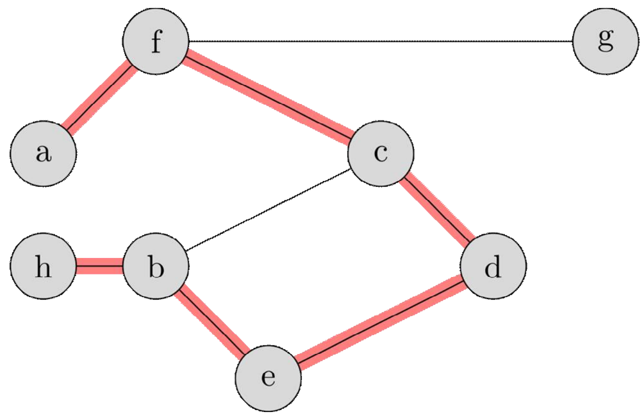
Connectivity

Paths: **A path of length n from u to v in G is a sequence of n edges $e_1, . . . , e_n$ of G for which there exists a sequence $x_0 = u, x_1, . . . , x_{n-1}, x_n = v$ of vertices**

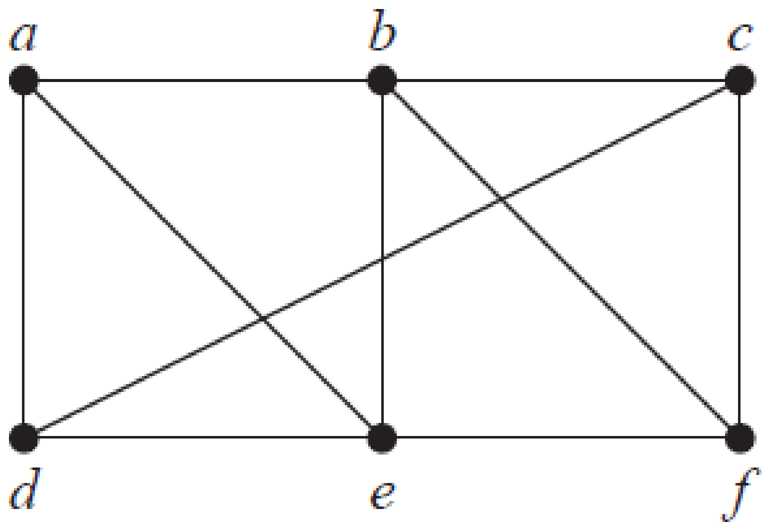
The path is a *circuit* if it begins and ends at the same vertex, that is, if $u = v$

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

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



Example: Paths and circuits in of simple graph



a, *d*, *c*, *f* , *e* is a simple path of length 4, because

{*a*, *d*}, {*d*, *c*}, {*c*, *f* }, and {*f*, *e*} are all edges.

d, e, c, a is not a path, because {e, c} is not an edge.

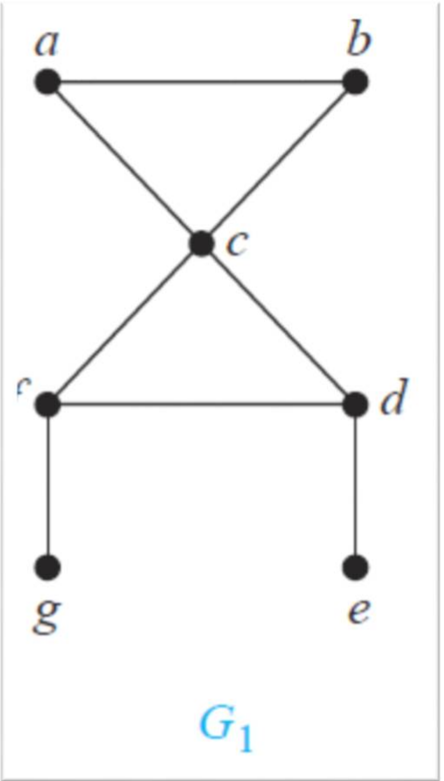
b, c, f , e, b is a circuit of length 4 because {b, c}, {c, f }, {f, e}, and {e, b} are edges

a, b, e, d, a, b, which is of length 5, is not simple because it contains the edge {a, b} twice.

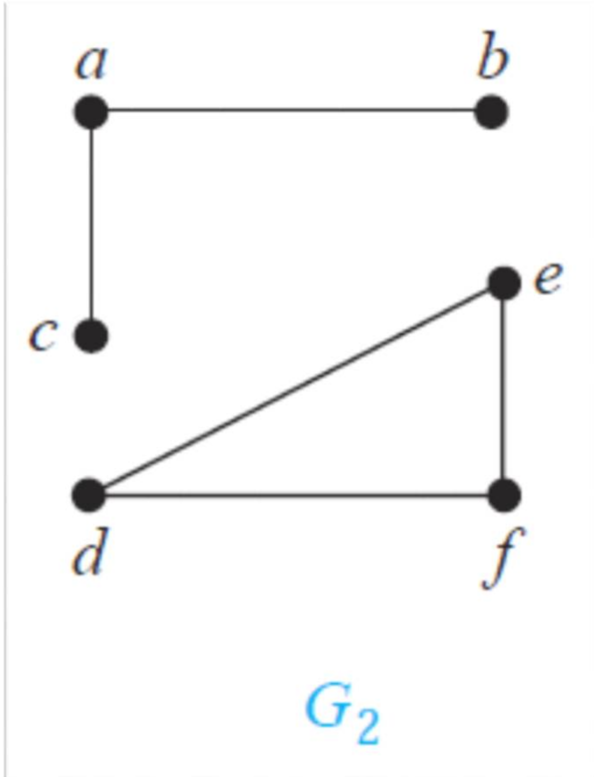
Connected Graph

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



The graph G_1 is connected and G_2 is not connected



How Connected is a Graph?

