ADITYA ENGINEERING COLLEGE(A)

UNIT-5

TREES

By
D.V. L. Prasanna
Sr. Asst. Professor
Aditya Engineering College(A)

Contents

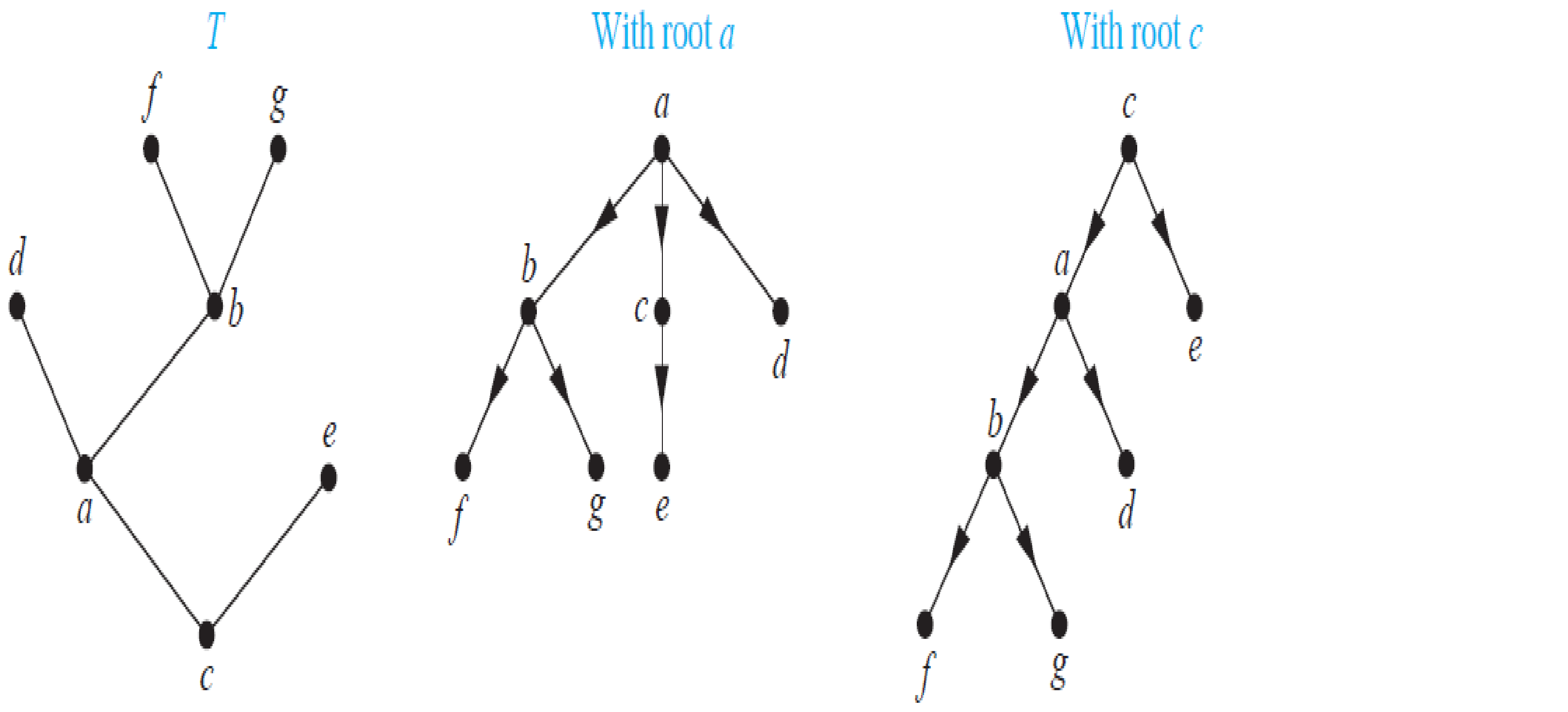
Trees-Properties, Spanning trees, BFS Algorithm, DFS Algorithm, Minimal Spanning Trees and Kruskal's Algorithm, Graph Colouring, Chromatic Number

TREES

- A connected graph that contains no cycles is called a Tree.
- Used to construct efficient algorithms for locating items in a list
- To construct efficient codes saving costs in data transmission and storage
- Can be used to determine winning strategies for playing chess
- Family trees are graphs that represent genealogical charts. Family members are represented as vertices and edges are used to represent parent-child relationship.

TREES

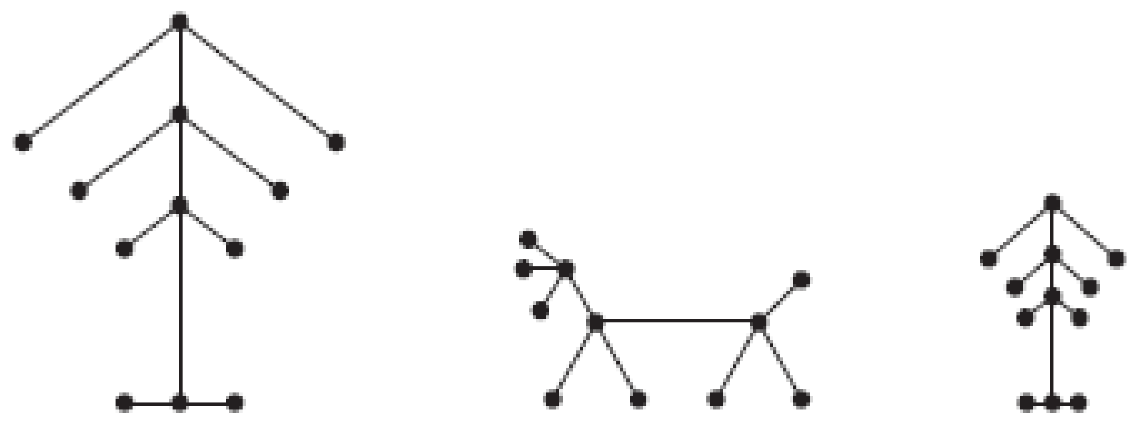
A graph that contains no cycles is called an acyclic graph. A connected acyclic graph is called a Tree. Its edges are called Branches.



Note: A connected graph that contains no simple circuit is a tree.
What about a graph containing no simple circuit that are not necessarily connected?

These graphs are called Forests.

This is one graph with three connected components.



PROPERTIES OF TREES

1)There is only one path between every pair of vertices in a Tree.

2)A tree with ‘n’ vertices has n-1 edges.

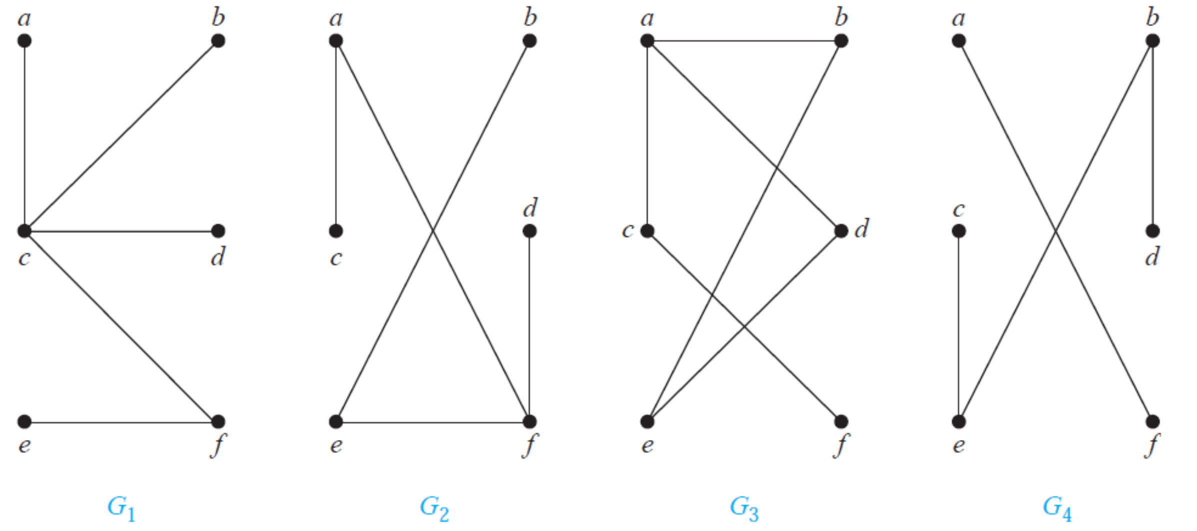
3)If in a graph G there is one and only path between every pair of vertices then G is a tree .

4)For any positive integer n, if G is a connected graph with n vertices and n-1 edges then G is a tree.

5)In a tree with more than one vertex, there are atleast two vertices of degree one.

TREES

Find which of the following graphs are trees?



Solution: G_1 and G_2 are trees, because both are connected graphs with no simple circuits. G_3 is not a tree because e, b, a, d, e is a simple circuit in this graph. Finally, G_4 is not a tree because it is not connected.

Rooted TREES- Terminologies

A *rooted tree* is a tree in which one vertex has been designated as the root and every edge is directed away from the root.

