

GRAPH THEORY

PRESENTED BY:-

ER. HANIT KARWAL

ASSISTANT PROFESSOR

INFORMATION TECHNOLOGY DEPT.

GNDEC, LUDHIANA

hanitgndec@gmail.com

SYLLABUS

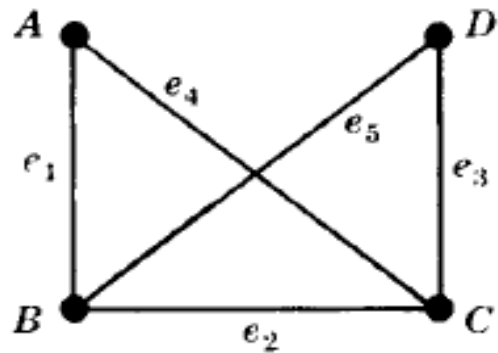
- Graph terminology
- Directed and undirected graphs
- Eulerian chains and cycles
- Hamiltonian chains and cycles
- Shortest path algorithms – Dijkstra's algorithm, Warshall's algorithm
- Graph coloring & Chromatic number
- Planar graphs & Euler's Theorem for Planar Graphs
- Isomorphic and homomorphic graphs
- Applications of graph theory.



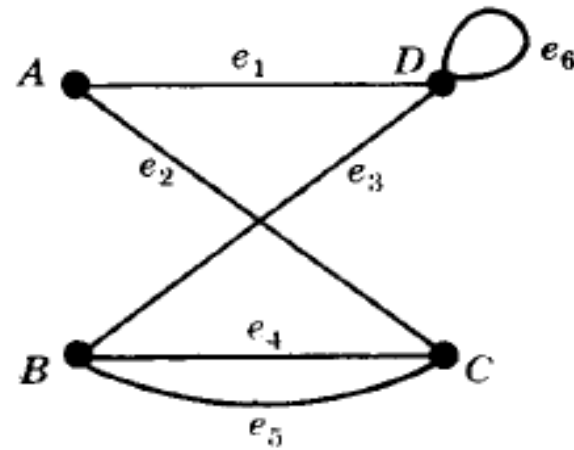
GRAPHS AND MULTIGRAPHS

- *A graph G consists of two things:*
 - (i) *A set $V = V(G)$ whose elements are called vertices, points, or nodes of G .*
 - (ii) *A set $E = E(G)$ of unordered pairs of distinct vertices called edges of G .*
- *We denote such a graph by $G(V, E)$ when we want to emphasize the two parts of G .*
 - (i) *V consists of vertices A, B, C, D .*
 - (ii) *E consists of edges $e1 = \{A, B\}$, $e2 = \{B, C\}$, $e3 = \{C, D\}$, $e4 = \{A, C\}$, $e5 = \{B, D\}$.*





(a) Graph



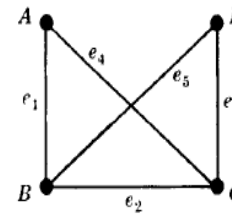
(b) Multigraph

The edges e_4 and e_5 are called *multiple edges* since they connect the same endpoints, and the edge e_6 is called a *loop* since its endpoints are the same vertex. Such a diagram is called a *multigraph*.

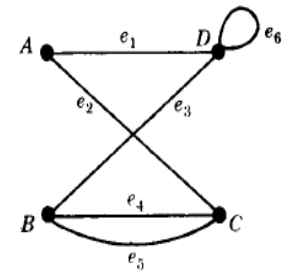
The formal definition of a graph permits neither multiple edges nor loops. Thus a graph may be defined to be a multigraph without multiple edges or loops.



DEGREE OF A VERTEX



(a) Graph



(b) Multigraph

- The *degree of a vertex v in a graph G , written $\deg(v)$, is equal to the number of edges in G which contain v , that is, which are incident on v . Since each edge is counted twice in counting the degrees of the vertices of G*
- We have $\deg(A) = 2$, $\deg(B) = 3$, $\deg(C) = 3$, $\deg(D) = 2$.
- The sum of the degrees equals 10 which, as expected, is twice the number of edges. A vertex is said to be *even or odd according as its degree is an even or an odd number*. Thus A and D are even vertices whereas B and C are odd vertices.
- For multigraphs where a loop is counted twice toward the degree of its endpoint. $\deg(D) = 4$ since the edge e_6 is counted twice; hence D is an even vertex.
- A vertex of degree zero is called an *isolated vertex and is known as trivial graph*.

TERMINOLOGIES:

1. The two vertices u and v are *end vertices of the edge (u, v)* .
2. Edges that have the same end vertices are *parallel*.
3. An edge of the form (v, v) is a *loop*.
4. A graph is *simple if it has no parallel edges or loops*.
5. A graph with no edges (i.e. E is empty) is *empty*.
6. A graph with no vertices (i.e. V and E are empty) is a *null graph*.
7. A graph with only one vertex is *trivial*.



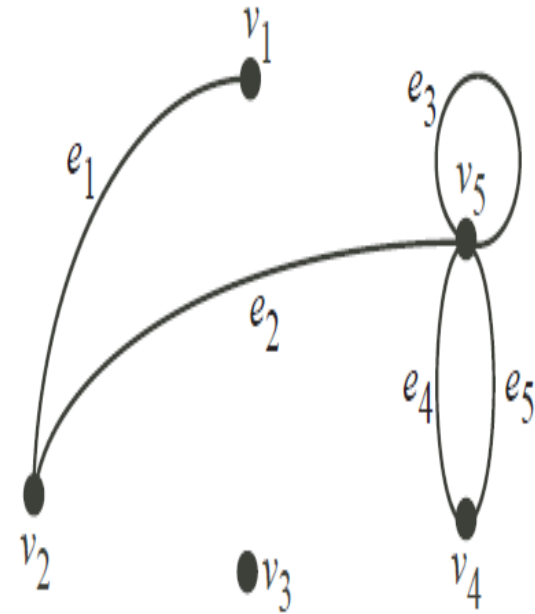
TERMINOLOGIES:

8. Edges are *adjacent* if they share a common end vertex.
9. Two vertices u and v are *adjacent* if they are connected by an edge, in other words, (u, v) is an edge.
10. The *degree* of the vertex v , written as $d(v)$, is the number of edges with v as an end vertex.
By convention, we count a loop twice and parallel edges contribute separately.
11. A *pendant vertex* is a vertex whose degree is 1.
12. An edge that has a pendant vertex as an end vertex is a *pendant edge*.
13. An *isolated vertex* is a vertex whose degree is 0.



EXAMPLE

- v_4 and v_5 are end vertices of e_5 .
- e_4 and e_5 are parallel.
- e_3 is a loop.
- The graph is not simple.
- e_1 and e_2 are adjacent.
- v_1 and v_2 are adjacent.
- The degree of v_1 is 1 so it is a pendant vertex.
- e_1 is a pendant edge.
- The degree of v_5 is 5.
- The degree of v_4 is 2.
- The degree of v_3 is 0 so it is an isolated vertex

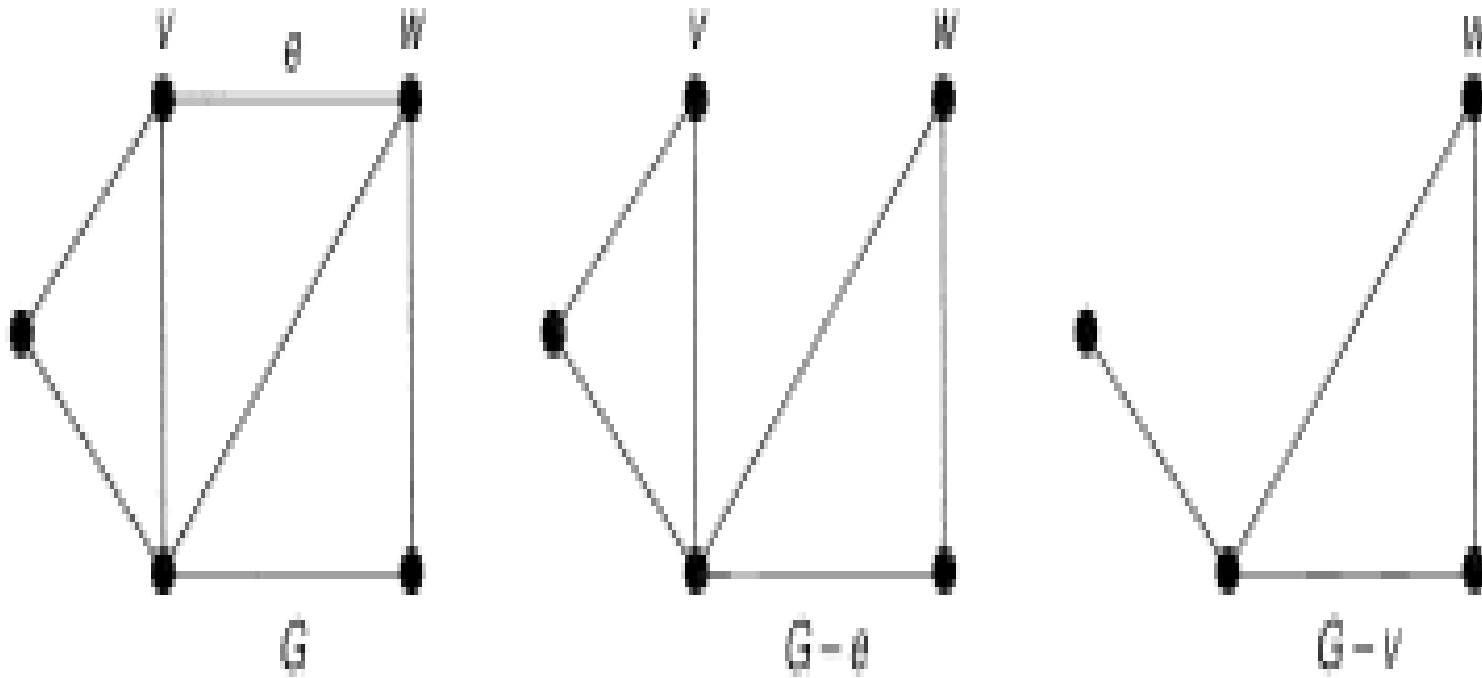


SUBGRAPHS

- A graph $G = G(V, E)$. A graph $H = H(V, E)$ is called a *subgraph of G* if the vertices and edges of H are contained in the vertices and edges of G , that is, if $V \subseteq V$ and $E \subseteq E$
- *Properties*
 - (i) A subgraph $H(V, E)$ of $G(V, E)$ is called the *subgraph induced by its vertices V* if its edge set E contains all edges in G whose endpoints belong to vertices in H .
 - (ii) If v is a vertex in G , then $G - v$ is the subgraph of G obtained by deleting v from G and deleting all edges in G which contain v .
 - (iii) If e is an edge in G , then $G - e$ is the subgraph of G obtained by simply deleting the edge e from G .

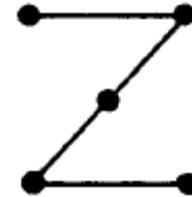
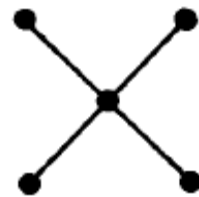
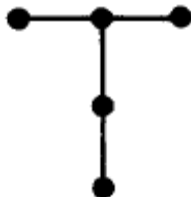
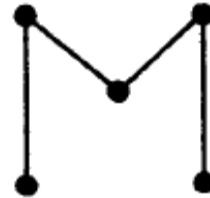
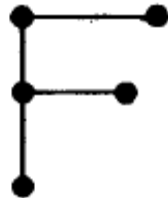
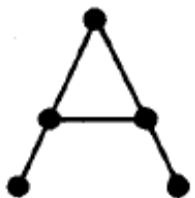


SUBGRAPHS



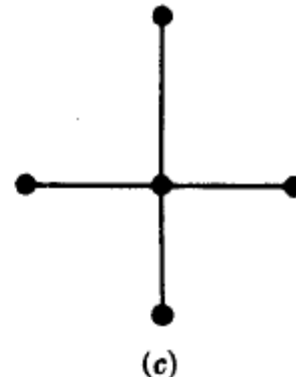
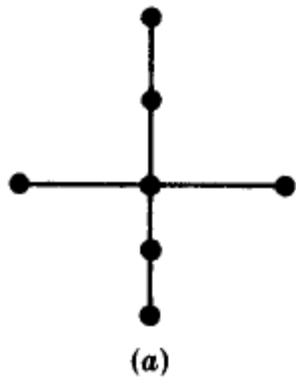
ISOMORPHIC GRAPHS

- Graphs $G(V,E)$ and $G(V^*,E^*)$ are said to be isomorphic if there exists a one-to-one correspondence
- $f: V \rightarrow V^*$ such that $\{u, v\}$ is an edge of G if and only if $\{f(u), f(v)\}$ is an edge of G^* .
- A and R are isomorphic graphs
- F and T are isomorphic graphs
- K and X are isomorphic graphs
- M, S, V , and Z are isomorphic graphs.



HOMEOMORPHIC GRAPHS

- Two graphs G and G^* are said to be homeomorphic if they can be obtained from the same graph or isomorphic graphs by this method.
- The graphs (a) and (b) in Fig. are not isomorphic, but they are homeomorphic since they can be obtained from the graph (c) by adding appropriate vertices.



PATHS, CONNECTIVITY

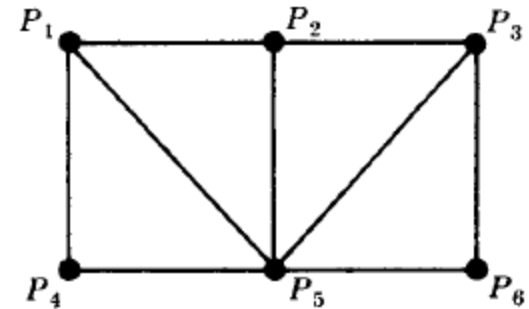
- *A path in a multigraph G consists of an alternating sequence of vertices and edges of the form $v_0, e_1, v_1, e_2, v_2, \dots, e_{n-1}, v_{n-1}, e_n, v_n$ where each edge e_i contains the vertices v_{i-1} and v_i (which appear on the sides of e_i in the sequence).*
- *The number n of edges is called the length of the path. When there is no ambiguity, we denote a path by its sequence of vertices (v_0, v_1, \dots, v_n) .*
- *The path is said to be closed if $v_0 = v_n$.*
- *Otherwise, we say the path is from v_0 , to v_n or between v_0 and v_n , or connects v_0 to v_n .*



- *A simple path is a path in which all vertices are distinct.*
- *A path in which all edges are distinct will be called a trail.*
- *A cycle is a closed path of length 3 or more in which all vertices are distinct except $v_0 = v_n$.*
- *A cycle of length k is called a k -cycle.*



- Consider the following sequences:
 $\alpha = (P_4, P_1, P_2, P_5, P_1, P_2, P_3, P_6)$,
 $\beta = (P_4, P_1, P_5, P_2, P_6)$,
 $\gamma = (P_4, P_1, P_5, P_2, P_3, P_5, P_6)$,
 $\delta = (P_4, P_1, P_5, P_3, P_6)$.