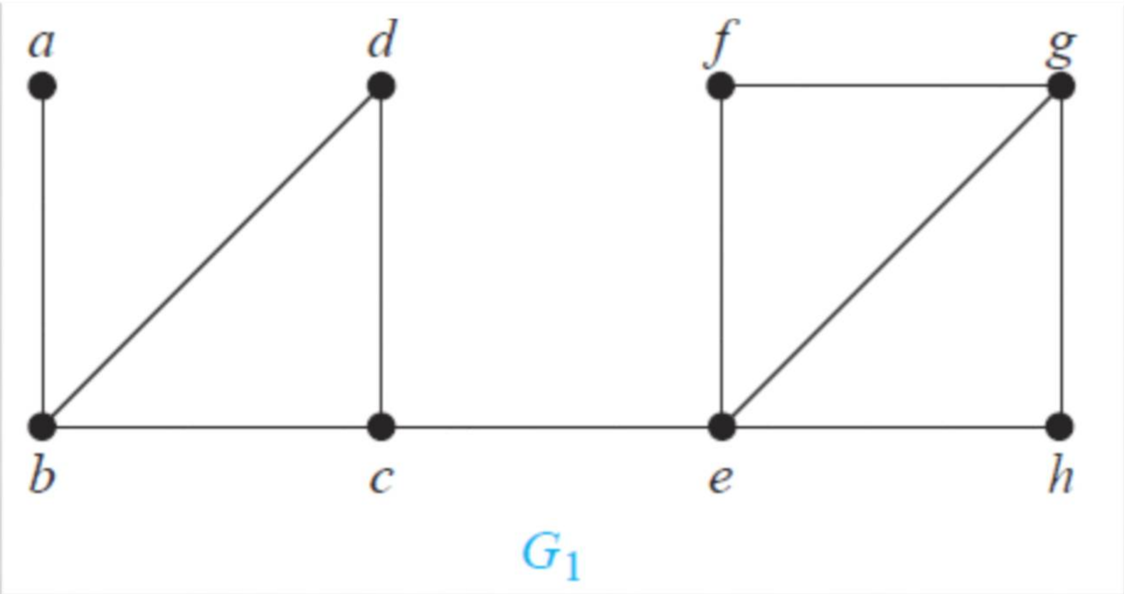
In computer network, will it still be possible for all computers to communicate after a router or a communications link fails?

Removal from a graph of a vertex and all incident edges produces a subgraph. **Such vertices are called cut vertices**

Similarly we define for cut edge or bridge

How Connected is a Graph?

Find the cut vertices and cut edges in the graph G_1

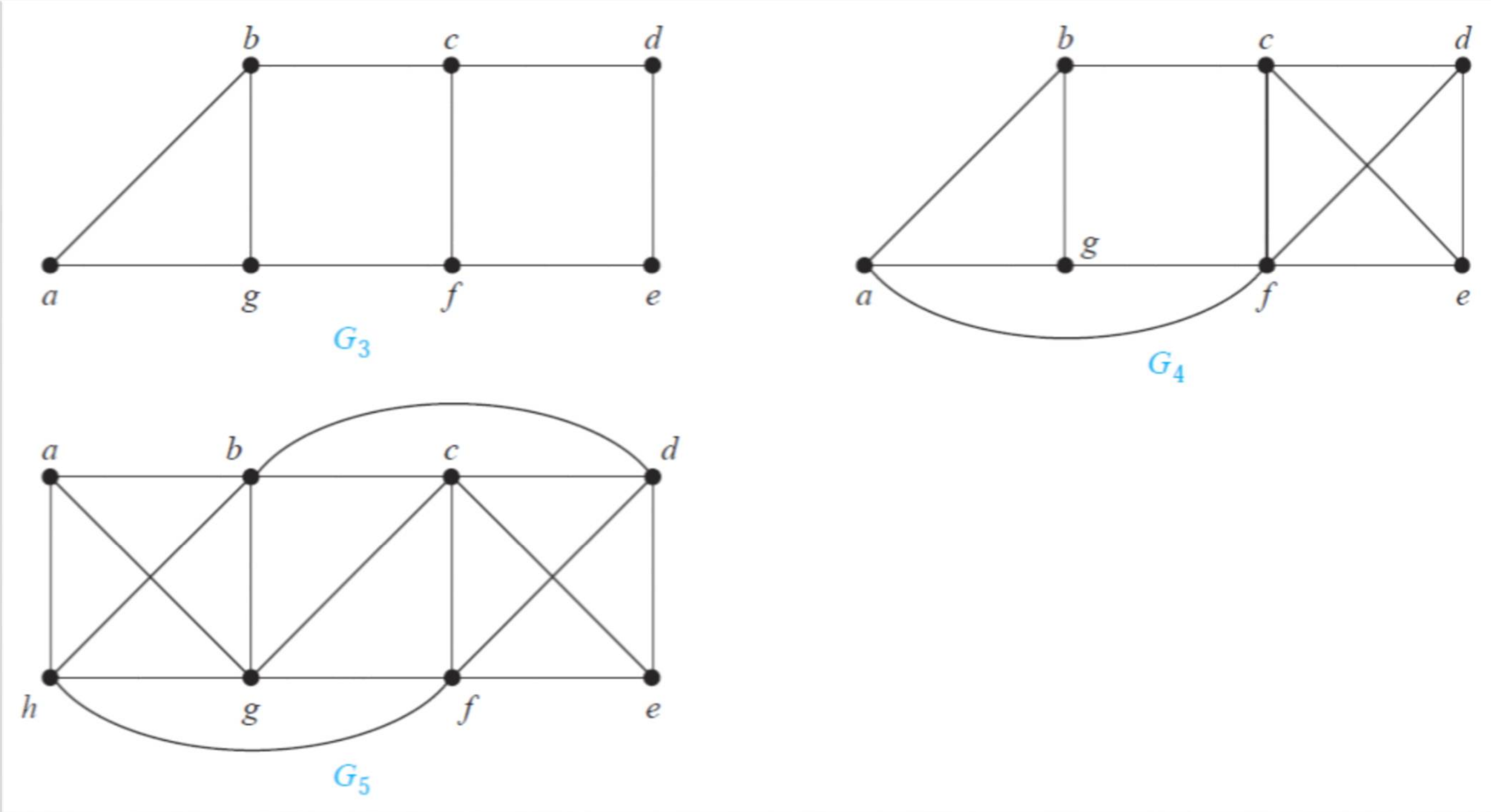


The cut vertices of G_1 are b, c, and e.

The cut edges are $\{a, b\}$ and $\{c, e\}$.