Suppose T is a rooted tree,

1. If v is a vertex in T other than the root, the PARENT of v is a unique vertex u such that there is a directed edge from u to v.
2. When u is the parent of v, v is called a CHILD of u.
3. Vertices with same parents are called SIBILINGS
4. ANCESTORS of a vertex are the vertices in the path from root to this vertex(excluding that vertex and including the root)

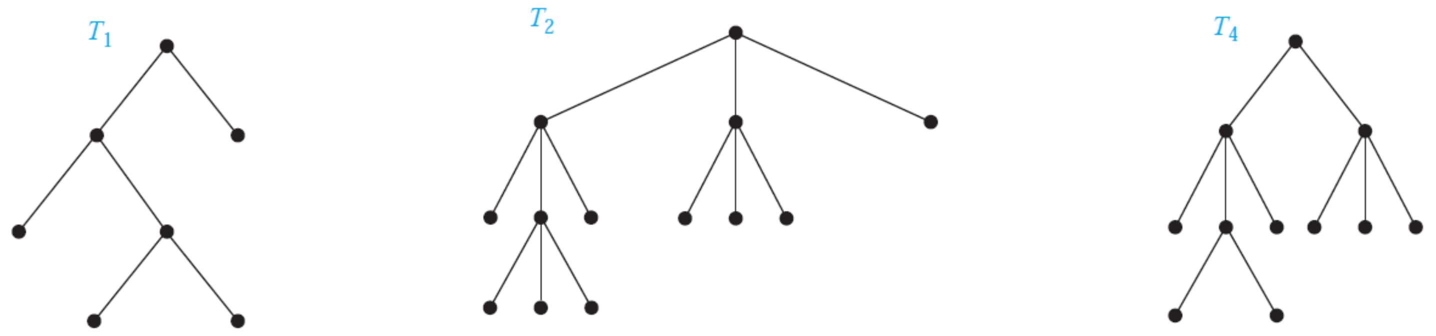
5. DESCENDANTS of a vertex v are the vertices that has v as an ancestor
6. A vertex of a rooted tree is called a LEAF or a TERMINAL VERTEXS if it has no children
7. Vertices that have children are called internal vertices
8. The level of a vertex is the number of edges along its unique path from the root
9. The height of the tree is the maximum level of the tree.

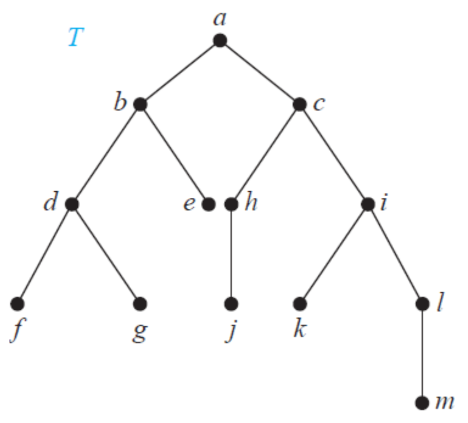
SUBTREE

If a is a vertex ,the SUBTREE with as its root is the subgraph of the tree consisting of a and its descendants and all edges incident to these descendants.

BINARY TREES

A rooted tree is an m-ary tree if every internal vertex has at most m children.
A 2-ary tree is called a Binary tree.





BINARY TREES

A full m-ary tree with i internal vertices contains $n = mi + 1$ vertices.

A full m-ary tree with

(i) n vertices has $i = (n - 1)/m$ internal vertices and $l = [(m - 1)n + 1]/m$ leaves,

(ii) i internal vertices has $n = mi + 1$ vertices and $l = (m - 1)i + 1$ leaves,

(iii) l leaves has $n = (ml - 1)/(m - 1)$ vertices and $i = (l - 1)/(m - 1)$ internal vertices.

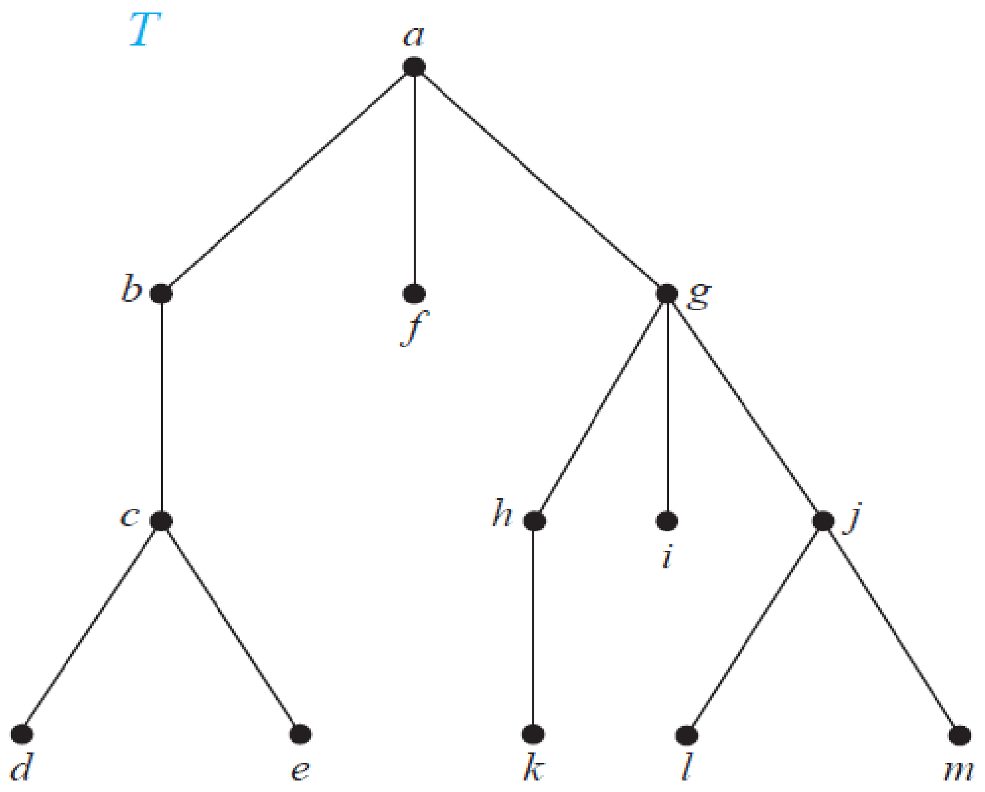
How many edges does a tree with 10,000 vertices have?

How many vertices does a full 5-ary tree with 100 internal vertices have?

How many edges does a full binary tree with 1000 internal vertices have?

How many leaves does a full 3-ary tree with 100 vertices have?

For the tree given below, Find the parent of c , children of g, ancestors of e, descendants of b. Find all the Internal vertices ,Leaves and the subtree rooted at g ?



Solution:

The parent of c is b

The children of g are h,I,j

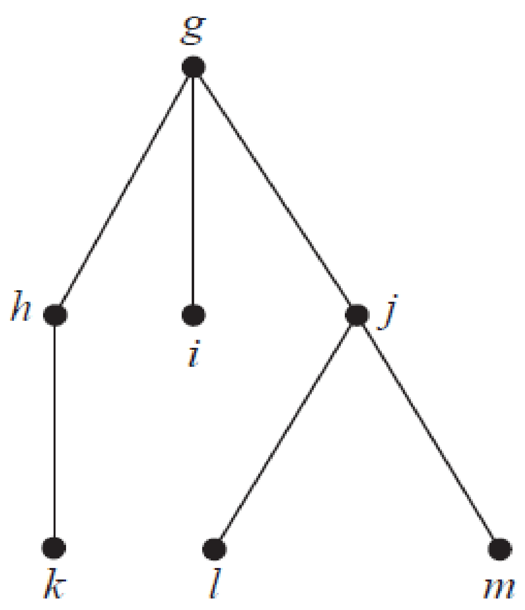
The ancestors of e are c,b and a

The descendants of b are c,d, and e

Internal vertices are a,b,c,g,h and j

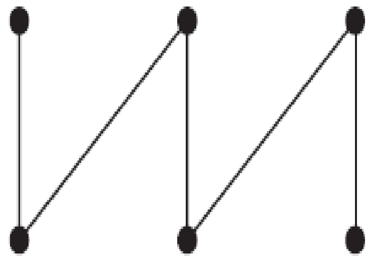
Leaves are d,e,f,I,k,l and m

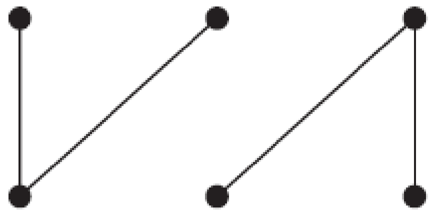
The subtree rooted at g is

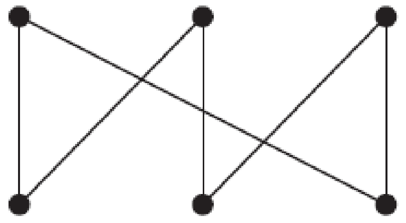


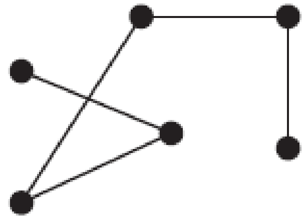
PRACTICE PROBLEMS

1) Which of these graphs are trees?