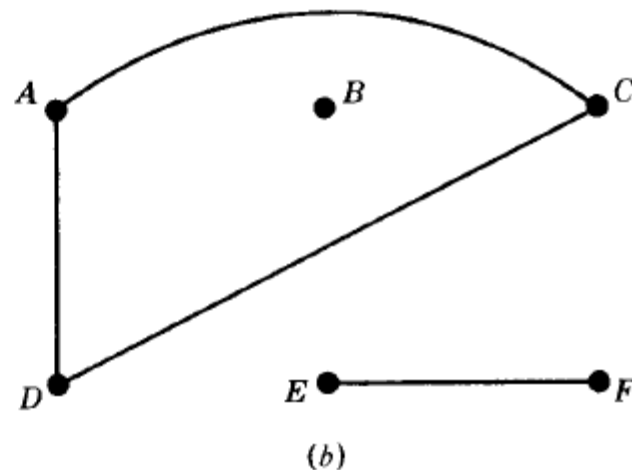
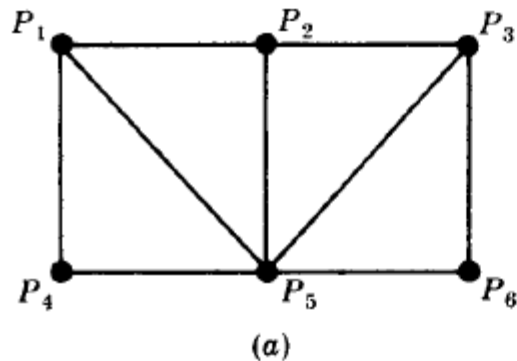


- The sequence α is a path from P_4 to P_6 ; but it is not a trail since the edge $\{P_1, P_2\}$ is used twice.
- The sequence β is not a path since there is no edge $\{P_2, P_6\}$.
- The sequence γ is a trail since no edge is used twice; but it is not a simple path since the vertex P_5 is used twice.
- The sequence δ is a simple path from P_4 to P_6 ; but it is not the shortest path (with respect to length) from P_4 to P_6 . The shortest path from P_4 to P_6 is the simple path (P_4, P_5, P_6) which has length 2.

There is a path from a vertex u to a vertex v if and only if there exists a simple path from u to v .

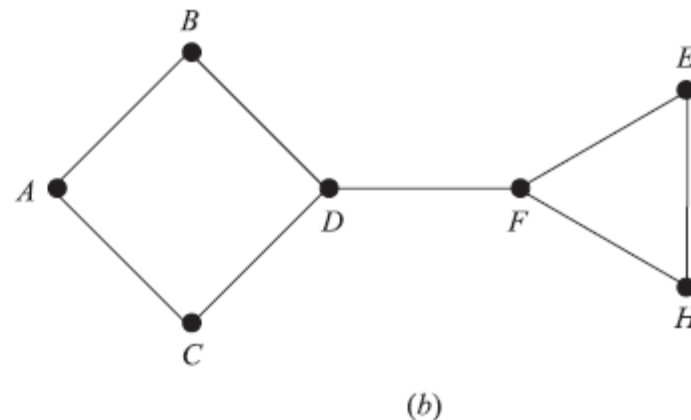
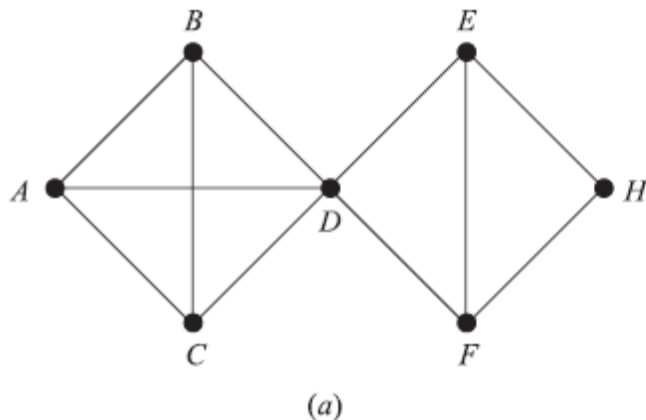


- A graph G is connected if there is a path between any two of its vertices. The graph in Fig.(a) is connected, but the graph in Fig.(b) is not connected since, for example, there is no path between vertices D and E .
- Suppose G is a graph. A connected subgraph H of G is called a connected component of G if H is not contained in any larger connected subgraph of G .
- It is intuitively clear that any graph G can be partitioned into its connected components. For example, the graph G in Fig.(b) has three connected components, the subgraphs induced by the vertex sets $\{A, C, D\}$, $\{E, F\}$, and $\{B\}$.
- The vertex B in Fig.(b) is called an isolated vertex since B does not belong to any edge or, in other words, $\deg(B) = 0$. Therefore, as noted, B itself forms a connected component of the graph.



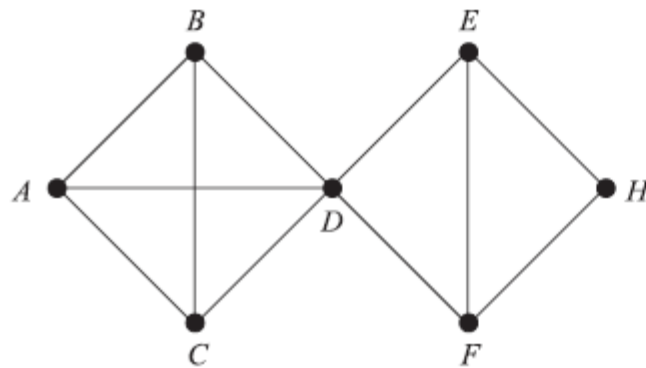
DISTANCE AND DISTANCE

- The *distance between vertices u and v in G* , written $d(u, v)$, is the length of the shortest path between u and v .
- The *diameter of G* , written $\text{diam}(G)$, is the maximum distance between any two points in G .
- For example, in Fig. (a), $d(A, F) = 2$ and $\text{diam}(G) = 3$,
- In Fig. (b), $d(A, F) = 3$ and $\text{diam}(G) = 4$.

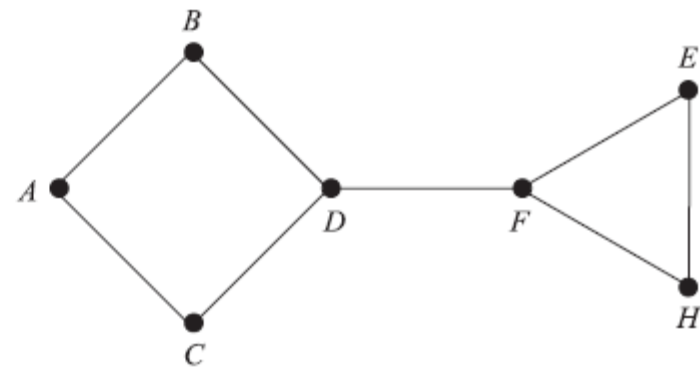


CUTPOINTS AND BRIDGES

- Let G be a connected graph.
- A vertex v in G is called a **cutpoint** if $G-v$ is disconnected. ($G-v$ is the graph obtained from G by deleting v and all edges containing v .)
- An edge e of G is called a **bridge** if $G-e$ is disconnected. ($G-e$ is the graph obtained from G by simply deleting the edge e).
- In Fig. (a), the vertex D is a cutpoint and there are no bridges.
- In Fig. (b), the edge $= \{D, F\}$ is a bridge. (Its endpoints D and F are necessarily cutpoints.)



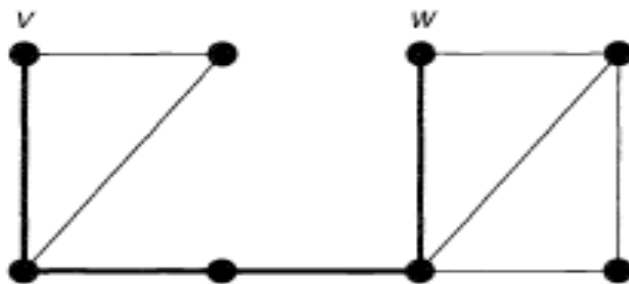
(a)



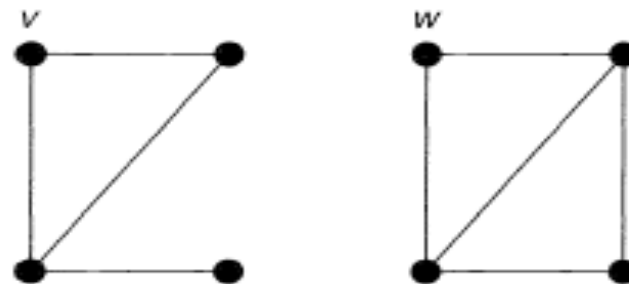
(b)

CONNECTIVITY

- A graph is connected if and only if there is a path between each pair of vertices.
- *If G is a bipartite graph, then each cycle of G has even length.*
- The **girth** of a graph is the length of its shortest cycle



connected



disconnected



EULERIAN GRAPHS

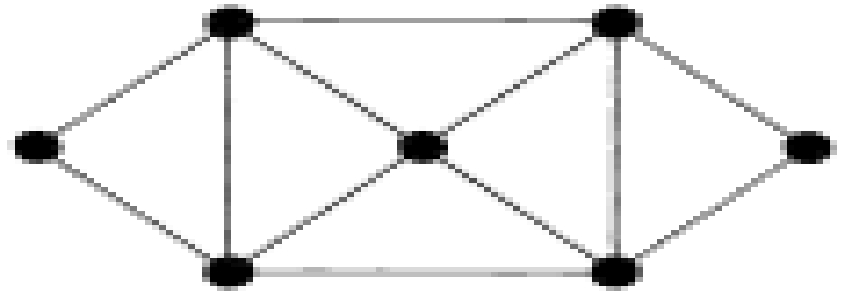


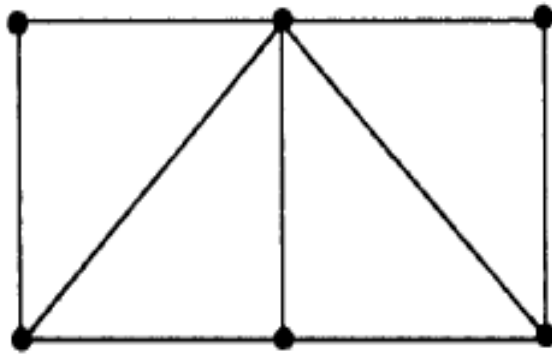
Fig. 6.1

- A connected graph G is Eulerian if there exists a closed trail containing every edge of G .
- Such a trail is an Eulerian trail.
- Each edge is traversed once and once only but may repeat the vertices.
- *If G is a graph in which the degree of each vertex is at least 2, then G contains a cycle.*
- *A connected graph G is Eulerian if and only if the degree of each vertex of G is even.*

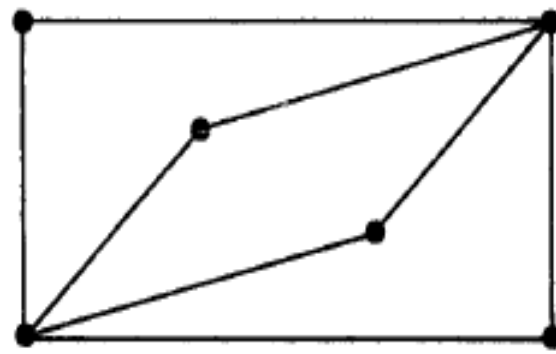


HAMILTONIAN GRAPHS

- A Hamiltonian circuit visits each vertex exactly once but may repeat edges.
- Let G be a connected graph with n vertices. Then G is Hamiltonian if $n \geq 3$ and $n \leq \deg(v)$ for each vertex v in G .



(a) Hamiltonian and non-Eulerian



(b) Eulerian and non-Hamiltonian



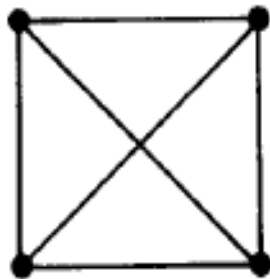
COMPLETE GRAPH

- A graph G is said to be complete if every vertex in G is connected to every other vertex in G . Thus a complete graph G must be connected.
- The complete graph with n vertices is denoted by K_n .

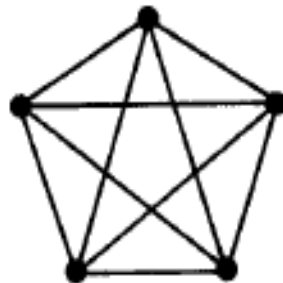
K_1 = isolated vertex: ●

K_2 = line segment: ●—●

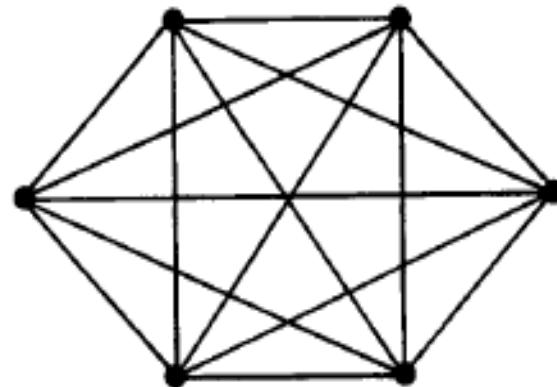
K_3 = triangle: 



K_4



K_5



K_6



REGULAR GRAPH

- A graph G is *regular of degree k or k -regular* if every vertex has degree k .
- In other words, a graph is regular if every vertex has the same degree.
- The connected 0-regular graph is the trivial graph with one vertex and no edges.
- The connected 1-regular graph is the graph with two vertices and one edge connecting them.
- The connected 2-regular graph with n vertices is the graph which consists of a single n -cycle.