How Connected is a Graph?

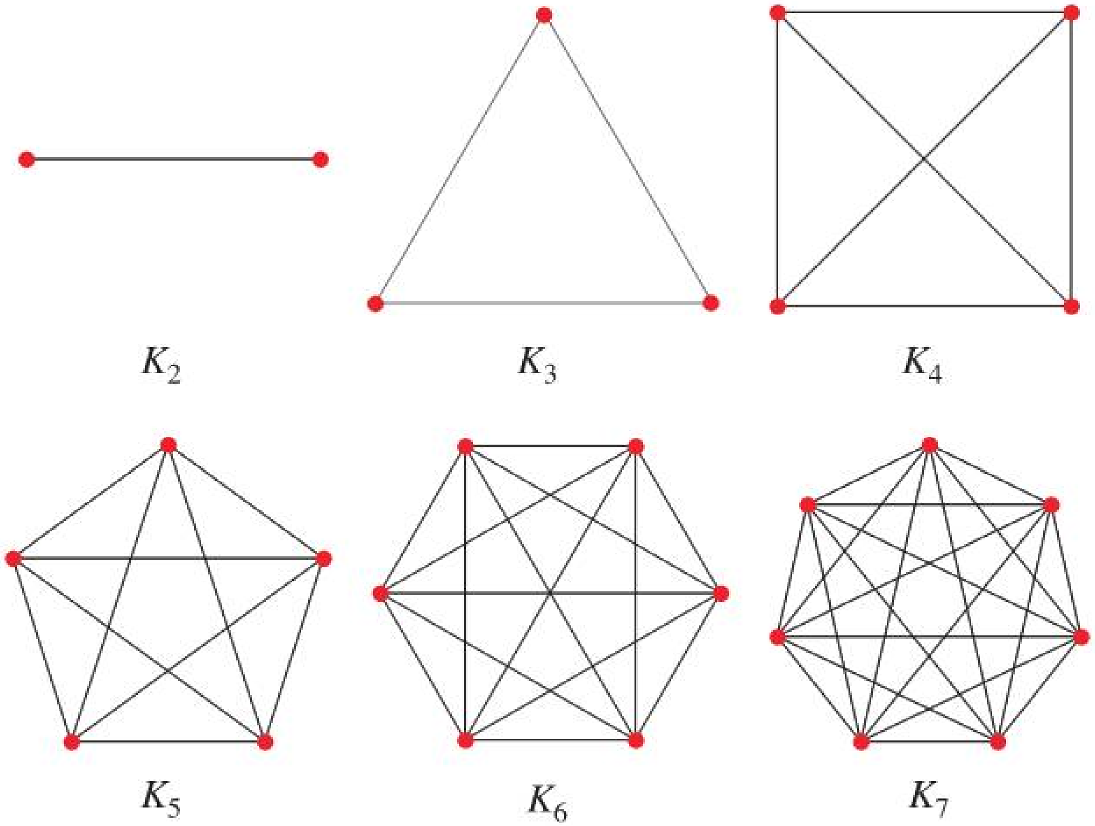
Find the cut vertices and cut edges in the graphs



How Connected is a Graph?

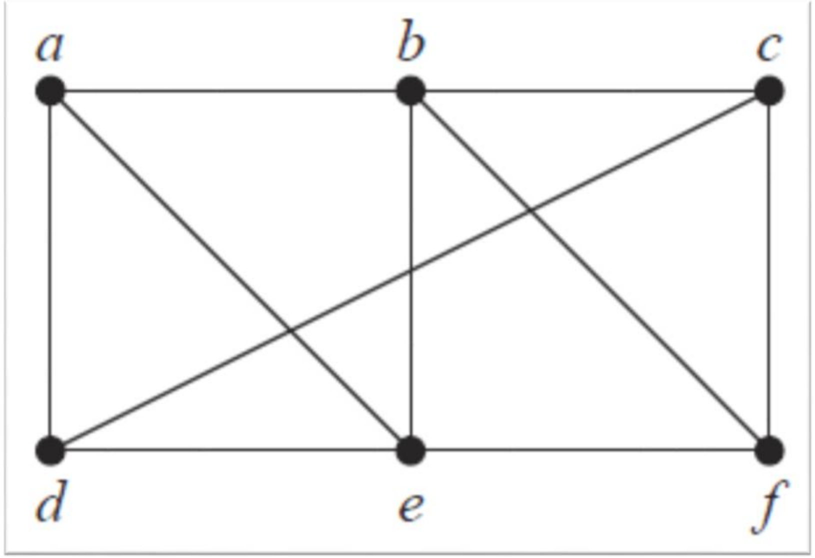
Not all graphs have cut vertices.

For example, the complete graph K_n, where $n \geq 3$, has no cut vertices.



Vertex cut, or separating set

A subset W of the vertex set V of $G = (V, E)$ is a **vertex cut**, or **separating set**, if $G - W$ is disconnected.