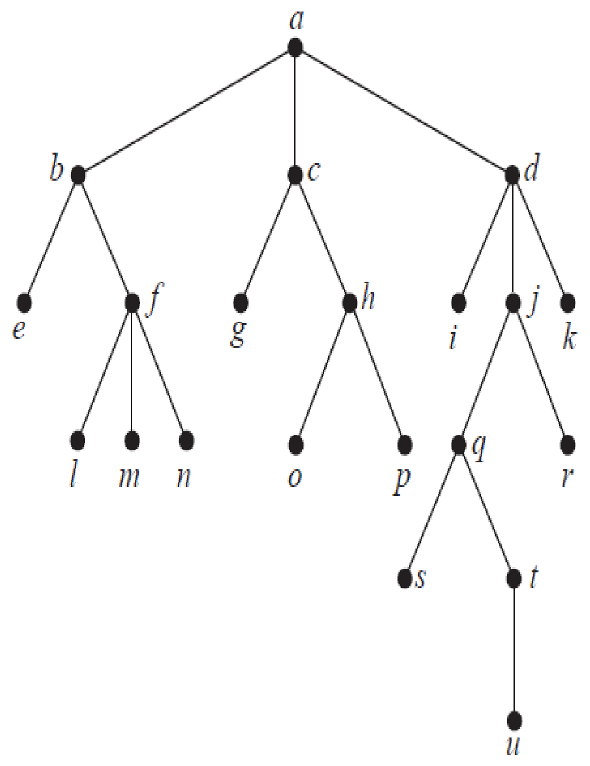






PRACTICE PROBLEMS

Answer these questions about the rooted tree illustrated.



a) Which vertex is the root?

b) Which vertices are internal?

c) Which vertices are leaves?

d) Which vertices are children of j?

e) Which vertex is the parent of h?

f) Which vertices are siblings of o?

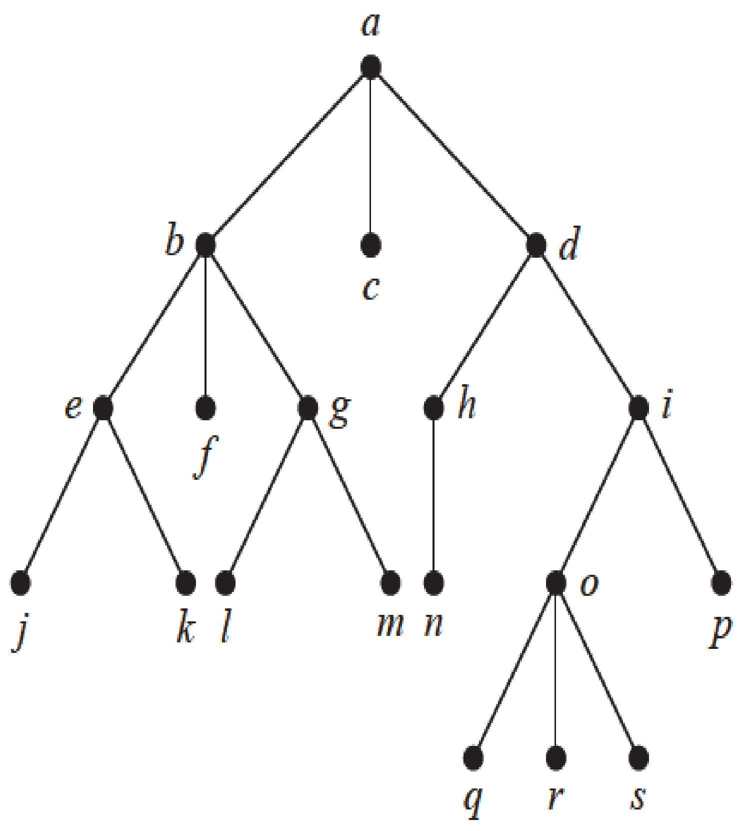
g) Which vertices are ancestors of m?

h) Which vertices are descendants of b?

Also draw the subtrees rooted at d and e

PRACTICE PROBLEMS

Answer these questions about the rooted tree illustrated.



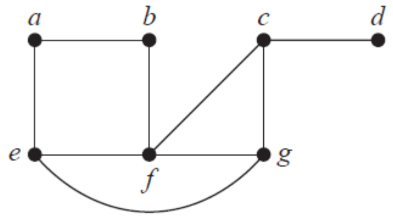
a) Which vertex is the root?
b) Which vertices are internal?
c) Which vertices are leaves?
d) Which vertices are children of j?
e) Which vertex is the parent of h?
f) Which vertices are siblings of o?
g) Which vertices are ancestors of m?
h) Which vertices are descendants of b?

Also draw the subtrees rooted at a and e

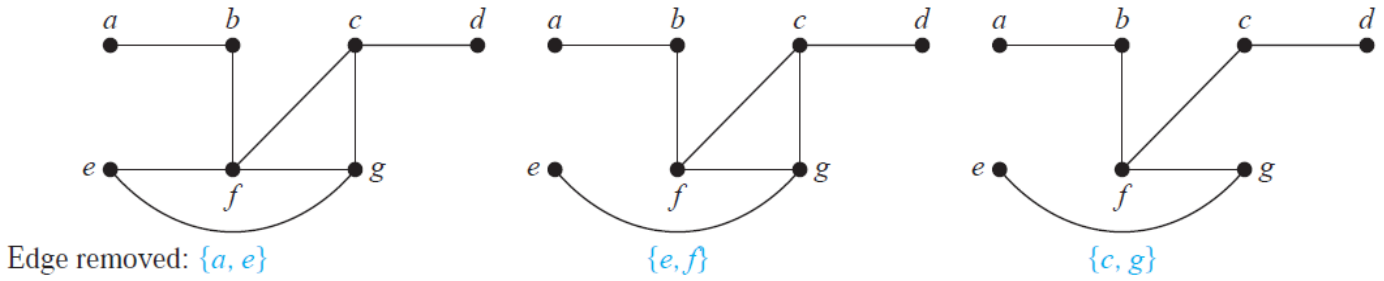
Spanning trees

Let G be a simple graph. A *spanning tree* of G is a subgraph of G that is a tree containing every vertex of G.

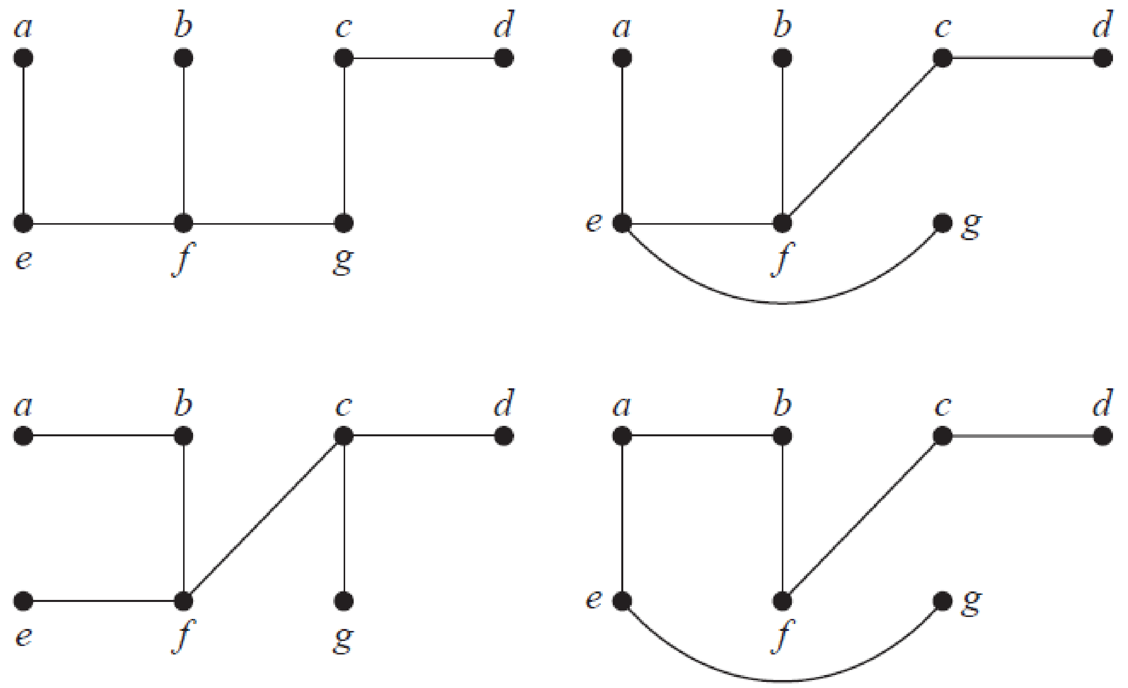
Find the spanning graph for the following simple graph



Solution: The graph G is connected, but it is not a tree because it contains simple circuits. Remove the edge $\{a, e\}$. This eliminates one simple circuit, and the resulting subgraph is still connected and still contains every vertex of G. Next remove the edge $\{e, f\}$ to eliminate a second simple circuit. Finally, remove edge $\{c, g\}$ to produce a simple graph with no simple circuits. This subgraph is a spanning tree, because it is a tree that contains every vertex of G.



other spanning graphs for the graph G are



Property of a Spanning Tree

A simple Graph G has a spanning tree if and only if G is connected.