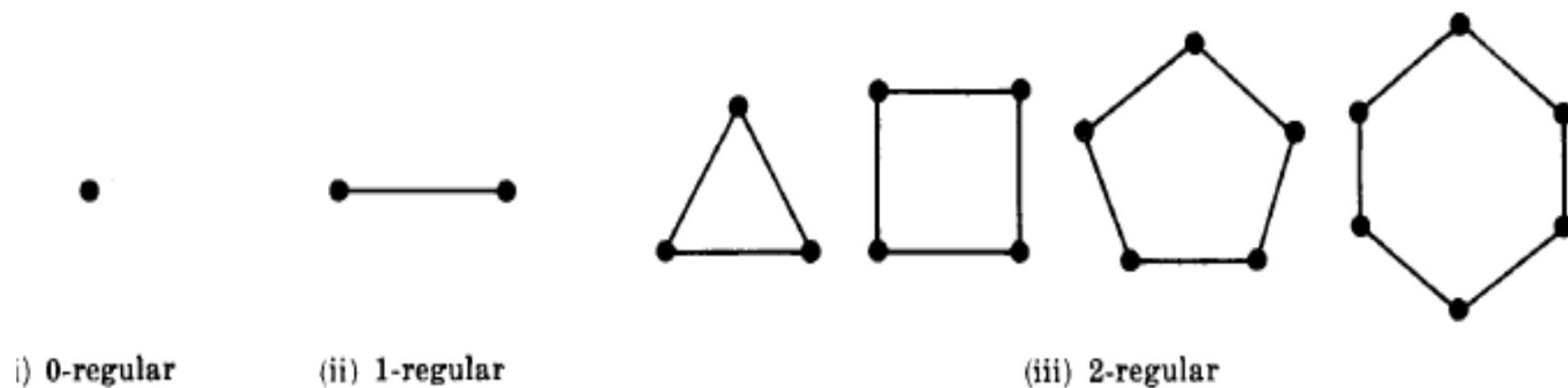
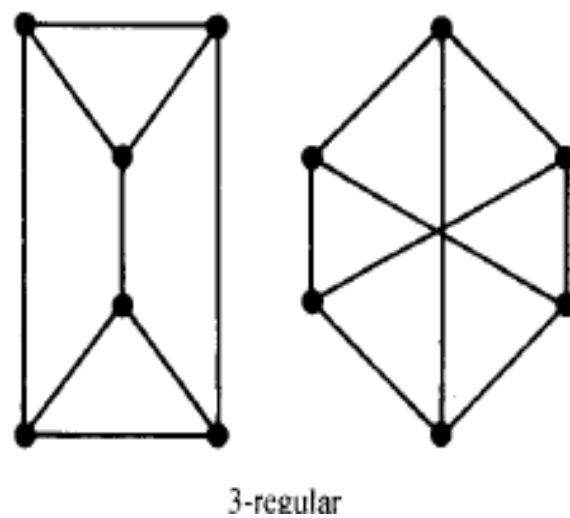


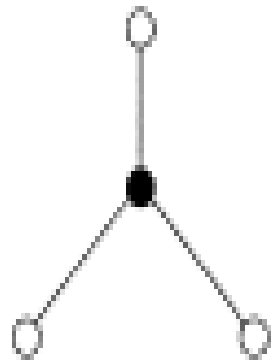
Fig. 8-14



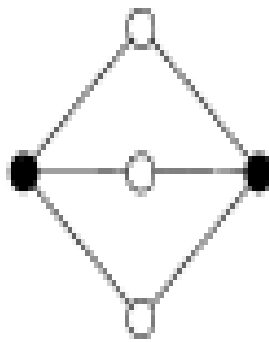
BIPARTITE GRAPHS

- A graph G is said to be bipartite if its vertices V can be partitioned into two subsets M and N such that each edge of G connects a vertex of M to a vertex of N .
- By a complete bipartite graph, we mean that each vertex of M is connected to each vertex of N ; this graph is denoted by $K_{m,n}$ where m is the number of vertices in M and n is the number of vertices in N , and, for standardization, we will assume $m \leq n$.

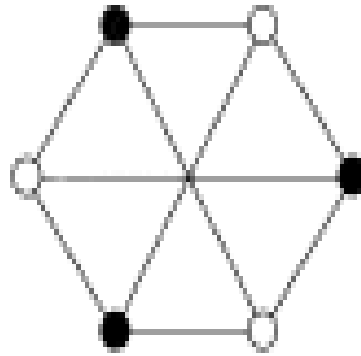




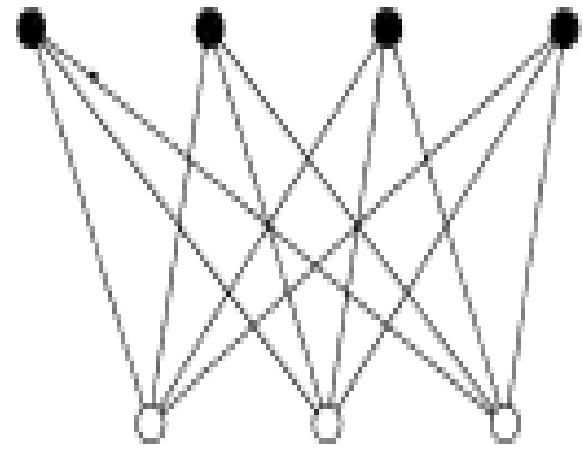
$K_{1,3}$



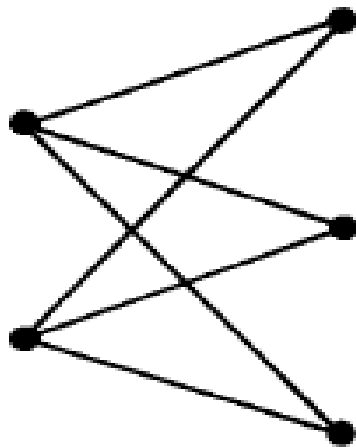
$K_{2,2}$



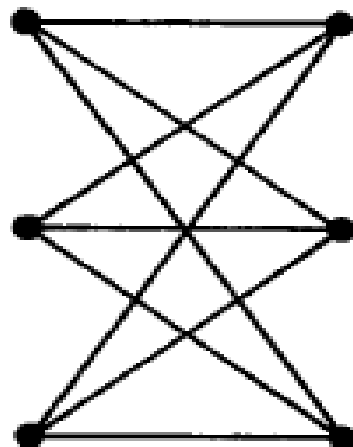
$K_{3,3}$



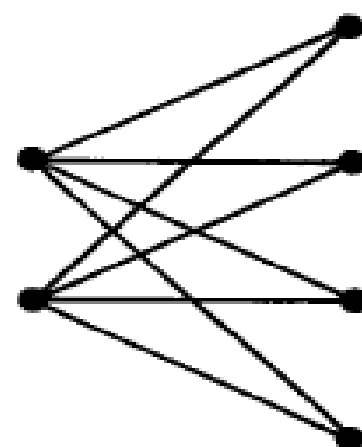
$K_{4,3}$



$K_{2,3}$



$K_{3,3}$



$K_{2,4}$

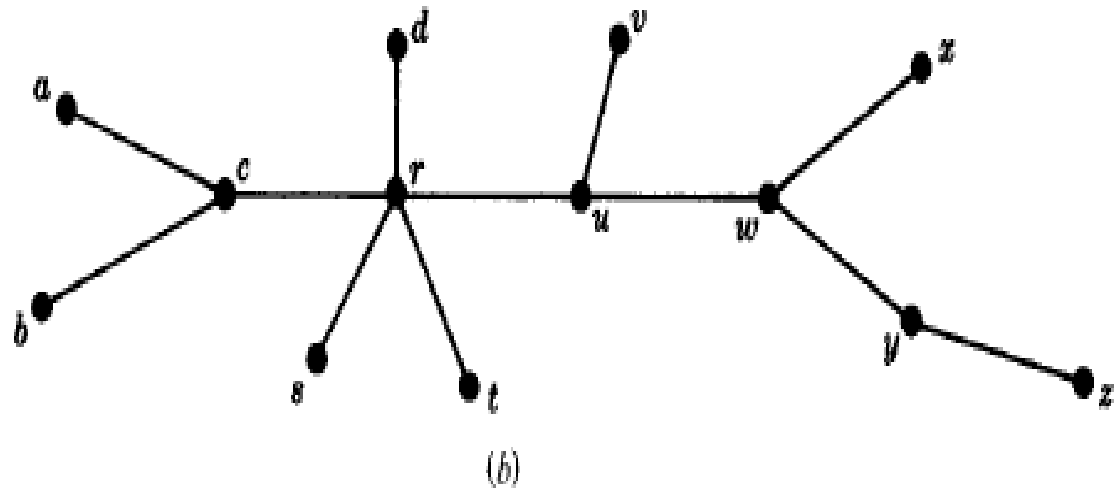
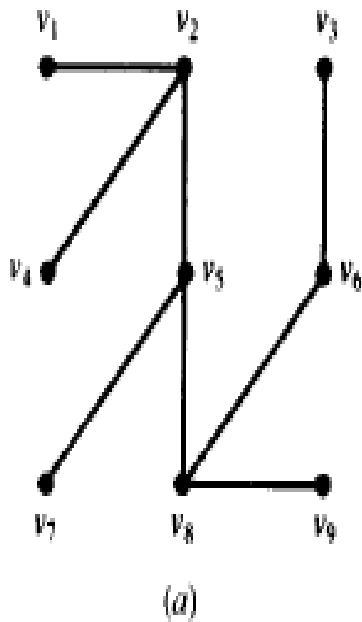


TREE GRAPHS

- A graph T is called a tree if T is connected and T has no cycles.
- A graph without cycles is said to be cycle-free
- The tree consisting of a single vertex with no edges is called the degenerate tree.
- Let G be a graph with $n > 1$ vertices. Then the following are equivalent:
 - (i) G is a tree.
 - (ii) G is a cycle-free and has $n - 1$ edges.
 - (iii) G is connected and has $n - 1$ edges.
- This theorem also tells us that a finite tree T with n vertices must have $n-1$ edges.

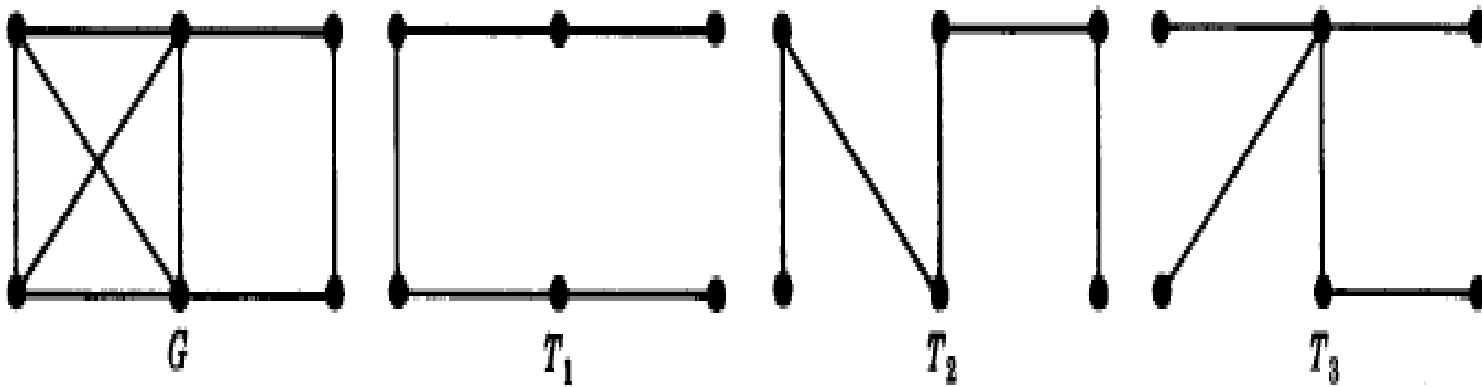


- The tree in Fig.(a) has 9 vertices and 8 edges, and the tree in Fig (b) has 13 vertices and 12 edges.



SPANNING TREES

- A subgraph T of a connected graph G is called a *spanning tree* of G if T is a tree and T includes all the vertices of G . Figure shows a connected graph G and spanning trees T_1 , T_2 , and T_3 of G .



MINIMUM SPANNING TREES

- Suppose G is a connected weighted graph. That is, each edge of G is assigned a nonnegative number called the weight of the edge.
- Then any spanning tree T of G is assigned a total weight obtained by adding the weights of the edges in T .
- A minimal spanning tree of G is a spanning tree whose total weight is as small as possible.



Algorithm 8.2: The input is a connected weighted graph G with n vertices.

Step 1. Arrange the edges of G in the order of decreasing weights.

Step 2. Proceeding sequentially, delete each edge that does not disconnect the graph until $n - 1$ edges remain.

Step 3. Exit.

Algorithm 8.3 (Kruskal): The input is a connected weighted graph G with n vertices.

Step 1. Arrange the edges of G in order of increasing weights.

Step 2. Starting only with the vertices of G and proceeding sequentially, add each edge which does not result in a cycle until $n - 1$ edges are added.

Step 3. Exit.

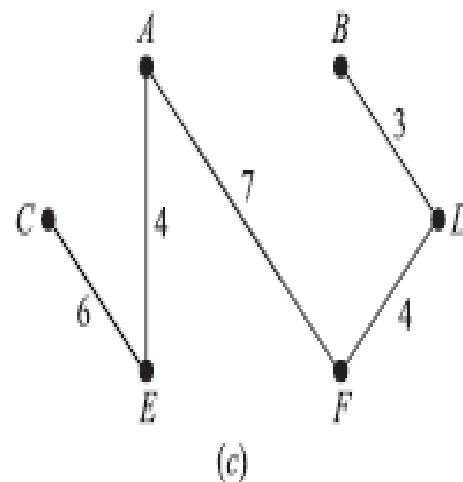
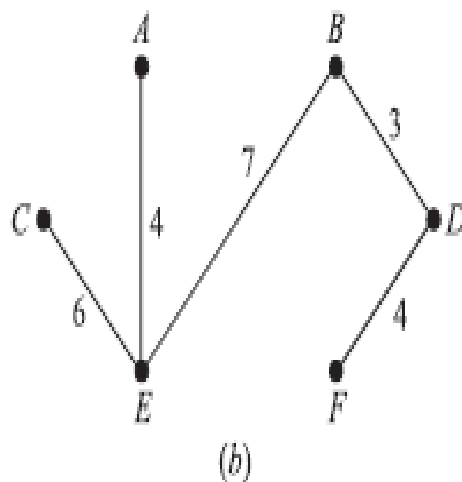
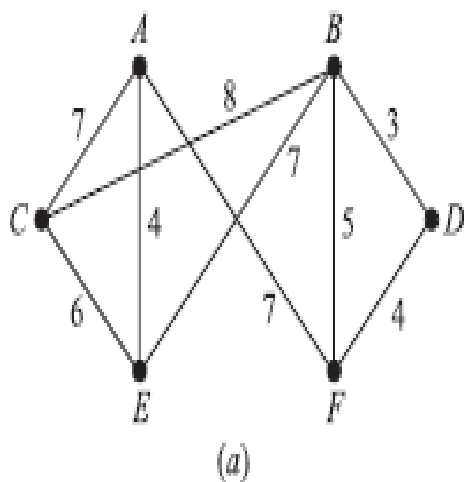


- First we order the edges by decreasing weights, and then we successively delete edges without disconnecting Q until five edges remain. This yields the following data

Edges	BC	AF	AC	BE	CE	BF	AE	DF	BD
Weight	8	7	7	7	6	5	4	4	3
Delete	Yes	Yes	Yes	No	No	Yes			

Thus the minimal spanning tree of Q which is obtained contains the edges BE, CE, AE, DF, BD

The spanning tree has weight 24

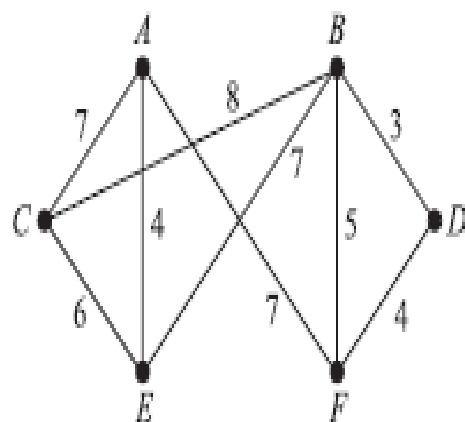


First we order the edges by increasing weights, and then we successively add edges without forming any cycles until five edges are included. This yields the following data:

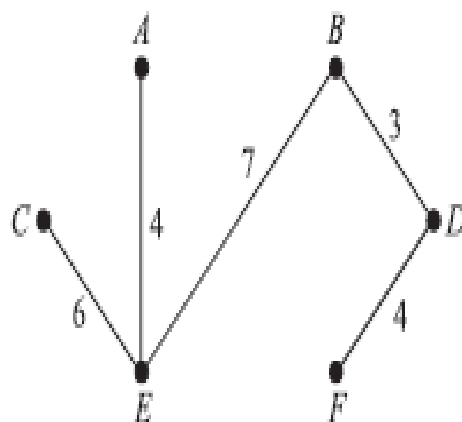
Edges	BD	AE	DF	BF	CE	AC	AF	BE	BC
Weight	3	4	4	5	6	7	7	7	8
Add?	Yes	Yes	Yes	No	Yes	No	Yes		