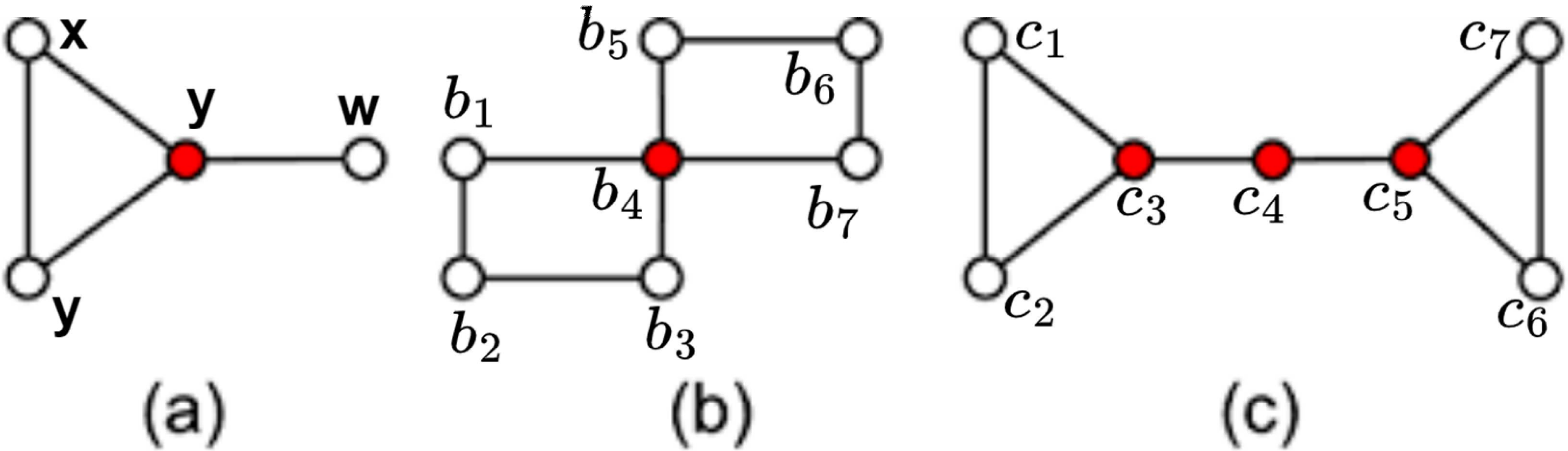


set $\{b, c, e\}$ is a vertex cut with three vertices

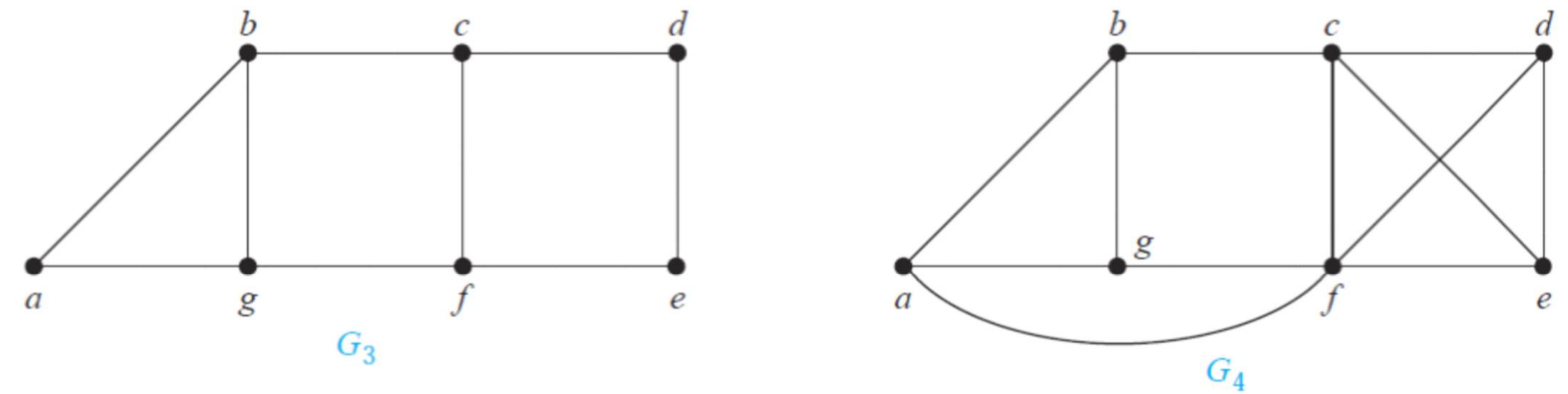
Every connected graph, except a complete graph, has a vertex cut

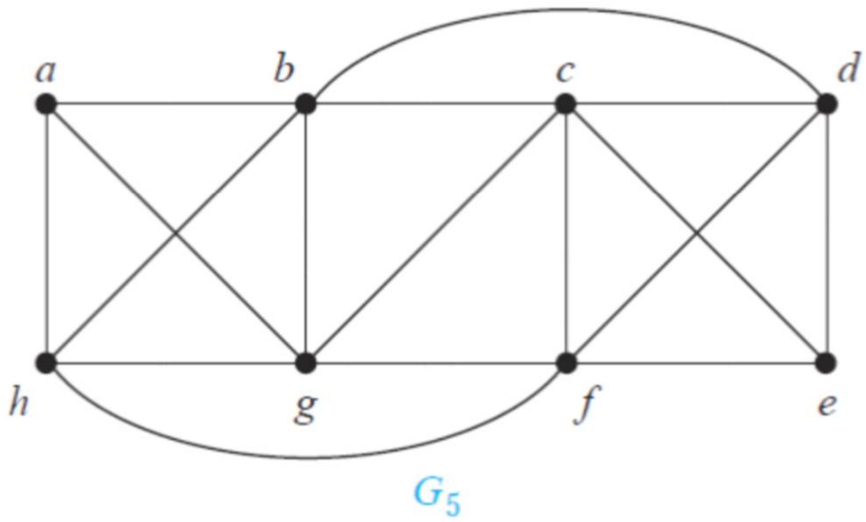
Vertex cut, or **separating set**

Vertex connectivity of a non-complete graph *G*, denoted by *κ(G)*, is the minimum number of vertices in a vertex cut.



Find the Vertex Connectivity *κ(G)*





find the edge connectivity λ(G) of each of the graphs

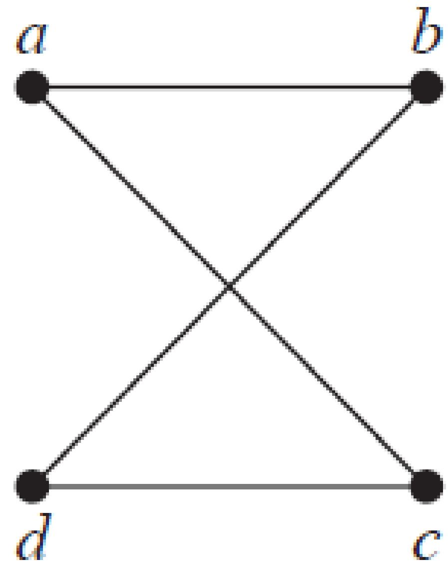
$$\kappa(G) \leq \lambda(G) \leq \min_{v \in V} \deg(v)$$

Counting Paths Between Vertices

$$\mathbf{A}^{r+1} = \mathbf{A}^r\mathbf{A}$$

Let G be a graph with adjacency matrix $\mathbf{A}$ with respect to the ordering $v_1, v_2, \ldots, v_n$ of the vertices of the graph (with directed or undirected edges, with multiple edges and loops allowed). The number of different paths of length r from v_i to v_j, where r is a positive integer, equals the (i, j)th entry of $\mathbf{A}^r$.