Algorithms used to construct spanning trees

1.BFS algorithm

2.DFS algorithm

1.BFS algorithm

In this algorithm, a rooted tree is constructed and the underlying undirected graph of this rooted tree forms the spanning tree.

The idea of BFS algorithm is to visit all vertices on a given level before going to the next level.

Step 1: Choose a vertex arbitrarily and designate it as the root.

Step 2: Add all the edges incident to this vertex such that the addition of edges does not produce any cycle.

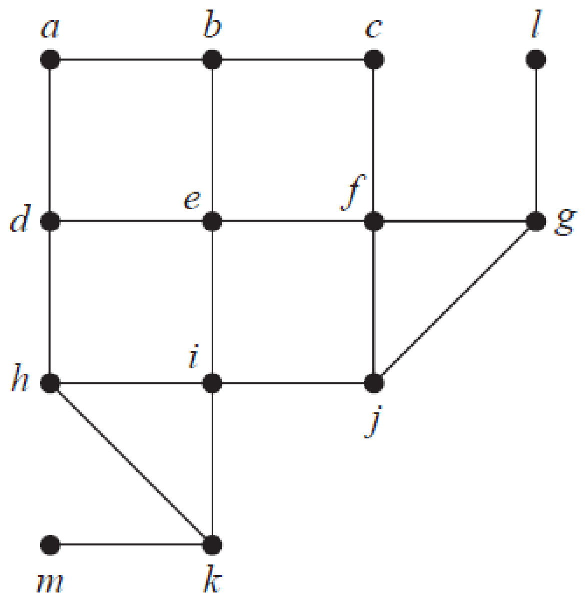
The new vertices added at this stage become the vertices at level 1 in the spanning tree .Order them in an arbitrary manner.

Step3: For each vertex at level 1 ,visited in order ,add each edge incident to this vertex to the tree as long as it does not produce any cycle.

Step4: Arbitrarily order the children of each vertex at level 1. This produces the vertices at level 2 in the tree.

Step5: Proceeding like this till all the vertices in the tree have been added ,we obtain the spanning tree .

Find the spanning tree of the following graph using BFS algorithm

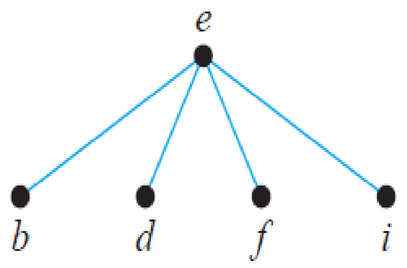


Find the spanning tree of the following graph using BFS algorithm

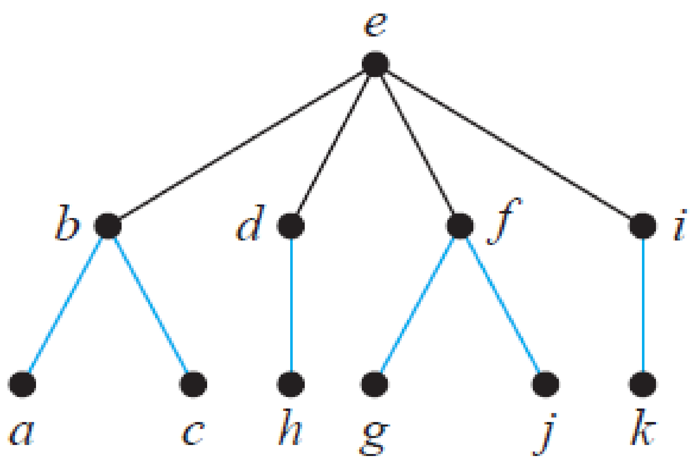
Solution:

Step1: Choose the vertex ‘e’ and designate it as root of the tree.

Step2: Find the edges incident with ‘e’, So edges from e to b,d,f,i are added. These 4 vertices form level 1 of the tree.

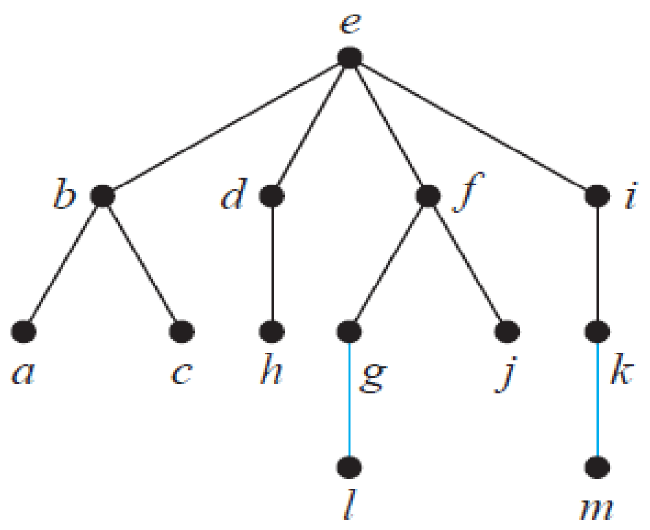


Step3: Now add the edges from level 1 vertices not already in the tree. Hence edges from b to a and c, also from d to h ,from f to j ,g ,from i to k are added. These new vertices are level 2 vertices.



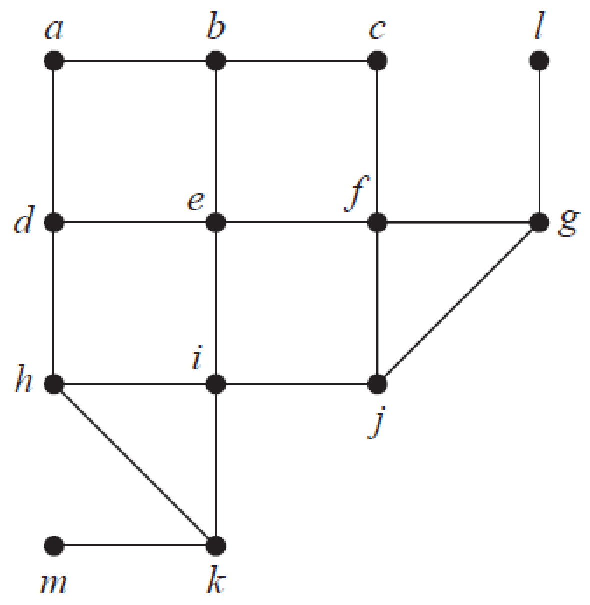
Step4: Now add the edges from level 2 vertices not already in the tree.

Hence edges from g to l and k to m are added.

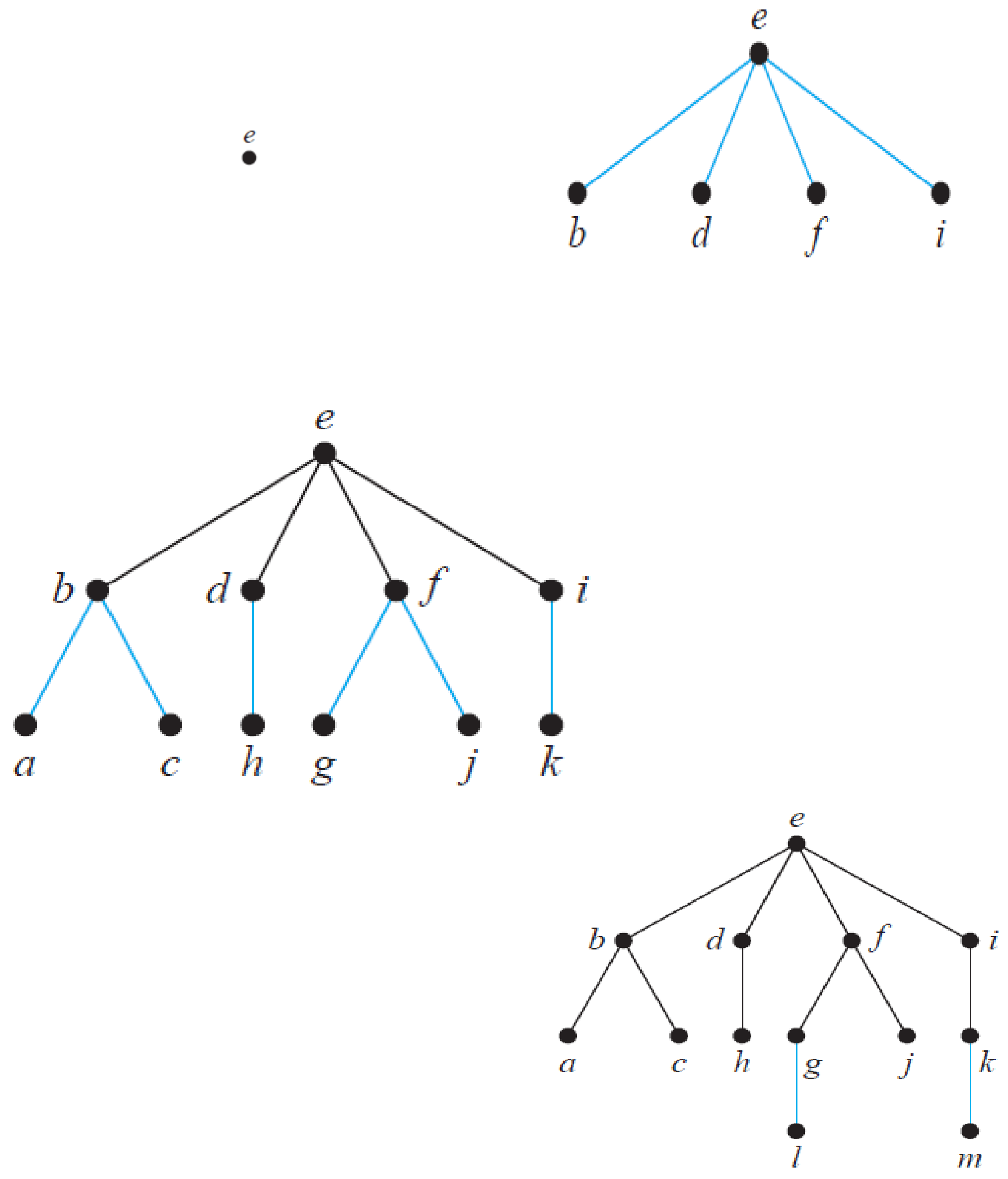


Hence the spanning tree of the graph G is obtained.