Thus the minimal spanning tree of Q which is obtained contains the edges

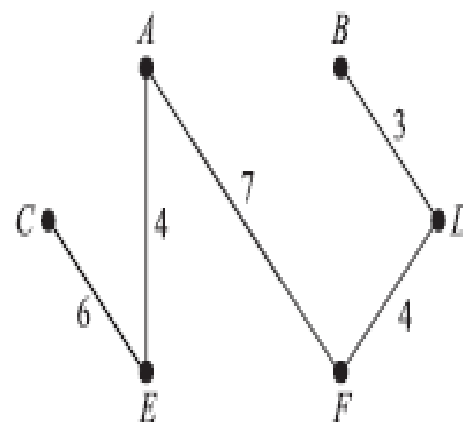
BD, AE, DF, CE, AF



(a)



(b)



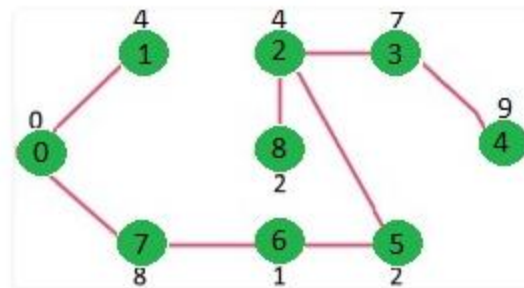
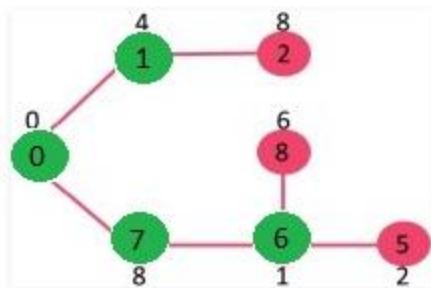
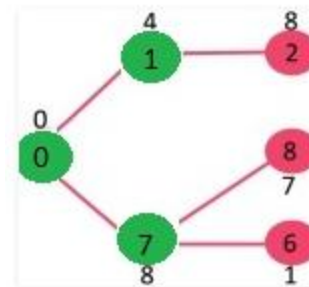
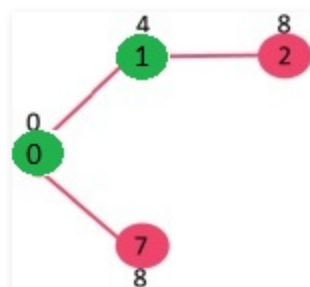
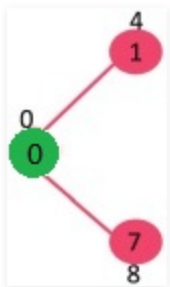
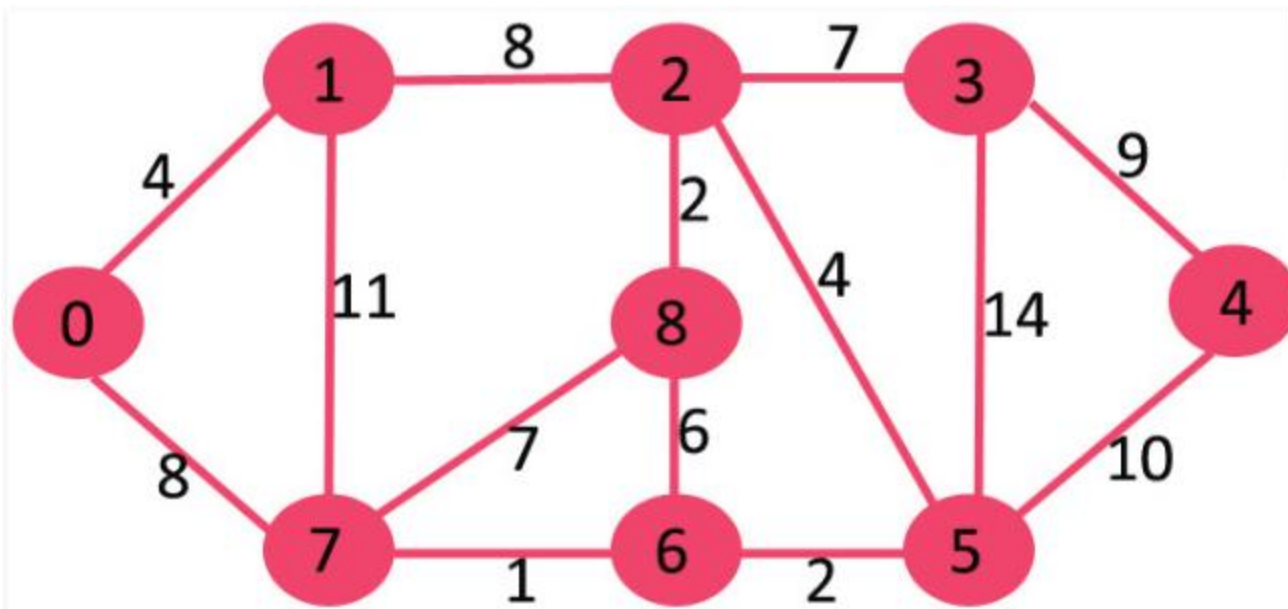
(c)



PRIM'S ALGORITHM

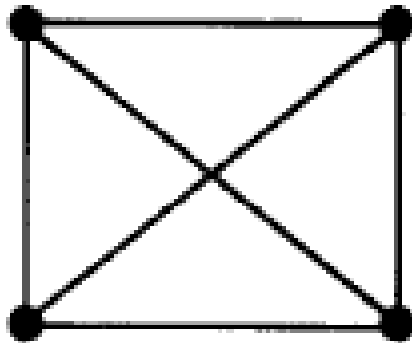
- Start with the initial vertex
- Look for its shortest path
- Choose that vertex
- Follow the same for the rest vertices too in order to get the minimum spanning tree.



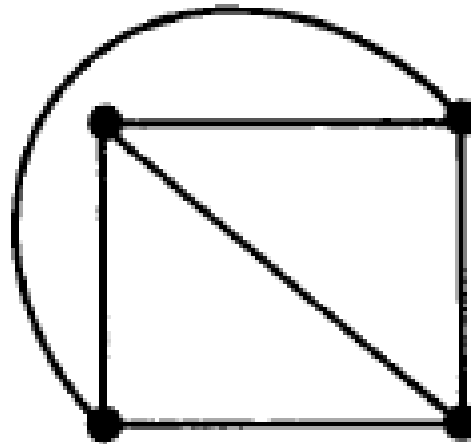


PLANAR GRAPHS

- A graph or multigraph which can be drawn in the plane so that its edges do not cross is said to be *planar*.



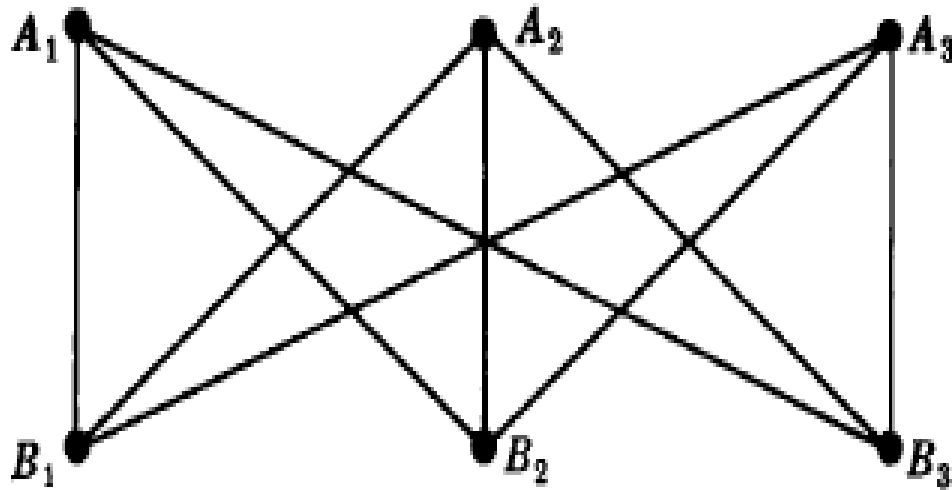
(a)



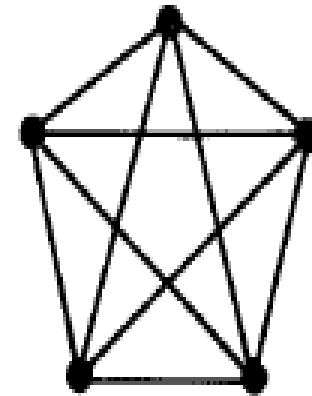
(b)



- A graph is nonplanar if and only if it contains a subgraph homeomorphic to $K_{3,3}$ or K_5 .



(a) $K_{3,3}$



(b) K_5



MAPS, REGIONS

- A particular planar representation of a finite planar multigraph is called a *map*
- The map is *connected* if the underlying multigraph is connected.
- A given map divides the plane into various regions.
- For example, the map in Fig. with six vertices and nine edges divides the plane into five regions.
- The *degree* of a region r , written $\deg(r)$, is the length of the cycle or closed walk which borders r .

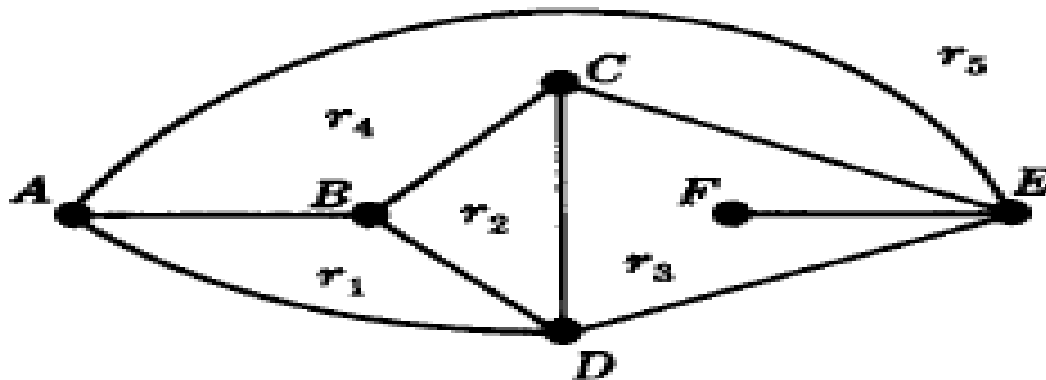


MAPS, REGIONS

- The sum of the degrees of the regions of a map is equal to twice the number of edges.

$$\deg(r_1) = 3, \deg(r_2) = 3, \deg(r_3) = 5, \deg(r_4) = 4, \deg(r_5) = 3$$

- The sum of the degrees is 18, which, as expected, is twice the number of edges.





EULER'S FORMULA

- Euler showed that all planer representations of a graph partition the plane into the same number of regions.
- Theorem (Euler's Formula)
If $G = (V; E)$ is a connected planar graph with r regions, v vertices, and e edges, then $v + r - e = 2$.



PROOF

- Starting with subgraph of G consisting a single vertex, we'll add edges and vertices (keeping the subgraphs connected) until we construct G .
 - The basis step is $R1$ 
 - Here $r = 1$, $v = 1$, $e = 0$ so $r + v = e + 2$ becomes $1 + 1 = 0 + 2$, which is true.
 - Assume that we have a subgraph of G with $e = k$ edges and that $r + v = e + 2$ holds.
 - We now draw the the $k + 1$ st edge.
 - There are two cases:
 - 1) Both vertices the new edge is incident to are already on the graph. Since the subgraph is connected, this will create a new region. Thus both r and e increase by one so $r + v = e + 2$ is still true.
 - 2) A new pendant vertex is introduced with the new edge. This does not increase r but does increase both e and v by one, so $r + v = e + 2$ continues to hold.
 - Since we can continue with these steps until we have constructed G , we conclude that $r + v = e + 2$ holds for any connected planar graph.
- 

GRAPH COLORINGS

- Consider a graph G . *A vertex coloring, or simply a coloring of G is an assignment of colors to the vertices of G such that adjacent vertices have different colors*

Algorithm 8.4 (Welch-Powell): The input is a graph G .

Step 1. Order the vertices of G according to decreasing degrees.

Step 2. Assign the first color C_1 to the first vertex and then, in sequential order, assign C_1 to each vertex which is not adjacent to a previous vertex which was assigned C_1 .

Step 3. Repeat Step 2 with a second color C_2 and the subsequence of noncolored vertices.

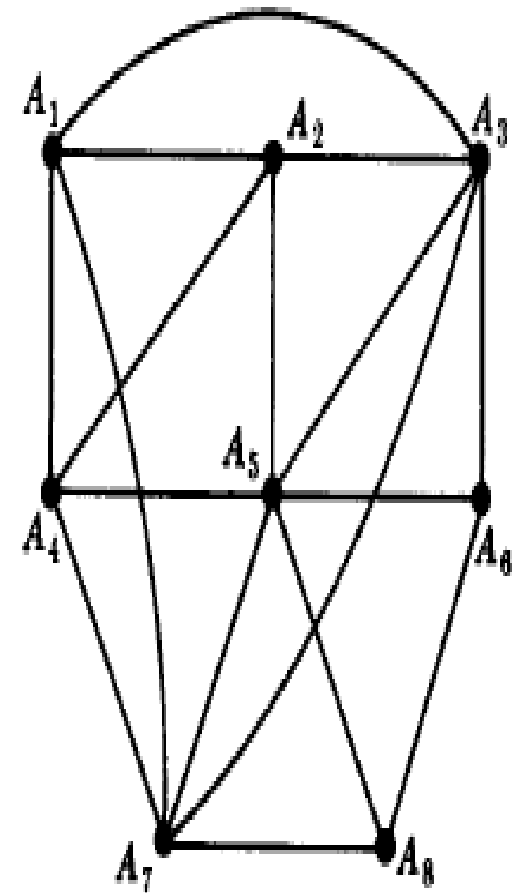
Step 4. Repeat Step 3 with a third color C_3 , then a fourth color C_4 , and so on until all vertices are colored.