How many paths of length four are there from *a* to *d* in the simple graph *G*



The adjacency matrix of *G*

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}$$

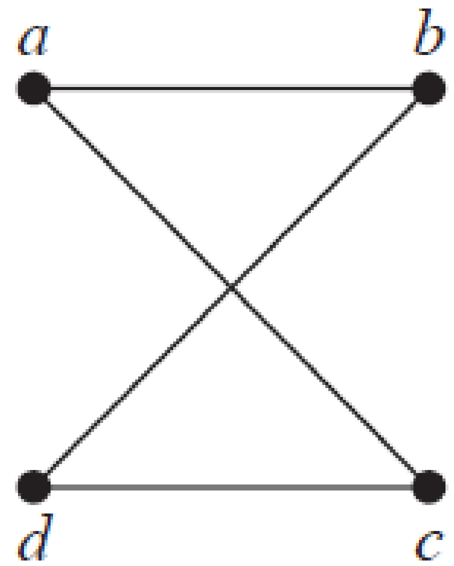
$$\mathbf{A}^4 = \begin{bmatrix} 8 & 0 & 0 & 8 \\ 0 & 8 & 8 & 0 \\ 0 & 8 & 8 & 0 \\ 8 & 0 & 0 & 8 \end{bmatrix}$$

The number of paths of length four from *a* to *d* is the (1, 4)th entry of A^4

Counting Paths Between Vertices

$$\mathbf{A}^{r+1} = \mathbf{A}^r\mathbf{A}$$

How many paths of length four are there from *a* to *d* in the simple graph *G*



The adjacency matrix of *G*

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}$$

$$\mathbf{A}^4 = \begin{bmatrix} 8 & 0 & 0 & 8 \\ 0 & 8 & 8 & 0 \\ 0 & 8 & 8 & 0 \\ 8 & 0 & 0 & 8 \end{bmatrix}$$

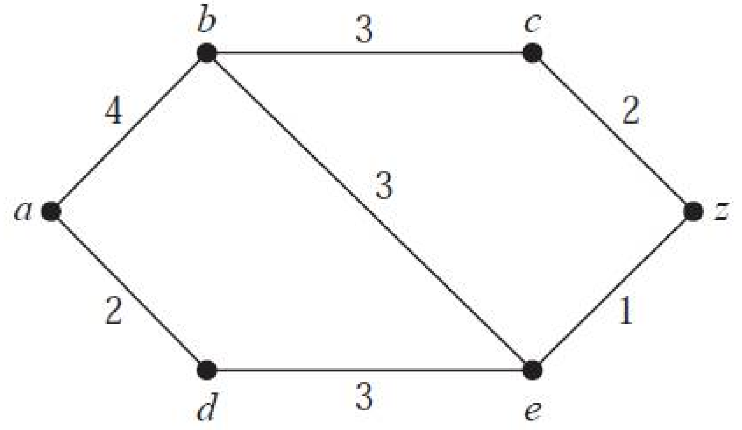
The number of paths of length four from *a* to *d* is the (1, 4)$^{\text{th}}$ entry of A^4

By inspection of the graph, we see that *a, b, a, b, d*; *a, b, a, c, d*; *a, b, d, b, d*; *a, b, d, c, d*; *a, c, a, b, d*; *a, c, a, c, d*; *a, c, d, b, d*; and *a, c, d, c, d* **are the eight paths of length four from *a* to *d*.**

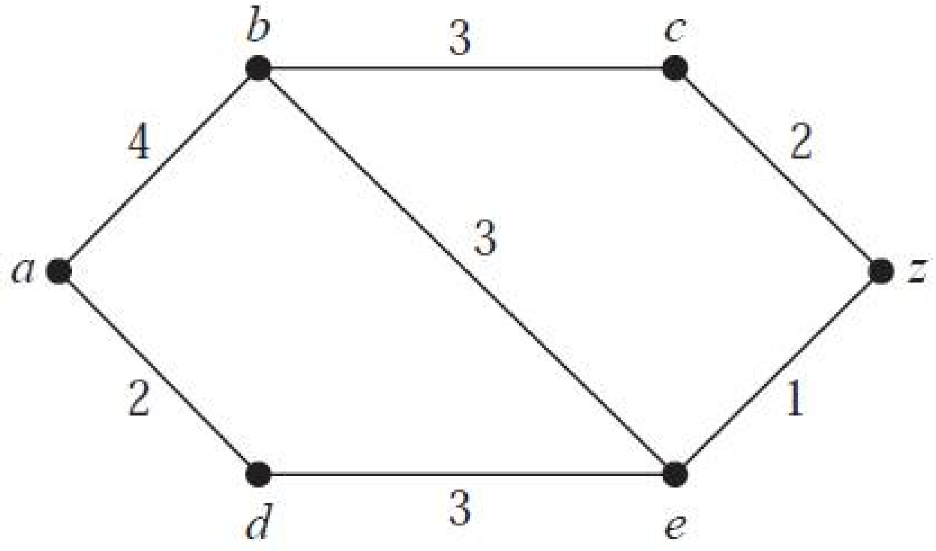
A Shortest-Path Algorithm

Algorithms that find a shortest path between two vertices in a weighted graph. discovered by the Dutch mathematician Edsger **Dijkstra** in 1959.