Given graph G



Construction of Tree



2.DFS algorithm

In this algorithm, a rooted tree is constructed and the underlying undirected graph of this rooted tree forms the spanning tree.

The idea of DFS algorithm is to visit the vertices on successive levels at the earliest possible opportunity. DFS algorithm is also called as backtracking.

Step 1: Choose a vertex arbitrarily and designate it as the root.

Step 2: Form a path starting with this vertex by adding edges as long as possible where each new edge is incident with the last vertex in the path without producing any cycle.

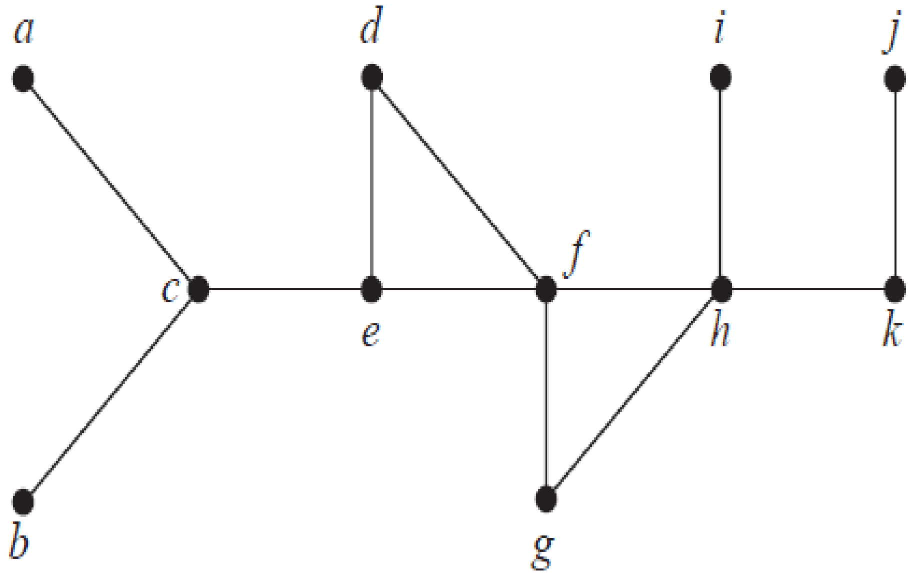
Step3: If the path goes through all the vertices of the graph ,then the tree consisting of this path is a spanning tree. Otherwise go to next step.

Step 4: Move back one vertex in the path and form a new path with the vertices not already visited. If not possible go to next step.

Step5: Move back two vertices in the path and form a new path covering the vertices not already visited .

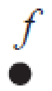
Proceeding like this, moving backwards one vertex at a time and adding the edges , the spanning tree is formed.

Find the spanning tree of the following graph using DFS algorithm

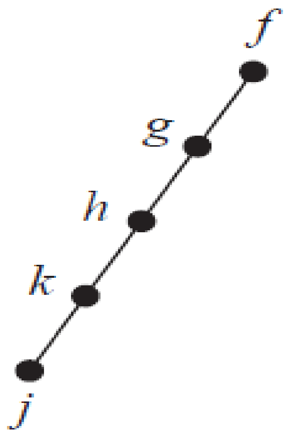


Solution:

Step1: Choose the vertex ‘f’ and designate it as root of the tree.

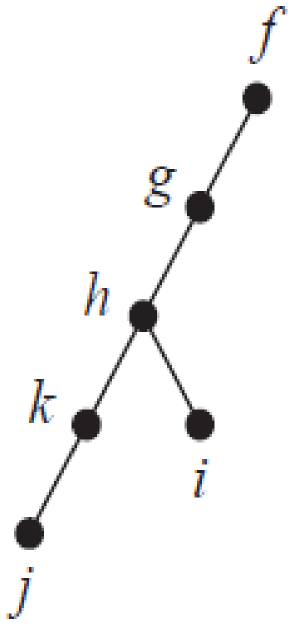


Step2: Find the vertex incident with ‘f’, continue to add vertices that are adjacent to the last vertex in the path. So g, h, k, j are added.



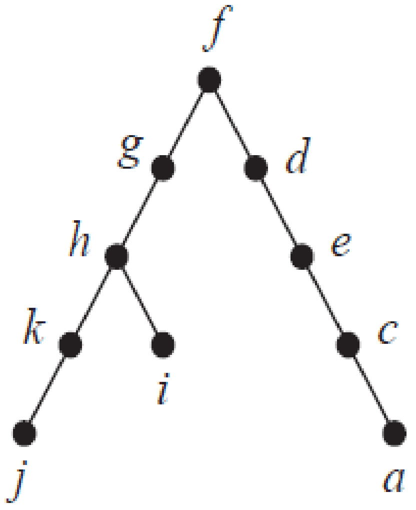
Step3: Now backtrack ‘k’ which is not possible.

Step4: backtrack ‘h’ and add vertices and edges that are connected with ‘h’



Step5: Now backtrack ‘g’ which is not possible.

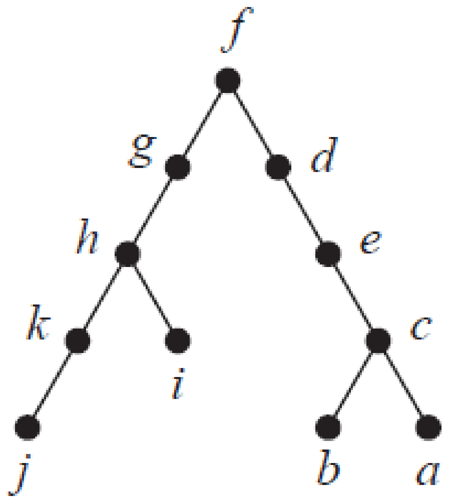
Step 6: Now backtracking ‘f’ is only possible . Add the edges and vertices d,e,c,a.



Step7: backtrack d, e, c, are not possible.

Step 8: Now backtracking ‘c’ is only possible . Add the edges and vertex

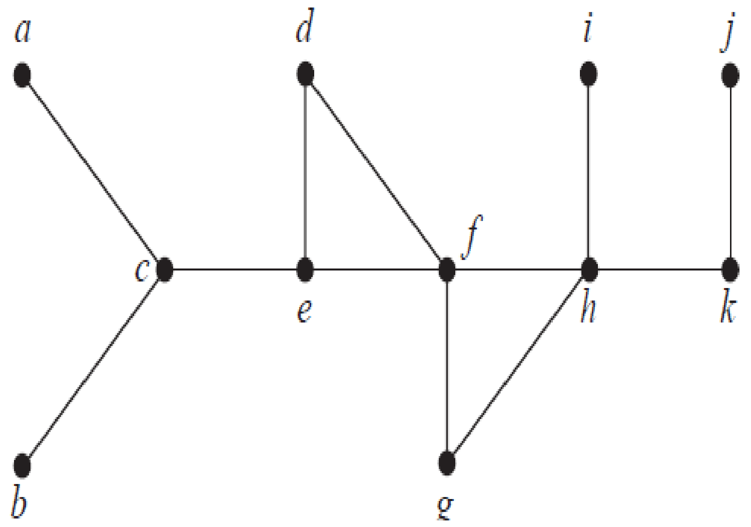
b. All the vertices in the graph are added in the tree.