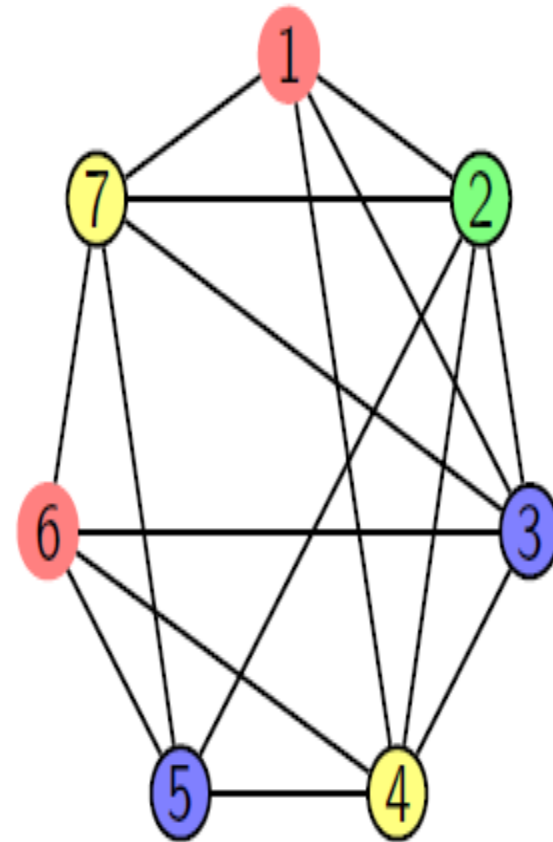
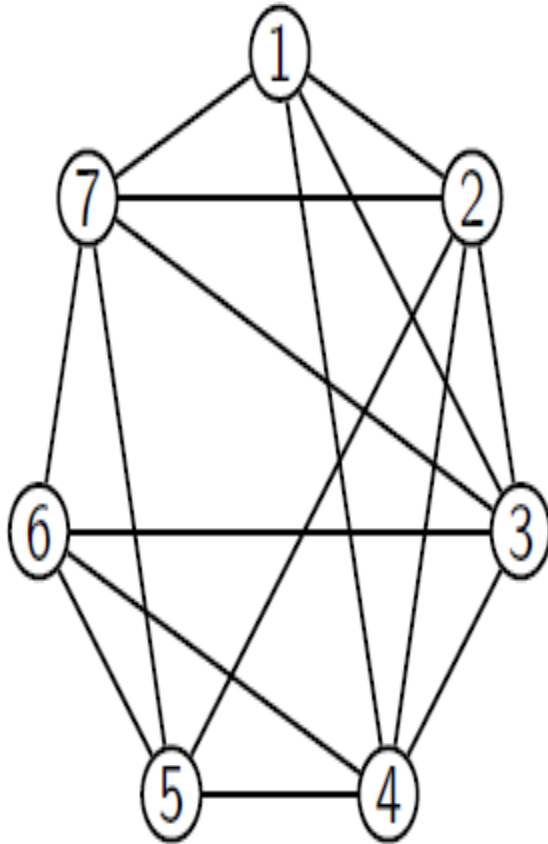
Step 5. Exit.

- We use the Welch-Powell Algorithm to obtain a coloring of G . *Ordering* the vertices according to decreasing degrees yields the following sequence:

$A_5, A_3, A_7, A_1, A_2, A_4, A_6, A_8$

- The first color is assigned to vertices A_5 and A_1 . The second color is assigned to vertices A_3, A_4 , and A_8 .
- The third color is assigned to vertices A_7, A_2 , and A_6 . All the vertices have been assigned a color, and so G is 3-colorable.
- Observe that G is not 2-colorable since vertices A_1, A_2 , and A_3 , which are connected to each other, must be assigned different colors. Accordingly, $\chi(G) = 3$.





The chromatic number is 4



DIRECTED GRAPHS

- *Directed graph G or digraph (or simply graph) consists of two things:*
 - (i) *A set V whose elements are called vertices, nodes, or points.*
 - (ii) *A set E of ordered pairs (u, v) of vertices called arcs or directed edges or simply edges*
- *Suppose $e = (u, v)$ is a directed edge in a digraph G . Then the following terminology is used:*
 - (a) *e begins at u and ends at v .*
 - (b) *u is the origin or initial point of e , and v is the destination or terminal point of e .*
 - (c) *v is a successor of u .*
 - (d) *u is adjacent to v , and v is adjacent from u .*
- *If $u = v$, then e is called a loop.*



◦ Degrees

Suppose G is a directed graph. The outdegree of a vertex v of G , written $\text{outdeg}(v)$, is the number of edges beginning at v , and the indegree of v , written $\text{indeg}(v)$, is the number of edges ending at v .

Since each edge begins and ends at a vertex we immediately obtain the following theorem.

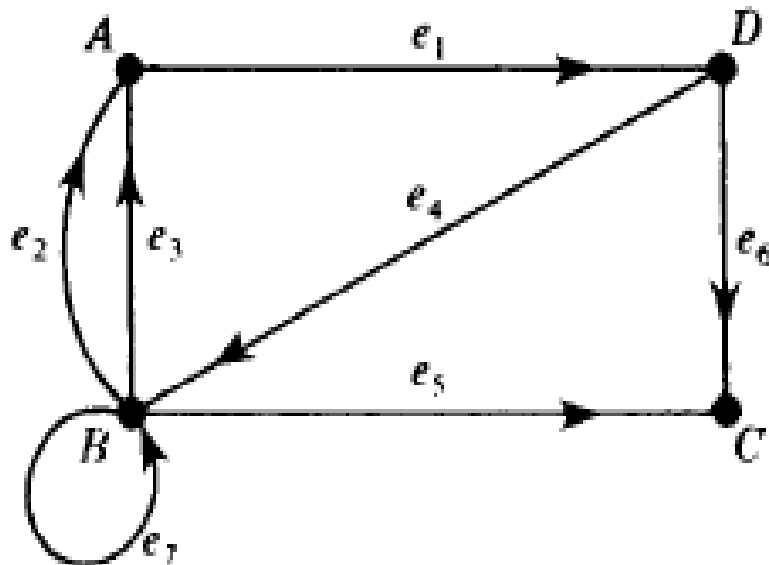
The sum of the outdegrees of the vertices of a digraph G equals the sum of the indegrees of the vertices, which equals the number of edges in G .

A vertex v with zero indegree is called a source, and a vertex v with zero outdegree is called a sink.



$\text{outdeg}(A) = 1$, $\text{outdeg}(B) = 4$, $\text{outdeg}(C) = 0$, $\text{outdeg}(D) = 2$,
 $\text{indeg}(A) = 2$, $\text{indeg}(B) = 2$, $\text{indeg}(C) = 2$, $\text{indeg}(D) = 1$.

- As expected, the sum of the outdegrees equals the sum of the indegrees, which equals the number 7 of edges.
- The vertex C is a sink since no edge begins at C . The graph has no sources.

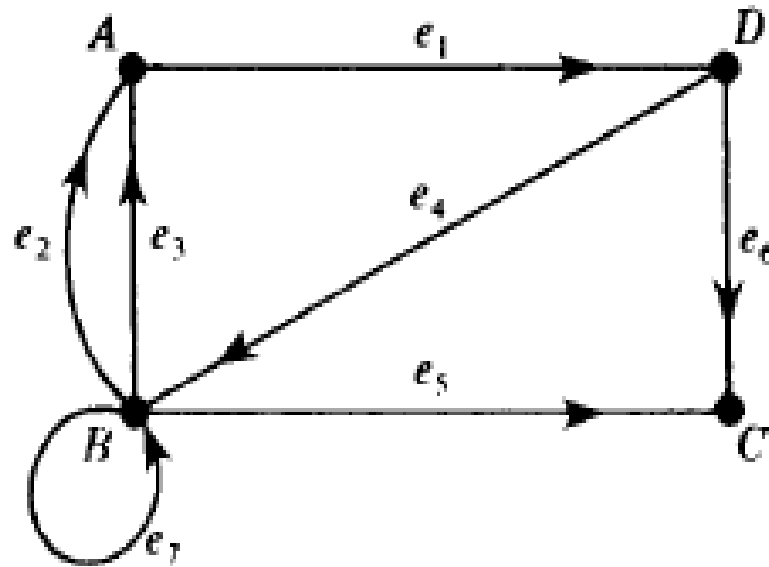


PATHS

- (i) A (directed) path P in G is an alternating sequence of vertices and directed edges, say,
 $P = v_0, e_1, v_1, e_2, v_2, \dots, e_n, v_n$ such that each edge $e(i)$ begins at v_{i-1} and ends at v_i . If there is no ambiguity, we denote P by its sequence of vertices or its sequence of edges.
- (ii) The length of the path P is n , its number of edges.
- (iii) A simple path is a path with distinct vertices. A trail is a path with distinct edges.
- (iv) A closed path has the same first and last vertices.
- (v) A spanning path contains all the vertices of G .
- (vi) A cycle (or circuit) is a closed path with distinct vertices (except the first and last).
- (vii) A semipath is the same as a path except the edge $e(i)$ may begin at v_{i-1} or v_i and end at the other vertex.

Semi trails and semi simple paths are analogously defined.





- (a) The sequence $P1 = (D, C, B, A)$ is a *semipath* but not a *path* since (C, B) is not an edge; that is, the direction of $e5 = (C, B)$ does not agree with the direction of $P1$.
- (b) The sequence $P2 = (D, B, A)$ is a *path* from D to A since (D, B) and (B, A) are edges. Thus A is *reachable* from D .

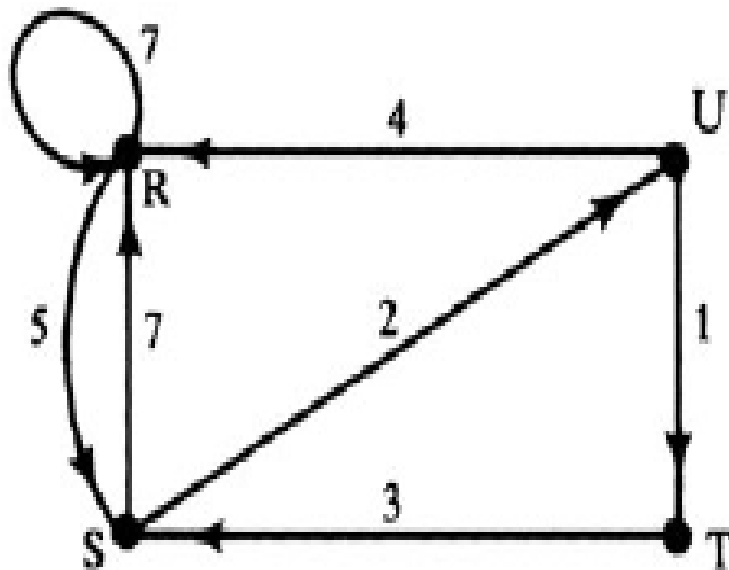
WARSHALL'S ALGORITHM : SHORTEST PATHS

Algorithm 9.1 (Warshall's Algorithm): A directed graph G with M vertices is maintained in memory by its adjacency matrix A . This algorithm finds the (Boolean) path matrix P of the graph G .

- Step 1.** Repeat for $I, J = 1, 2, \dots, M$: [Initializes P .]
 If $A[I, J] = 0$, then: Set $P[I, J] := 0$;
 Else: Set $P[I, J] := 1$.
[End of loop.]
- Step 2.** Repeat Steps 3 and 4 for $K = 1, 2, \dots, M$: [Updates P .]
- Step 3.** Repeat Step 4 for $I = 1, 2, \dots, M$:
- Step 4.** Repeat for $J = 1, 2, \dots, M$:
 Set $P[I, J] := P[I, J] \vee (P[I, K] \wedge P[K, J])$.
[End of loop.]
[End of Step 3 loop.]
[End of Step 2 loop.]
- Step 5.** Exit.

EXAMPLE

- Figure shows a weighted graph G and its weight matrix W where we assume that $v1 = R$, $v2 = S$, $v3 = T$, $v4 = U$.
- Suppose we apply the modified Warshall's algorithm to our weighted graph G in Fig.
- We will obtain the matrices $Q0$, $Q1$, $Q3$, and $Q4$



$$W = \begin{bmatrix} 7 & 5 & 0 & 0 \\ 7 & 0 & 0 & 2 \\ 0 & 3 & 0 & 0 \\ 4 & 0 & 1 & 0 \end{bmatrix}$$



$$Q_1[4, 2] = \text{MIN} (Q_0[4, 2], Q_0[4, 1] + Q_0[1, 2]) = \text{MIN}(\infty, 4 + 5) = 9$$

$$Q_2[1, 3] = \text{MIN} (Q_1[1, 3], Q_1[1, 2] + Q_1[2, 3]) = \text{MIN}(\infty, 5 + \infty) = \infty$$

$$Q_3[4, 2] = \text{MIN} (Q_2[4, 2], Q_2[4, 3] + Q_2[3, 2]) = \text{MIN}(9, 3 + 1) = 4$$

$$Q_4[3, 1] = \text{MIN} (Q_3[3, 1], Q_3[3, 4] + Q_3[4, 1]) = \text{MIN}(10, 5 + 4) = 9$$

The last matrix $Q_4 = Q$, the desired shortest-path matrix.

$$Q_0 = \begin{bmatrix} 7 & 5 & \infty & \infty \\ 7 & \infty & \infty & 2 \\ \infty & 3 & \infty & \infty \\ 4 & \infty & 1 & \infty \end{bmatrix} \quad \begin{bmatrix} \text{RR} & \text{RS} & \text{—} & \text{—} \\ \text{SR} & \text{—} & \text{—} & \text{SU} \\ \text{—} & \text{TS} & \text{—} & \text{—} \\ \text{UR} & \text{—} & \text{UT} & \text{—} \end{bmatrix}$$

$$Q_1 = \begin{bmatrix} 7 & 5 & \infty & \infty \\ 7 & 12 & \infty & 2 \\ \infty & 3 & \infty & \infty \\ 4 & \textcircled{9} & 1 & \infty \end{bmatrix} \quad \begin{bmatrix} \text{RR} & \text{RS} & \text{—} & \text{—} \\ \text{SR} & \text{SRS} & \text{—} & \text{SU} \\ \text{—} & \text{TS} & \text{—} & \text{—} \\ \text{UR} & \text{URS} & \text{UT} & \text{—} \end{bmatrix}$$

$$Q_2 = \begin{bmatrix} 7 & 5 & \textcircled{\infty} & 7 \\ 7 & 12 & \infty & 2 \\ 10 & 3 & \infty & 5 \\ 4 & 9 & 1 & 11 \end{bmatrix} \quad \begin{bmatrix} \text{RR} & \text{RS} & \text{—} & \text{RSU} \\ \text{SR} & \text{SRS} & \text{—} & \text{SU} \\ \text{TSR} & \text{TS} & \text{—} & \text{TSU} \\ \text{UR} & \text{URS} & \text{UT} & \text{URS} \end{bmatrix}$$

$$Q_3 = \begin{bmatrix} 7 & 5 & \infty & 7 \\ 7 & 12 & \infty & 2 \\ 10 & 3 & \infty & 5 \\ 4 & \textcircled{4} & 1 & 6 \end{bmatrix} \quad \begin{bmatrix} \text{RR} & \text{RS} & \text{—} & \text{RSU} \\ \text{SR} & \text{SRS} & \text{—} & \text{SU} \\ \text{TSR} & \text{TS} & \text{—} & \text{TSU} \\ \text{UR} & \text{UTS} & \text{UT} & \text{UTSU} \end{bmatrix}$$

$$Q_4 = \begin{bmatrix} 7 & 5 & 8 & 7 \\ 7 & 11 & 3 & 2 \\ \textcircled{9} & 3 & 6 & 5 \\ 4 & 4 & 1 & 6 \end{bmatrix} \quad \begin{bmatrix} \text{RR} & \text{RS} & \text{RSUT} & \text{RSU} \\ \text{SR} & \text{SURS} & \text{SUT} & \text{SU} \\ \text{TSUR} & \text{TS} & \text{TSUT} & \text{TSU} \\ \text{UR} & \text{UTS} & \text{UT} & \text{UTSU} \end{bmatrix}$$