What is the length of a shortest path between *a* and *z* in the weighted graph



some ideas useful in understanding Dijkstra's algorithm



Distinguised set {a}

We find the **first closest** vertex to **a**

✓ They are a, b of length 4 and

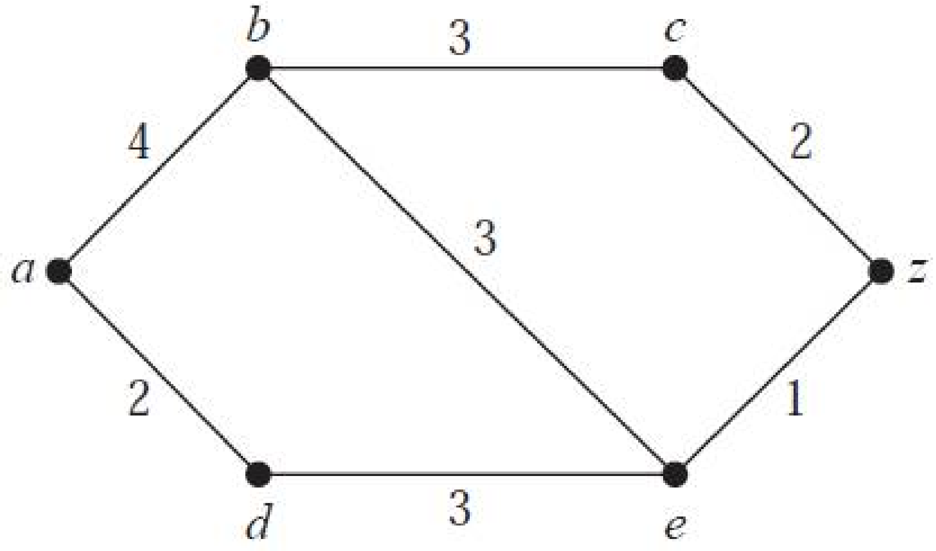
✓ a, d of length 2

We find the **second closest** vertex by examining all paths that begin with the shortest path from **a** to vertex in the set **{a, d}**

Distinguised set {a, d}

There are two such paths to consider, **a, d, e** of length 5 and **a, b** of length 4

some ideas useful in understanding Dijkstra's algorithm



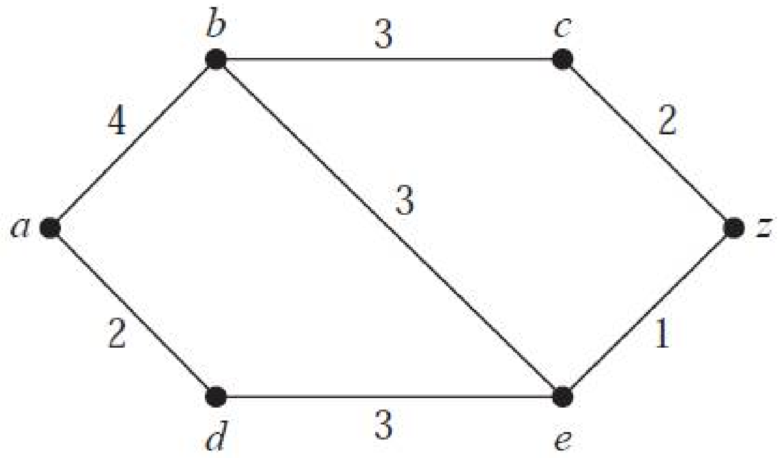
There are two such paths to consider, **a, d, e** of length 5 and **a, b** of length 4

Distinguished set = {a, d, b}

Third closest vertex: There are three such paths, ***a, b, c*** of length 7, ***a, b, e*** of length 7, and ***a, d, e*** of length 5.

Distinguished set = {a, d, b, e}

some ideas useful in understanding Dijkstra's algorithm



Third closest vertex: There are three such paths, ***a, b, c*** of length 7, ***a, b, e*** of length 7, and ***a, d, e*** of length 5.

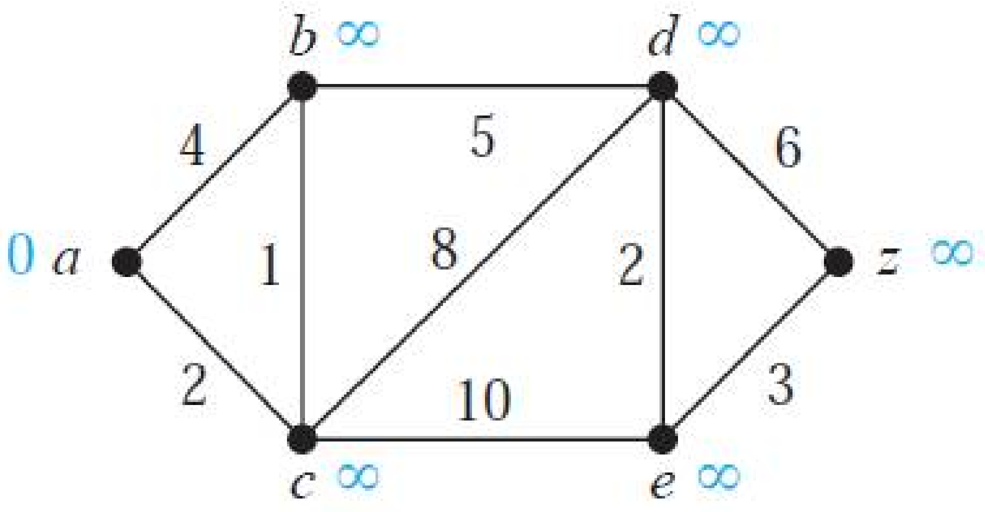
Distinguished set = {a, d, b, e}

forth closest vertex: There are two such paths, **a, b, c** of length 7 and **a, d, e, z** of length 6.

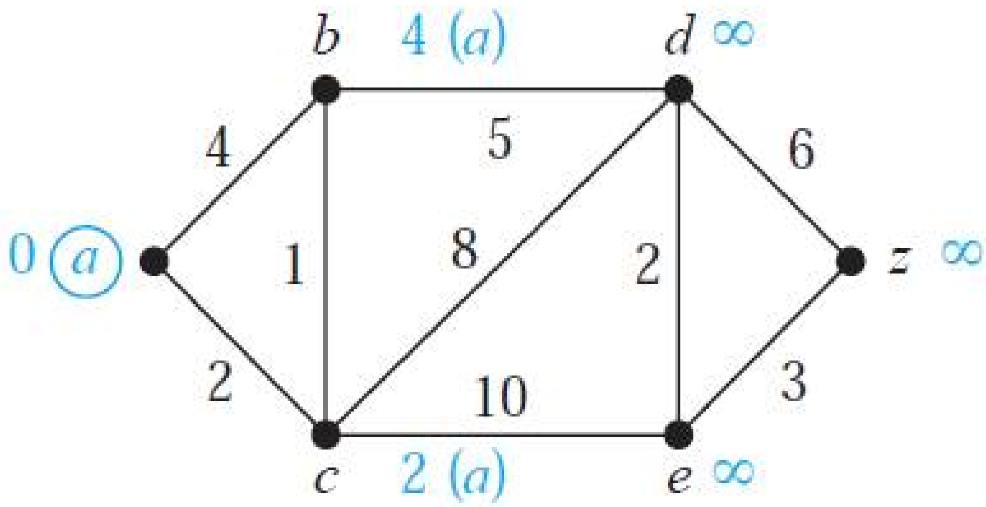
the **fourth closest vertex** to ***a*** **is** ***z*** and the length of the shortest path from *a* to *z* is 6.