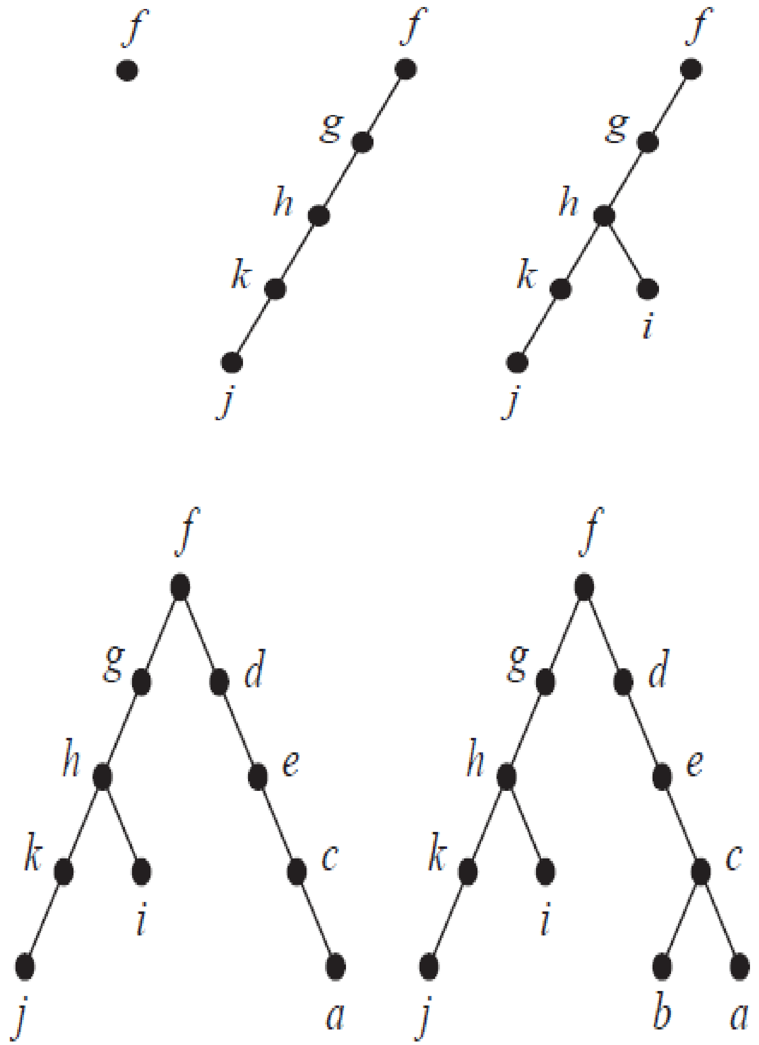


Hence the spanning tree of the graph G is obtained.

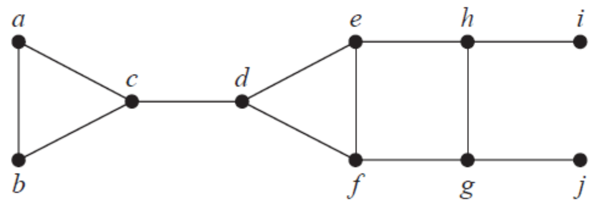
Construction of Tree

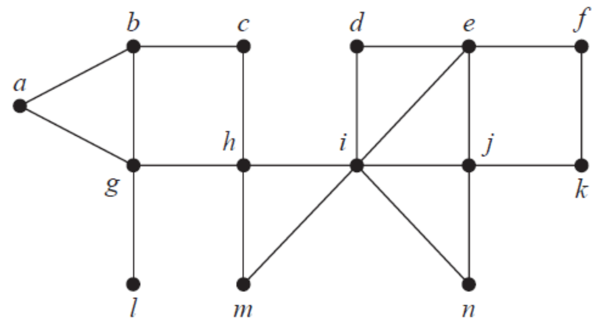
Given graph G

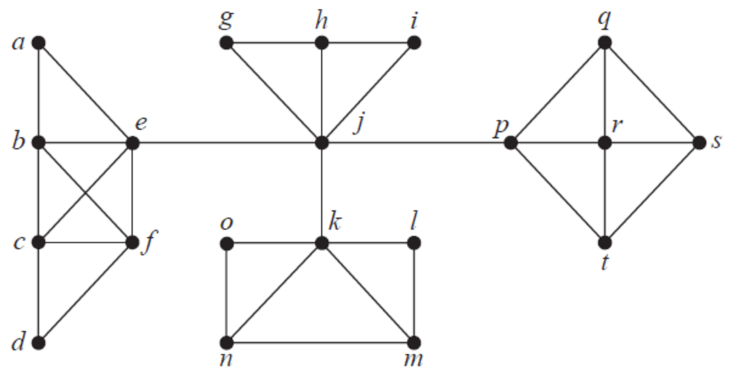


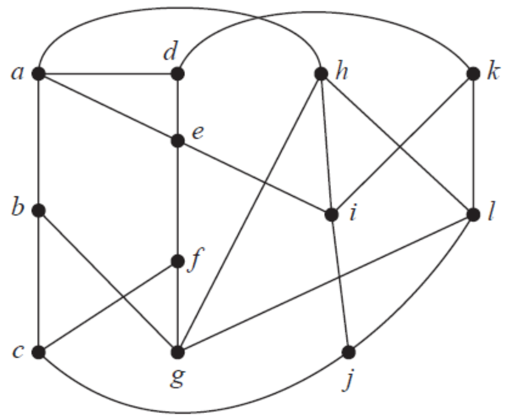


PRACTICE PROBLEMS

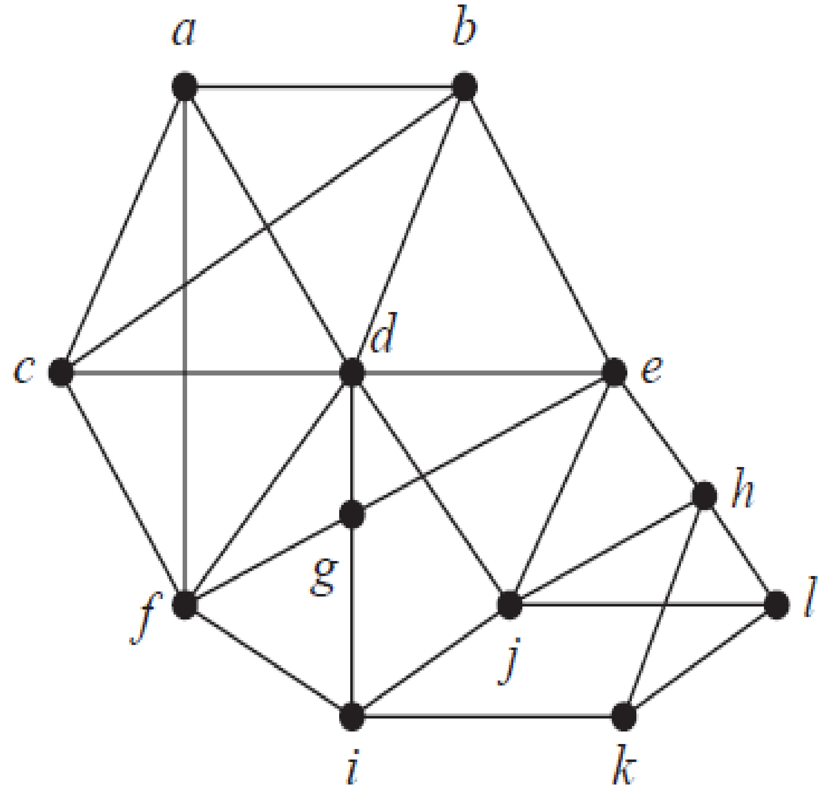




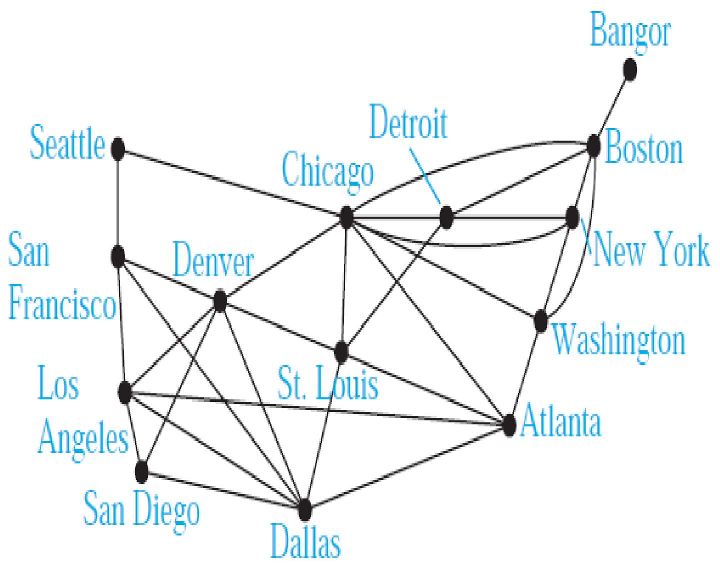




PRACTICE PROBLEMS

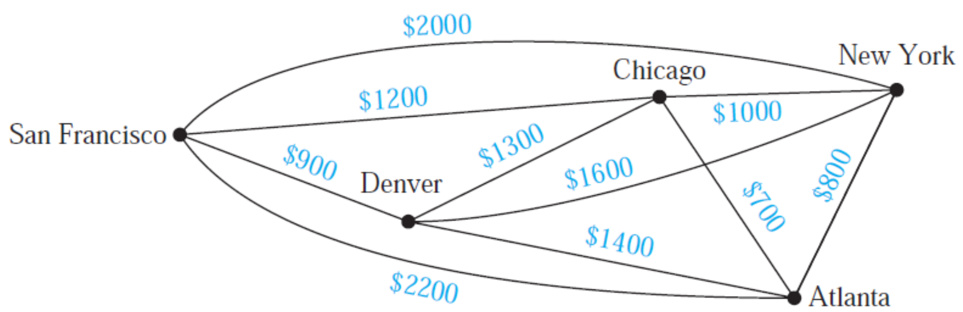


Suppose that an airline must reduce its flight schedule to save money. If its original routes are as illustrated here, which flights can be discontinued to retain service between all pairs of cities (where it may be necessary to combine flights to fly from one city to another)?