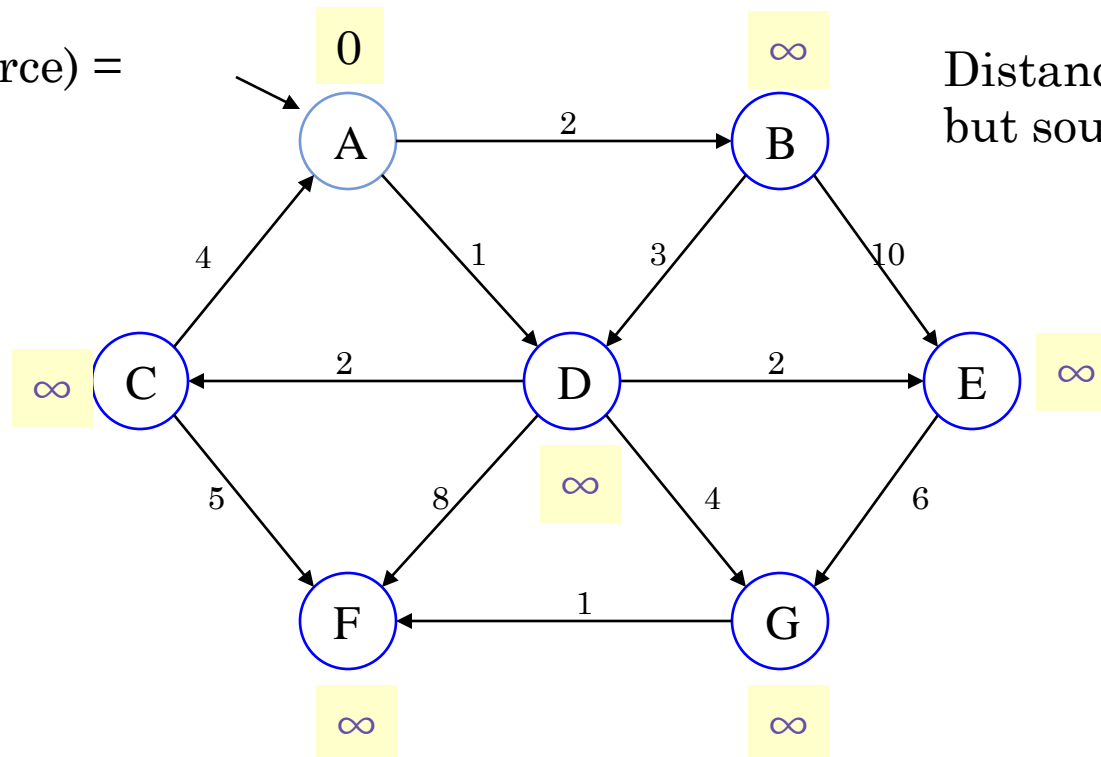
DIJKSTRA'S ALGO: SHORTEST PATH ALGORITHM

- The algorithm computes for each vertex u the **distance** to u from the start vertex v , that is, the weight of a shortest path between v and u .
- the algorithm keeps track of the set of vertices for which the distance has been computed, called the **cloud** C
- Every vertex has a label D associated with it. For any vertex u , $D[u]$ stores an approximation of the distance between v and u . The algorithm will update a $D[u]$ value when it finds a shorter path from v to u .
- When a vertex u is added to the cloud, its label $D[u]$ is equal to the actual (final) distance between the starting vertex v and vertex u .



EXAMPLE: INITIALIZATION

Distance(source) =
0

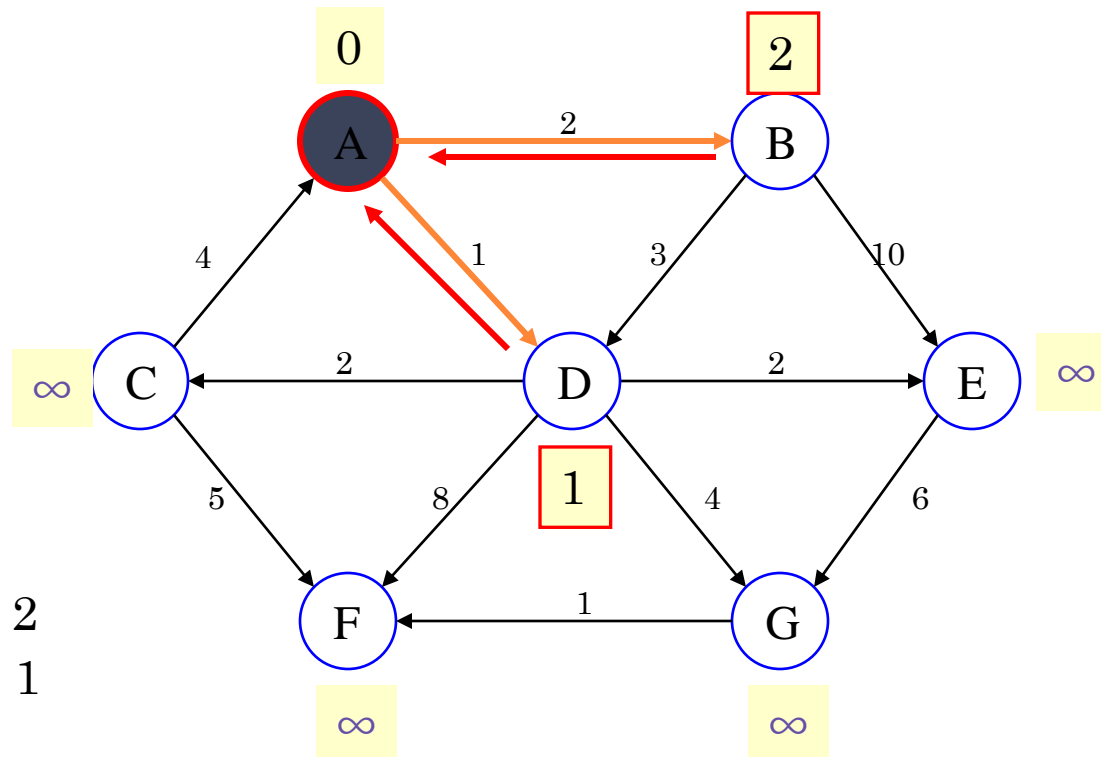


Distance (all vertices
but source) = ∞

Pick vertex in List with minimum distance.



EXAMPLE: UPDATE NEIGHBORS' DISTANCE

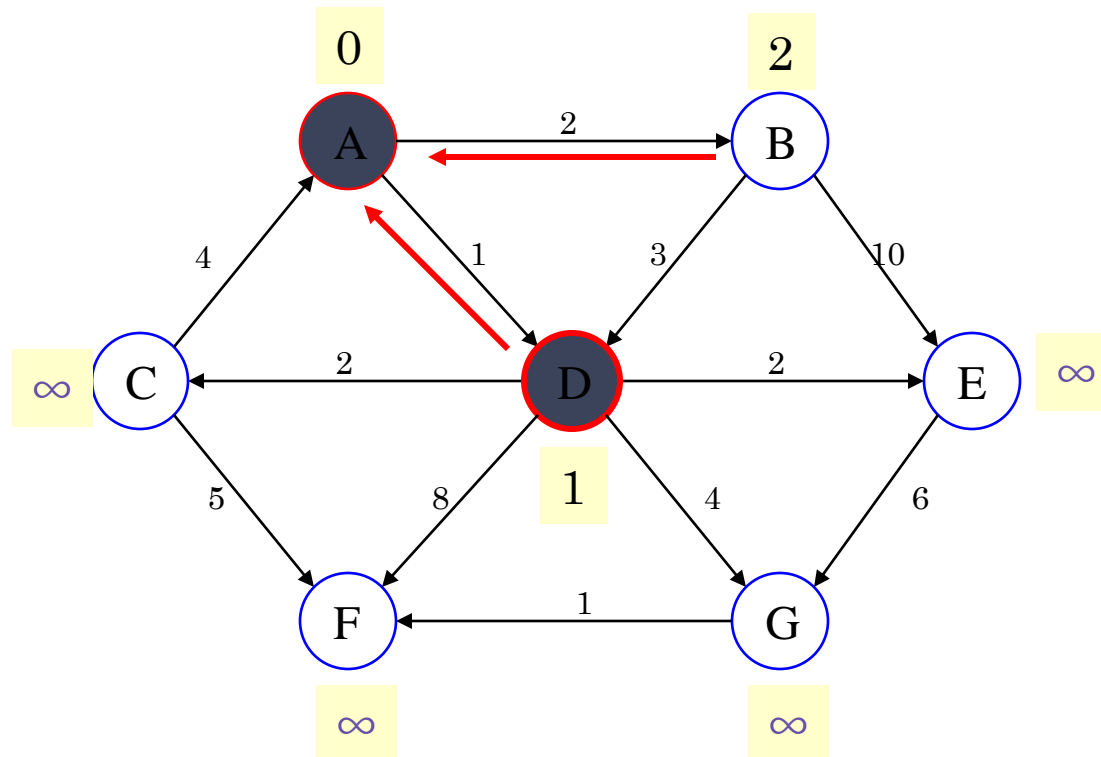


Distance(B) = 2

Distance(D) = 1



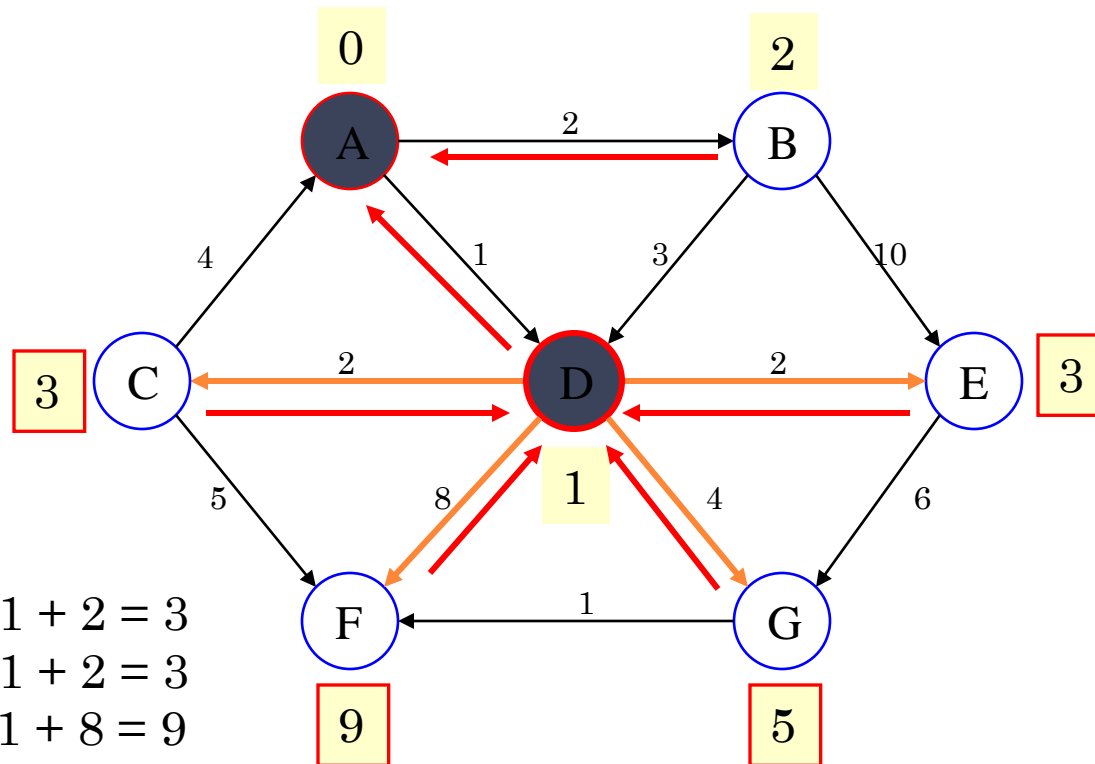
EXAMPLE: REMOVE VERTEX WITH MINIMUM DISTANCE



Pick vertex in List with minimum distance, i.e., D



EXAMPLE: UPDATE NEIGHBORS



Distance(C) = 1 + 2 = 3

Distance(E) = 1 + 2 = 3

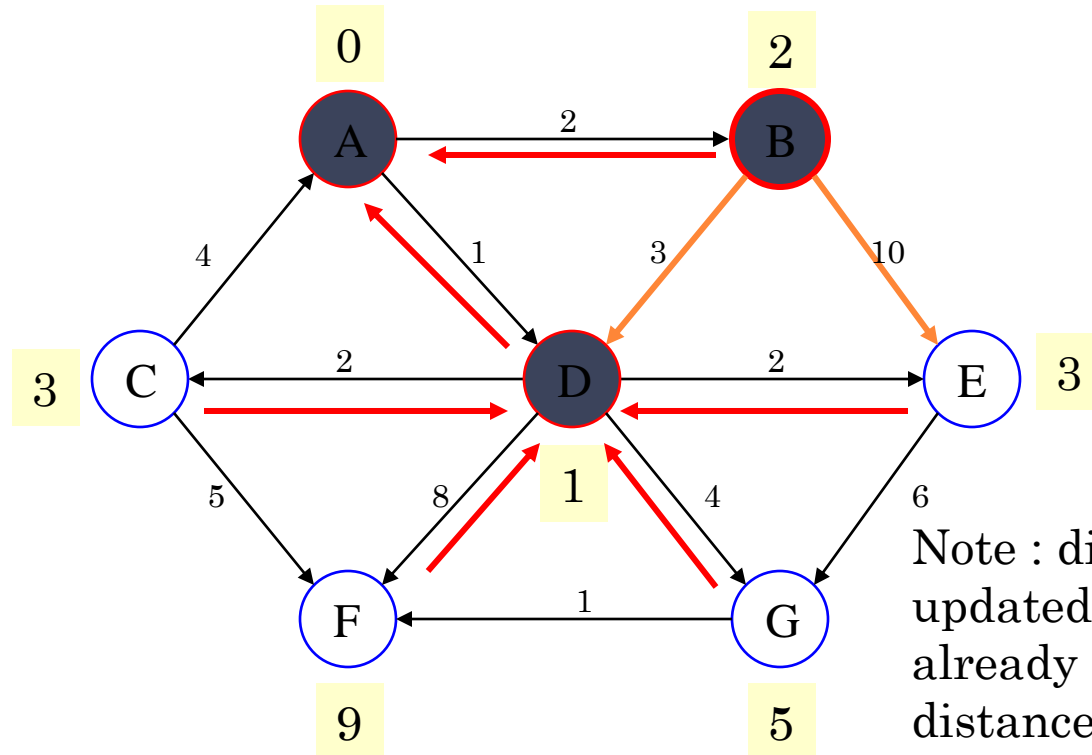
Distance(F) = 1 + 8 = 9

Distance(G) = 1 + 4 = 5



EXAMPLE: CONTINUED...