Example 2

Use Dijkstra's algorithm to find the length of a shortest path between the vertices **a** and **z**

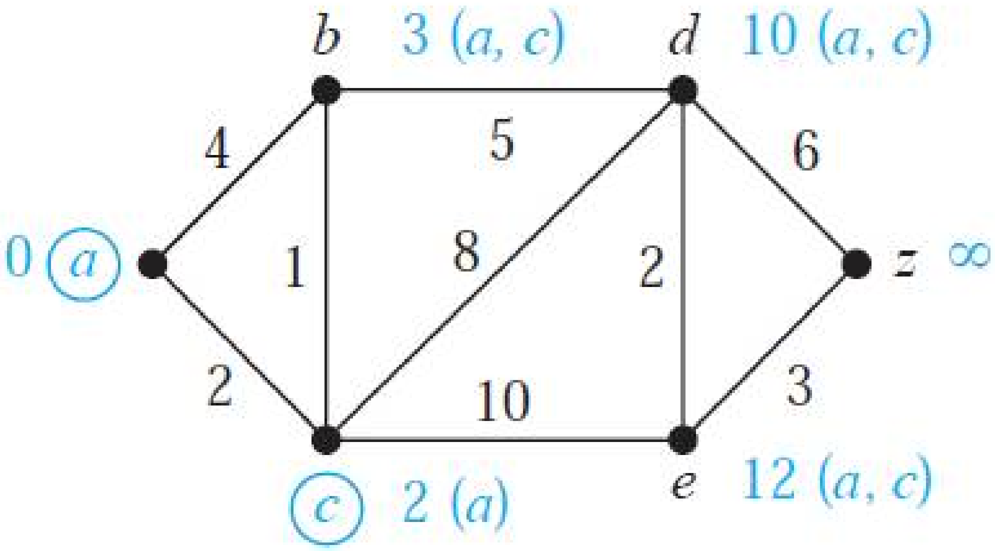


(a)

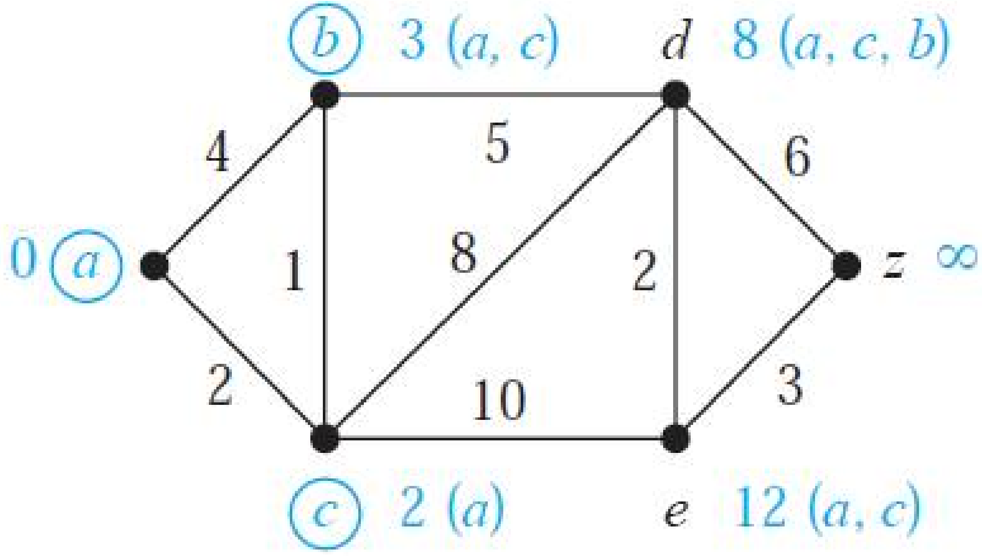


(b)

Example 2

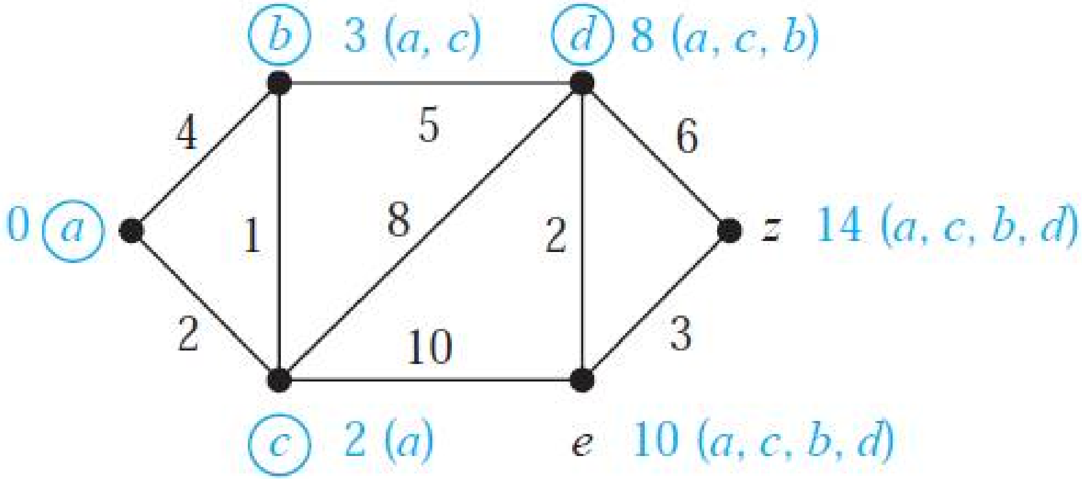

b 3 (a, c)
d 10 (a, c)
0 a
z ∞
c 2 (a)
e 12 (a, c)
4
2
1
5
8
10
2
6
3

(c)