Minimal Spanning trees

Weighted Graph:
Graphs assigned with number for each edge is called a weighted graph.



A *minimum spanning tree* in a connected weighted graph is a spanning tree that has the smallest possible sum of weights of its edges.

Kruskal’s algorithm

In this algorithm, a minimal spanning tree is constructed.

Step 1: Arrange all the edges in the ascending order of their weights.

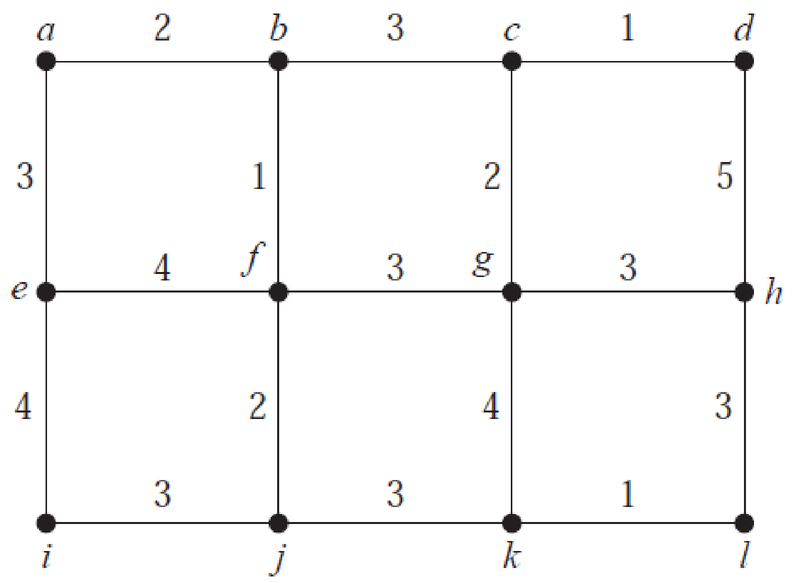
Step 2: Choose an edge of minimal weight. If there are more than one edge of minimal weight then choose arbitrarily one of these edges.

Step 3: At each stage, choose from the edges not yet chosen and *Check if it forms a cycle with the spanning tree formed so far. If cycle is not formed, include this edge. Else, discard it. .*

Step 4: If G has n vertices then stop after n-1 steps. Otherwise proceed with step2.

Kruskal's algorithm

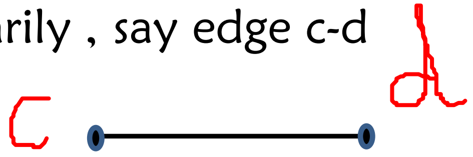
Find the minimal spanning tree for the following weighted graph.



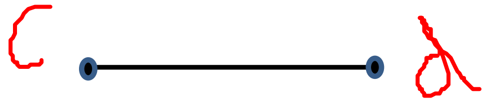
Choice	Edge	Weight
1	$\{c, d\}$	1
2	$\{k, l\}$	1
3	$\{b, f\}$	1
4	$\{c, g\}$	2
5	$\{a, b\}$	2
6	$\{f, j\}$	2
7	$\{b, c\}$	3
8	$\{j, k\}$	3
9	$\{g, h\}$	3
10	$\{i, j\}$	3
11	$\{a, e\}$	3
	Total:	24

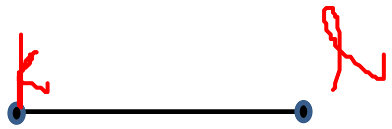
No.of vertices in G= 12. Maximum no. of steps=11

Step 1: Start with least weighted edge. There are 3 edges with weight 1.Choose arbitrarily , say edge c-d



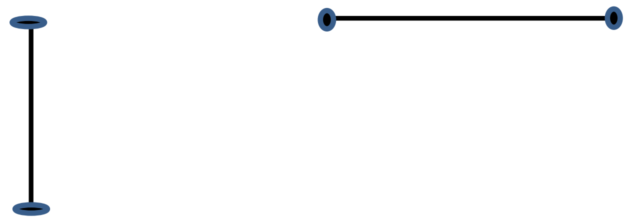
Step2: choose the next minimal edge k-l .





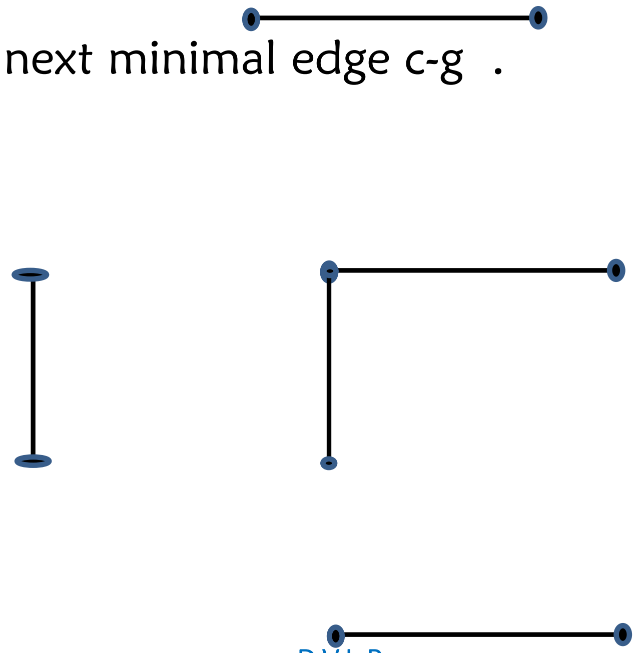
Choice	Edge	Weight
1	$\{c, d\}$	1
2	$\{k, l\}$	1
3	$\{b, f\}$	1
4	$\{c, g\}$	2
5	$\{a, b\}$	2
6	$\{f, j\}$	2
7	$\{b, c\}$	3
8	$\{j, k\}$	3
9	$\{g, h\}$	3
10	$\{i, j\}$	3
11	$\{a, e\}$	3
	Total:	24

Step 3: Add next least weighted edge b-f.

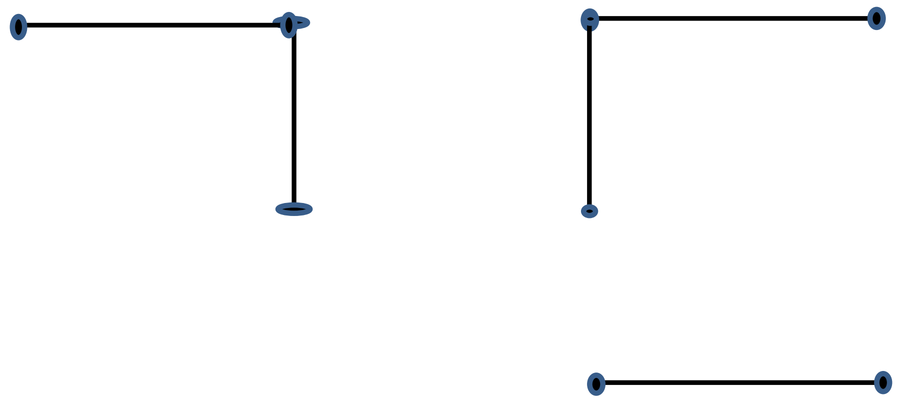


Choice	Edge	Weight
1	$\{c,\ d\}$	1
2	$\{k,\ l\}$	1
3	$\{b,\ f\}$	1
4	$\{c,\ g\}$	2
5	$\{a,\ b\}$	2
6	$\{f,\ j\}$	2
7	$\{b,\ c\}$	3
8	$\{j,\ k\}$	3
9	$\{g,\ h\}$	3
10	$\{i,\ j\}$	3
11	$\{a,\ e\}$	3
	Total:	24

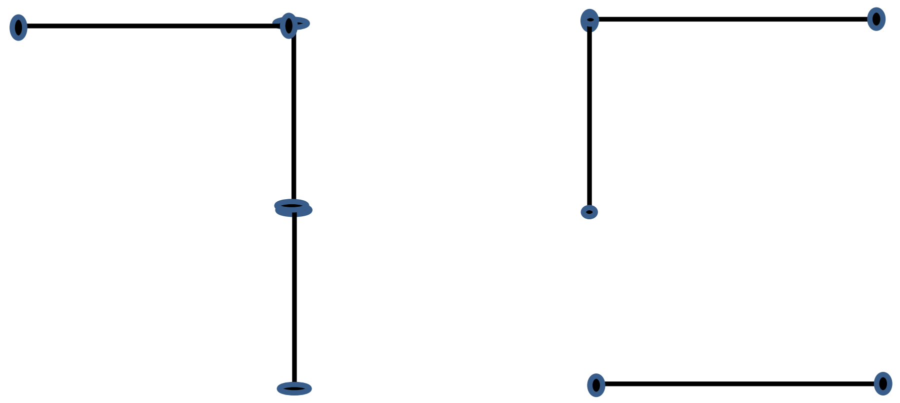
Step4: choose the next minimal edge c-g .