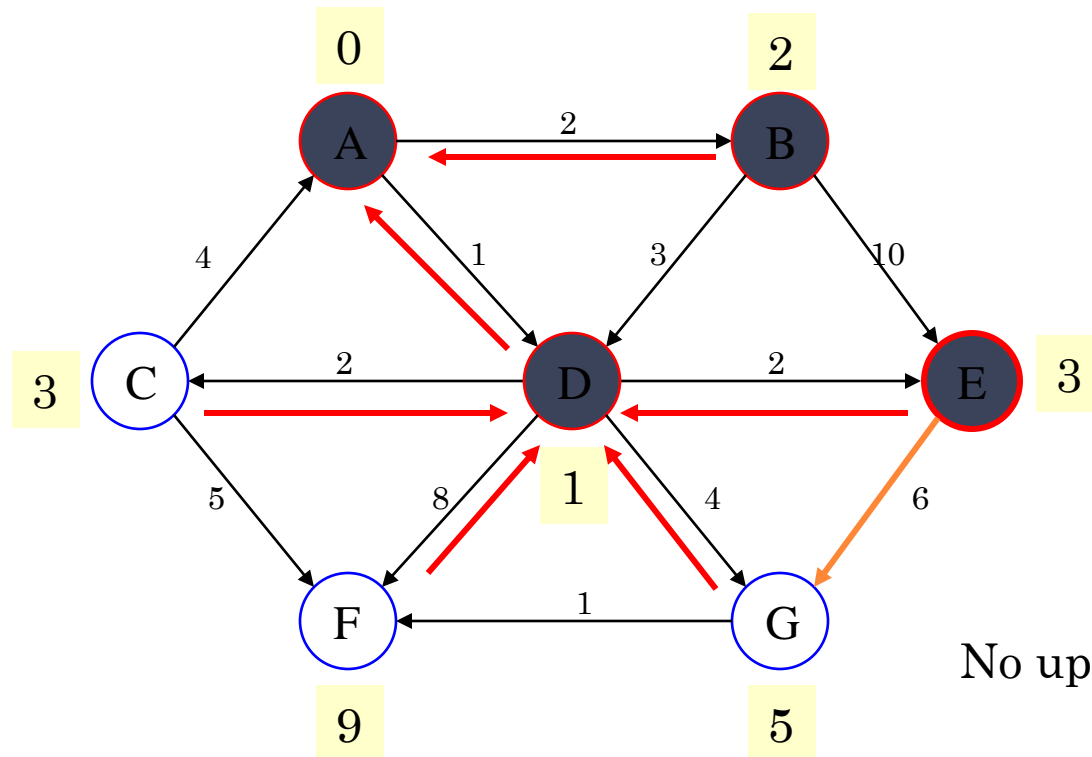
Pick vertex in List with minimum distance (B) and update neighbors



Note : distance(D) not updated since D is already known and distance(E) not updated since it is larger than previously computed

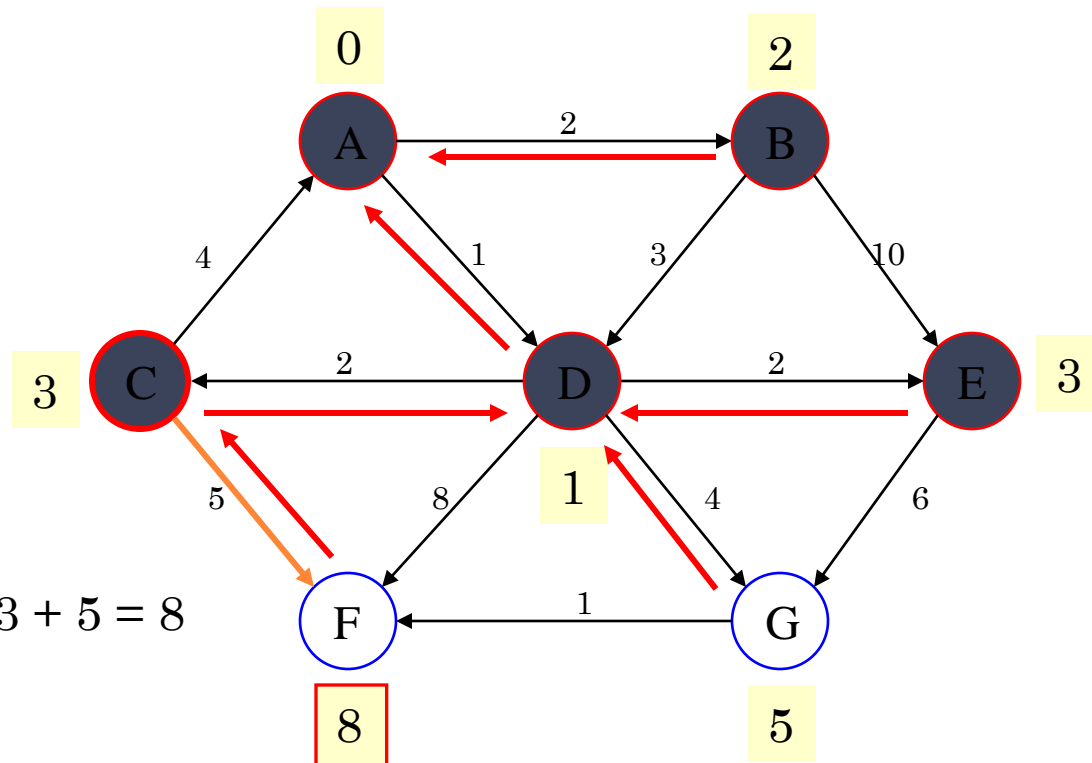
EXAMPLE: CONTINUED...

Pick vertex List with minimum distance (E) and update neighbors



EXAMPLE: CONTINUED...

Pick vertex List with minimum distance (C) and update neighbors

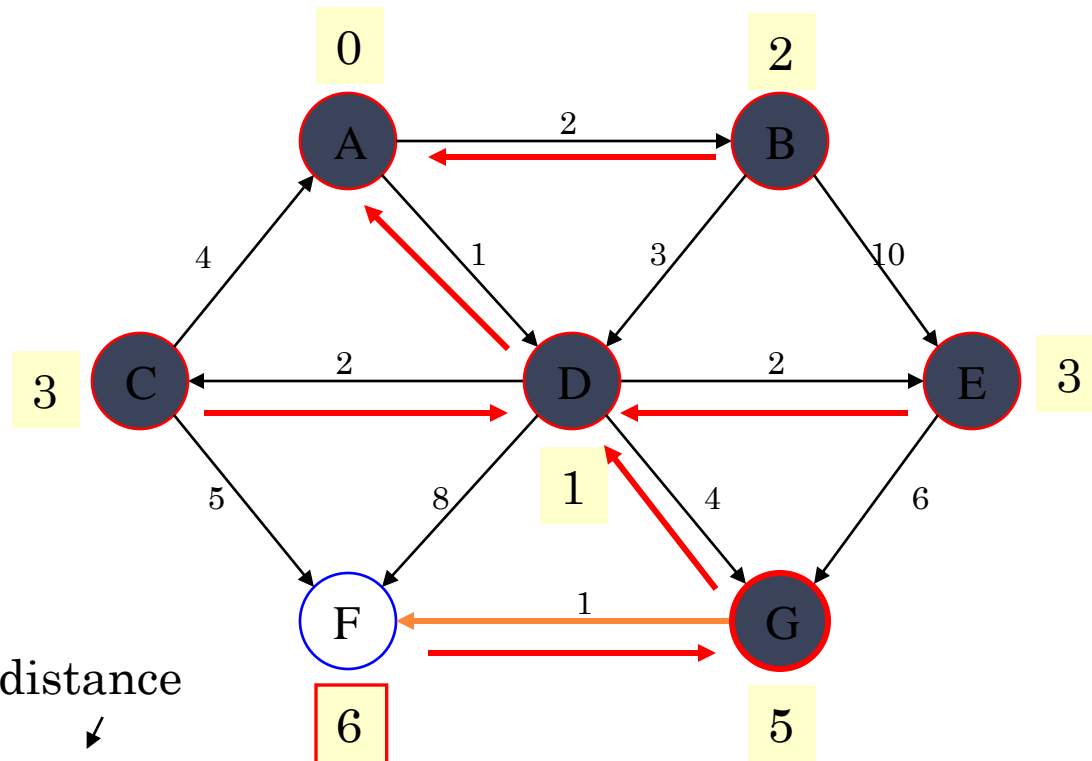


$$\text{Distance}(F) = 3 + 5 = 8$$



EXAMPLE: CONTINUED...

Pick vertex List with minimum distance (G) and update neighbors



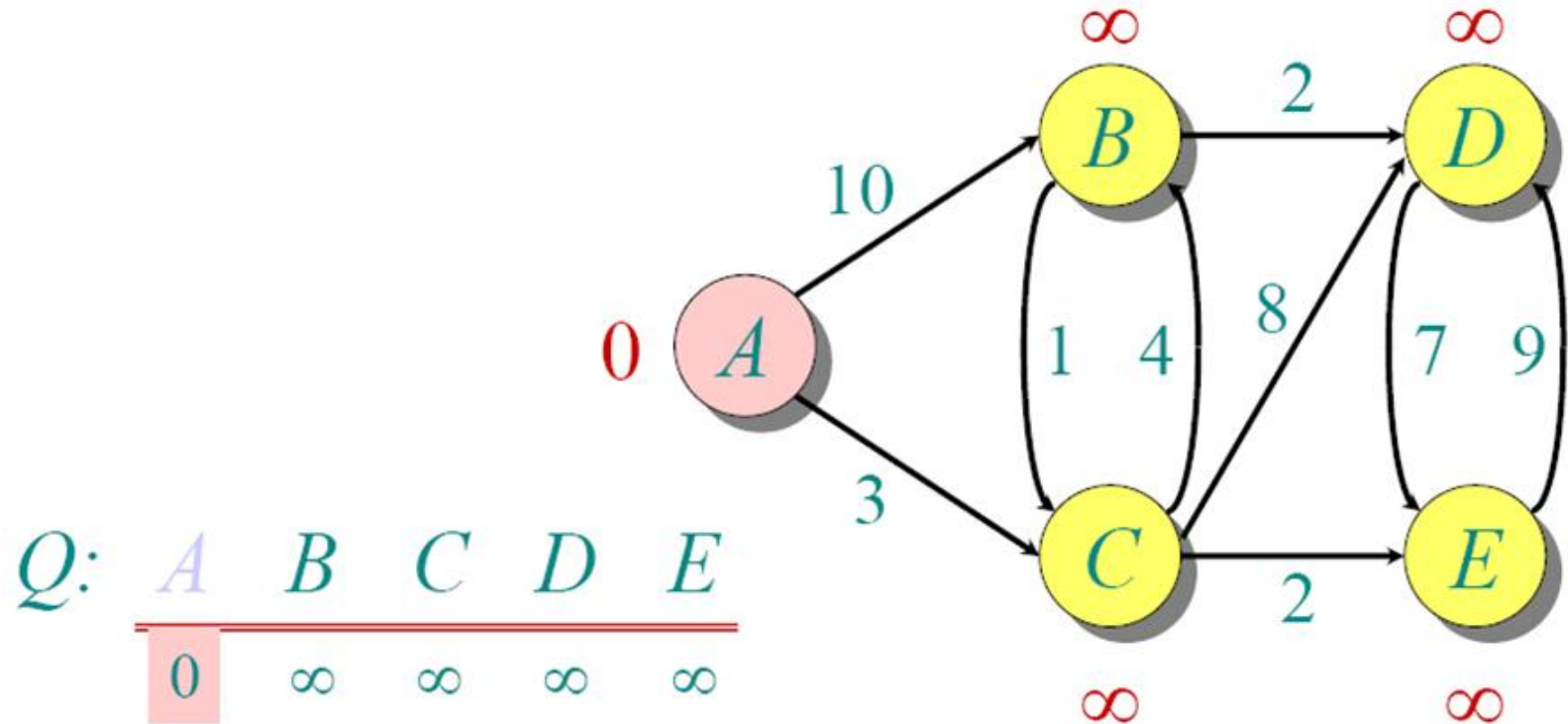
Previous distance



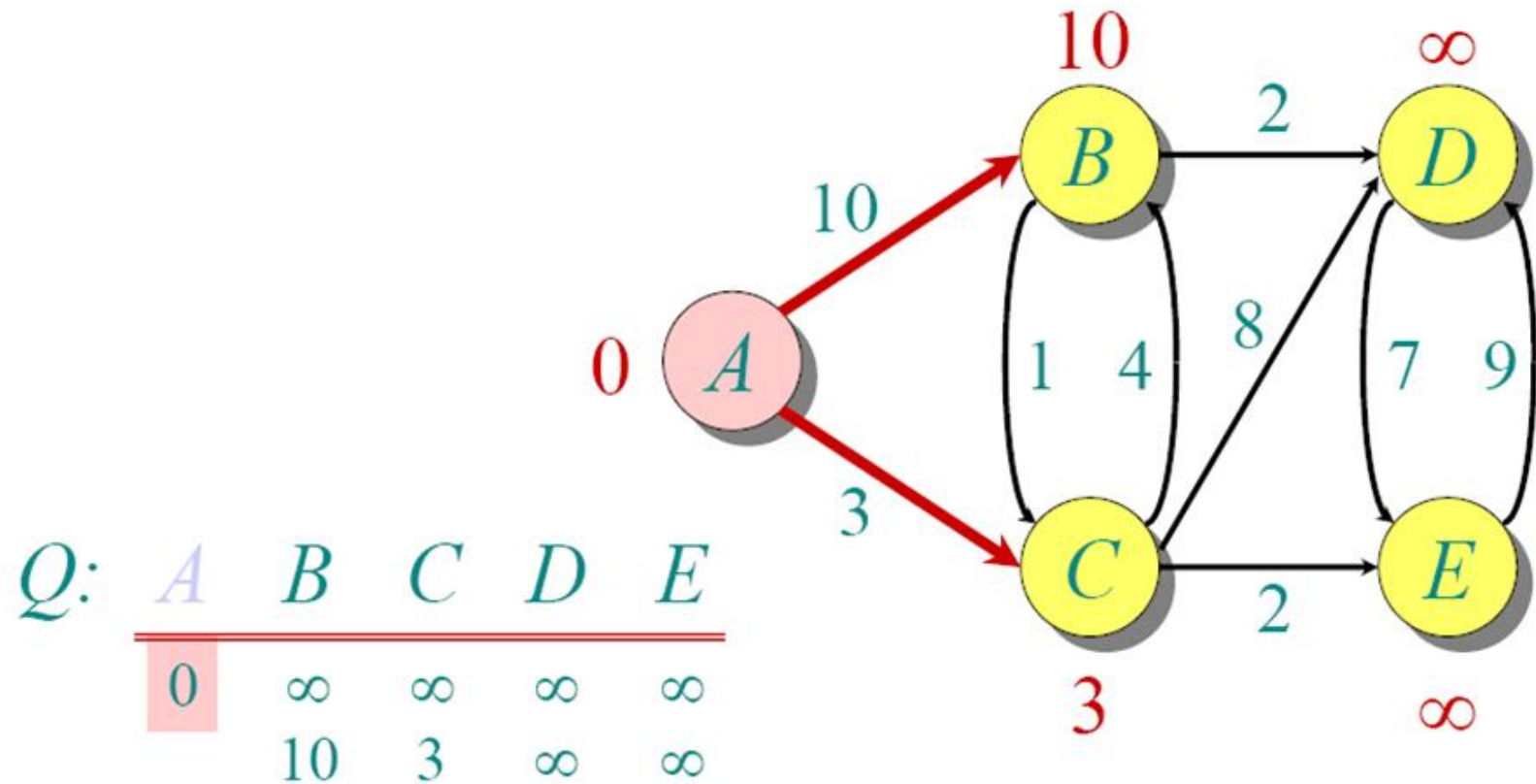
$$\text{Distance}(F) = \min (8, 5+1) = 6$$