b 3 (a, c)
d 8 (a, c, b)
0 a
z ∞
c 2 (a)
e 12 (a, c)
4
2
1
5
8
10
2
6
3

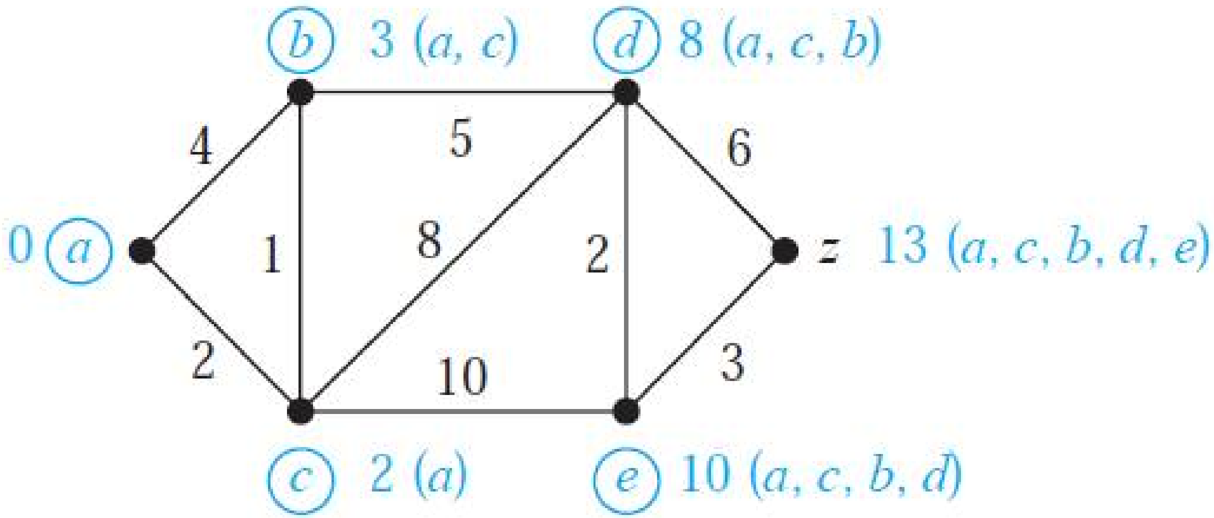
(d)

Example 2



b 3 (a, c)
d 8 (a, c, b)
0 a
z 14 (a, c, b, d)
c 2 (a)
e 10 (a, c, b, d)
4
5
6
1
8
2
2
10
3

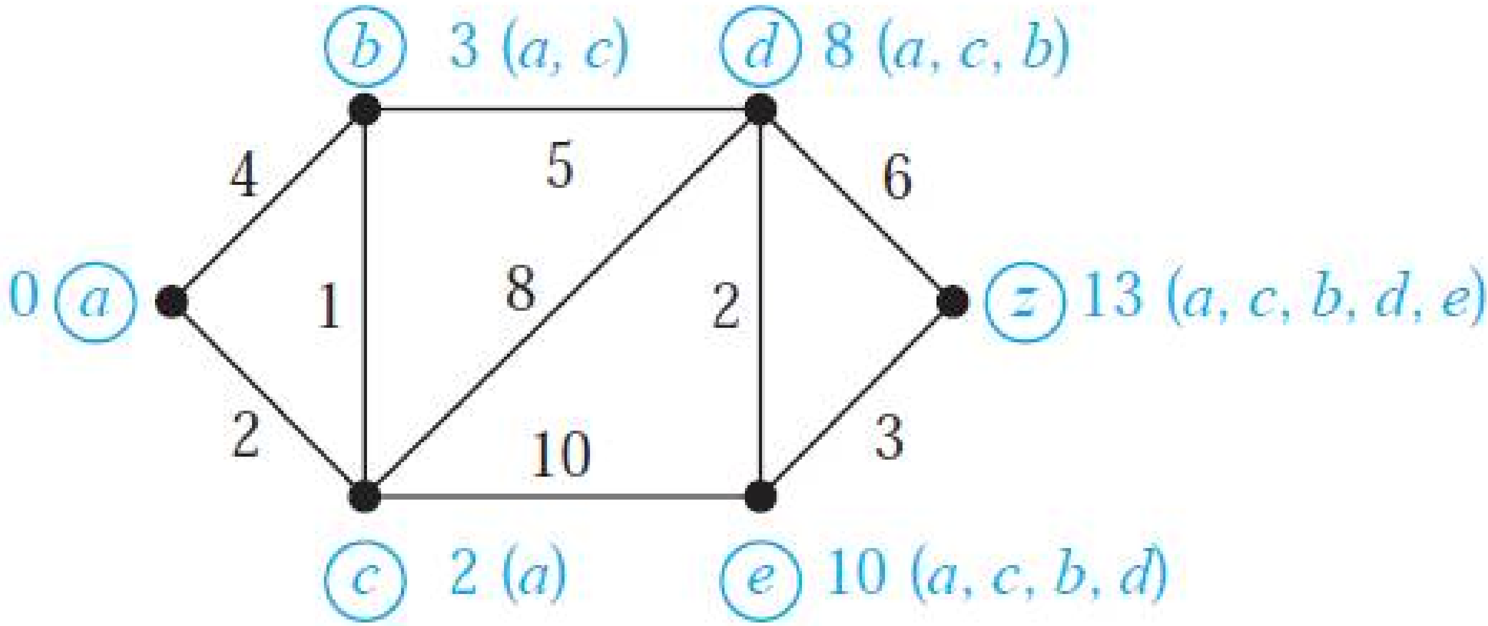
(e)



b 3 (a, c)
d 8 (a, c, b)
0 a
z 13 (a, c, b, d, e)
c 2 (a)
e 10 (a, c, b, d)
4
5
6
1
8
2
2
10
3

(f)

Example 2



(g)

Dijkstra’s algorithm finds the length of a shortest path between two vertices in a connected simple undirected weighted graph.

Example 3: find the length of a shortest path between ***a*** and ***z*** in the given weighted graph