ANOTHER EXAMPLE



ANOTHER EXAMPLE



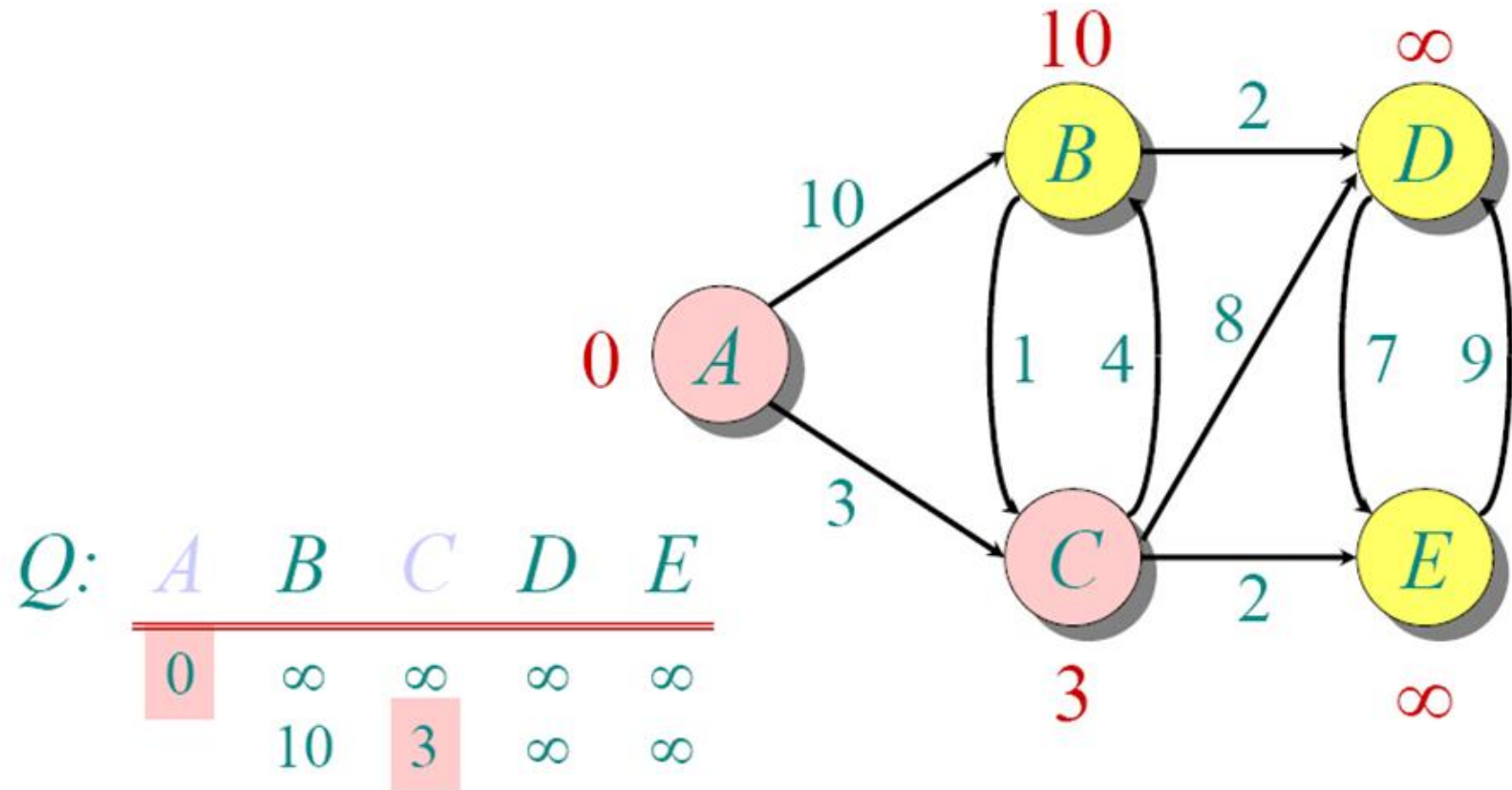
$Q:$

A	B	C	D	E
0	∞	∞	∞	∞
	10	3	∞	∞

$S: \{A\}$



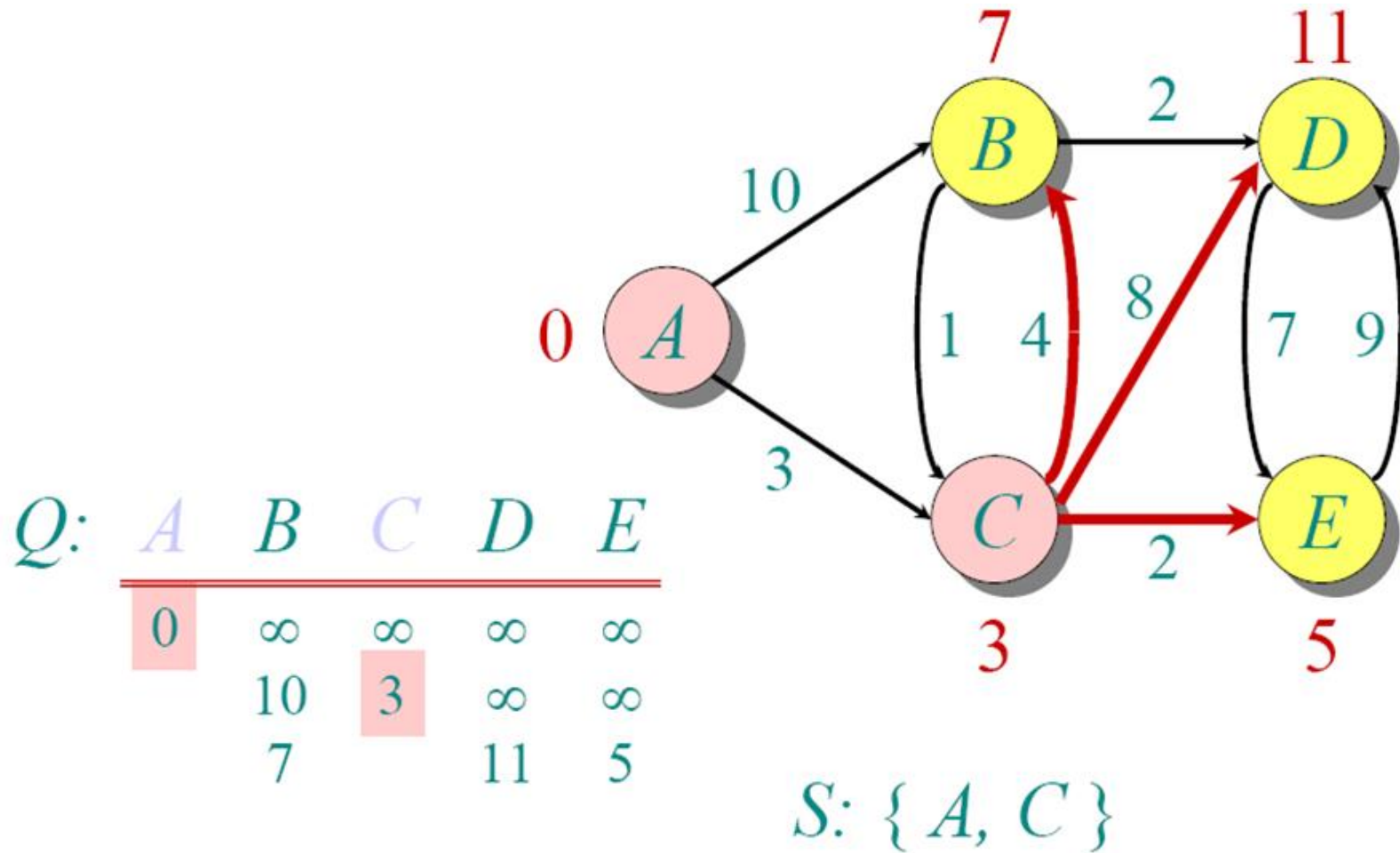
ANOTHER EXAMPLE



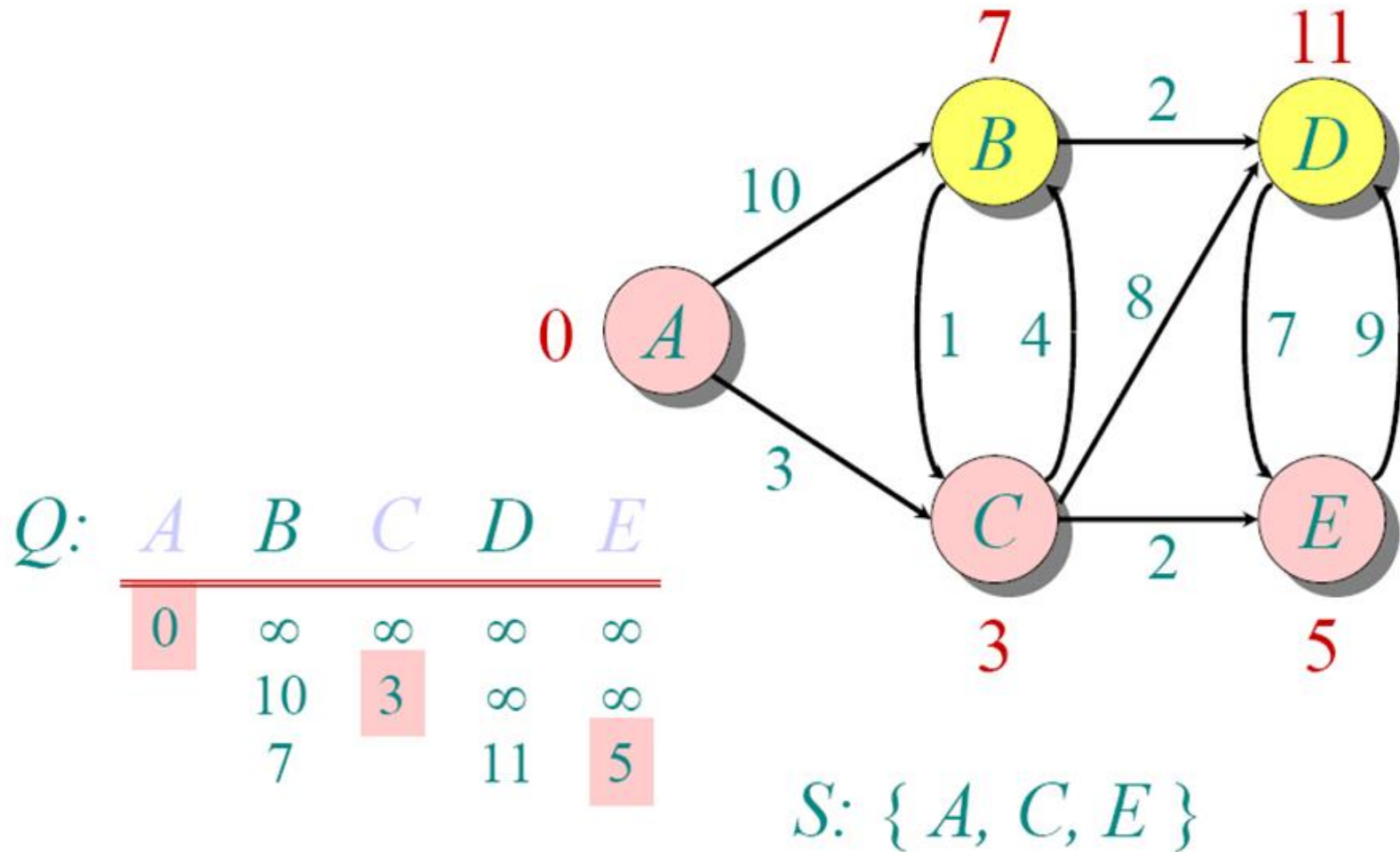
S: { A, C }



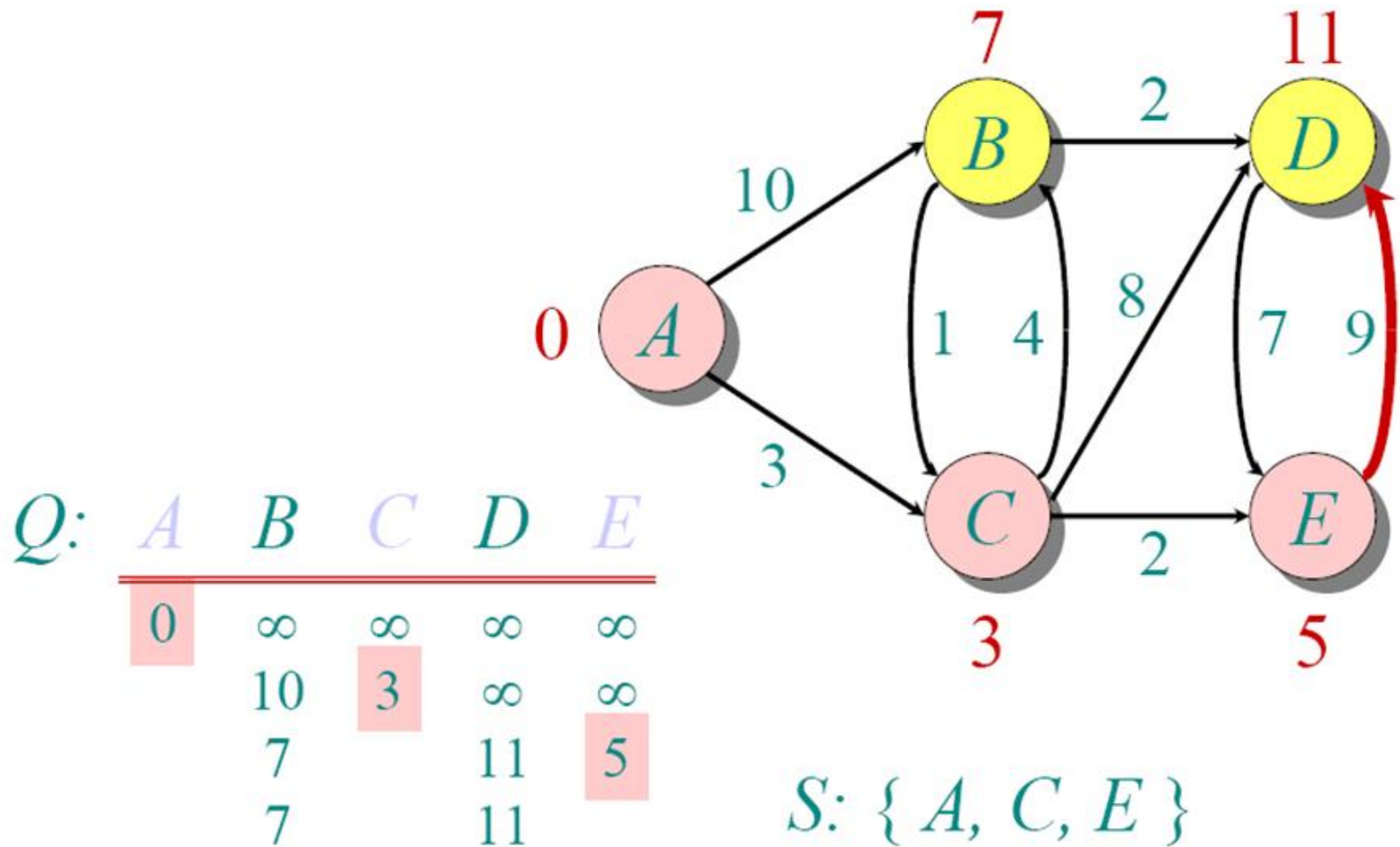
ANOTHER EXAMPLE



ANOTHER EXAMPLE



ANOTHER EXAMPLE



ANOTHER EXAMPLE