Step 5: Add next least weighted edge b-f.

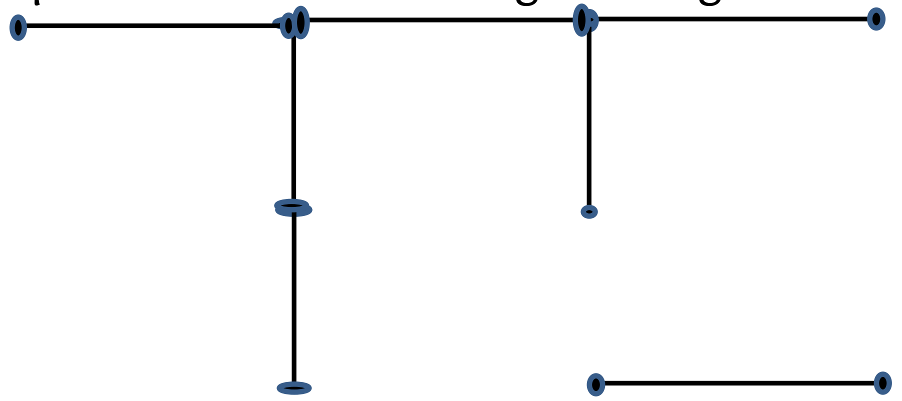


Step6: choose the next minimal edge f-j .

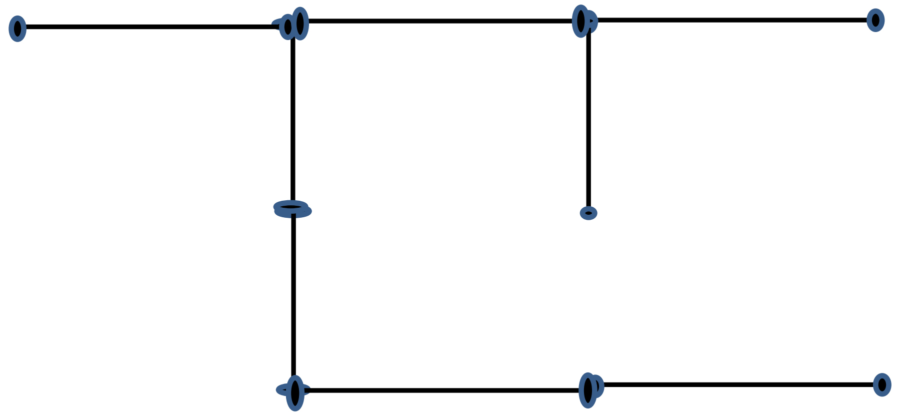


Choice	Edge	Weight
1	$\{c, d\}$	1
2	$\{k, l\}$	1
3	$\{b, f\}$	1
4	$\{c, g\}$	2
5	$\{a, b\}$	2
6	$\{f, j\}$	2
7	$\{b, c\}$	3
8	$\{j, k\}$	3
9	$\{g, h\}$	3
10	$\{i, j\}$	3
11	$\{a, e\}$	3
	Total:	24

Step 7: Add next least weighted edge b-c.



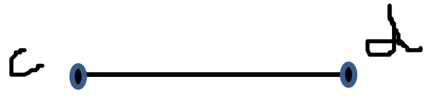
Step8: choose the next minimal edge j-k .



Choice	Edge	Weight
1	$\{c, d\}$	1
2	$\{k, l\}$	1
3	$\{b, f\}$	1
4	$\{c, g\}$	2
5	$\{a, b\}$	2
6	$\{f, j\}$	2
7	$\{b, c\}$	3
8	$\{j, k\}$	3
9	$\{g, h\}$	3
10	$\{i, j\}$	3
11	$\{a, e\}$	3
	Total:	24

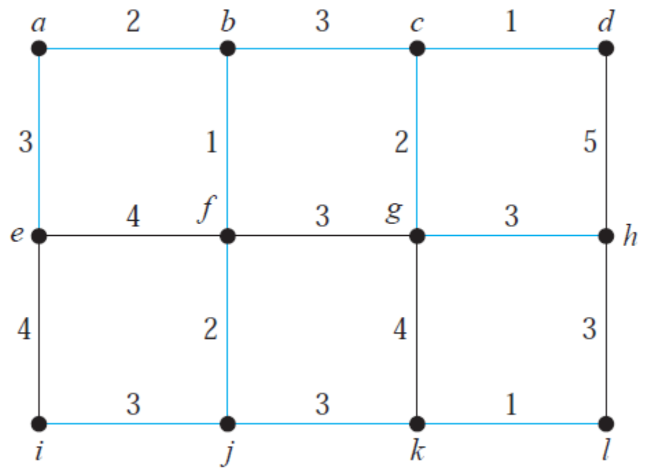
No.of vertices in G= 12.Maximum no.of steps=11

Step 1: Start with least weighted edge. There are 3 edges with weight 1.Choose arbitrarily , say edge c-d



Choice	Edge	Weight
1	$\{c, d\}$	1
2	$\{k, l\}$	1
3	$\{b, f\}$	1
4	$\{c, g\}$	2
5	$\{a, b\}$	2
6	$\{f, j\}$	2
7	$\{b, c\}$	3
8	$\{j, k\}$	3
9	$\{g, h\}$	3
10	$\{i, j\}$	3
11	$\{a, e\}$	3
	Total:	24

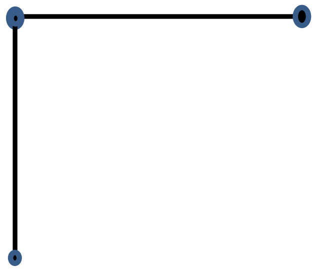
Step2: find edges incident with c. choose minimal edge

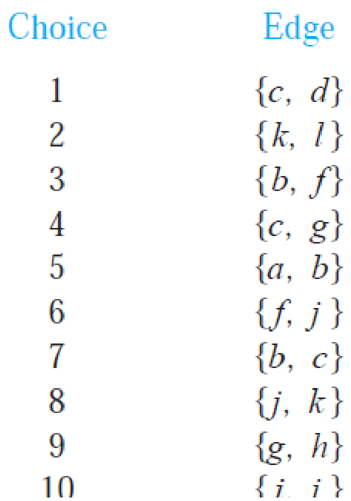


No.of vertices in G= 12.Maximum no.of steps=11

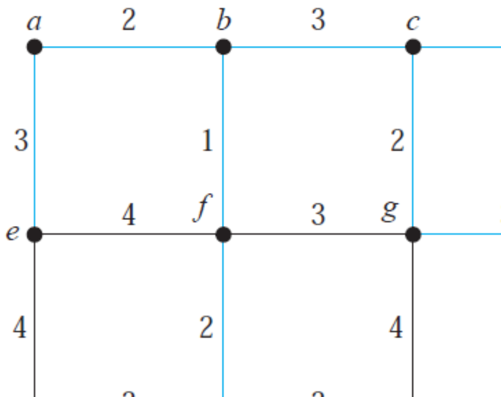
Step 1: Start with least weighted edge. There are 3 edges with weight 1.Choose arbitrarily , say edge c-d

Step2: find edges incident with c.choose minimal edge c-g.



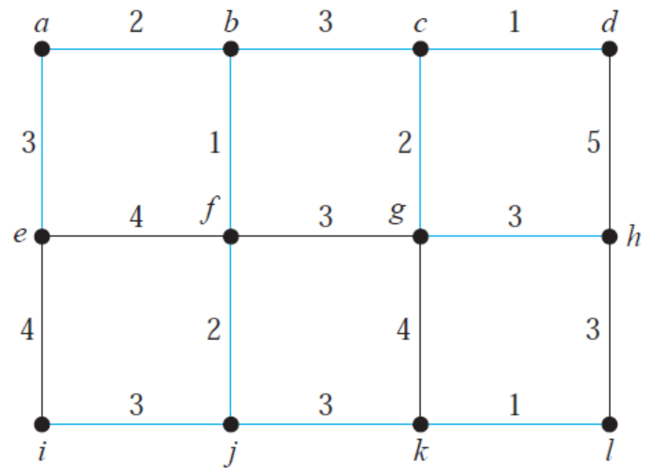


Choice	Edge
1	$\{c, d\}$
2	$\{k, l\}$
3	$\{b, f\}$
4	$\{c, g\}$
5	$\{a, b\}$
6	$\{f, j\}$
7	$\{b, c\}$
8	$\{j, k\}$
9	$\{g, h\}$
10	$\{i, j\}$

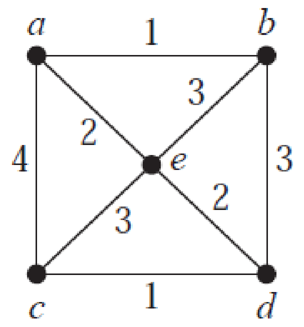


Kruskal's algorithm

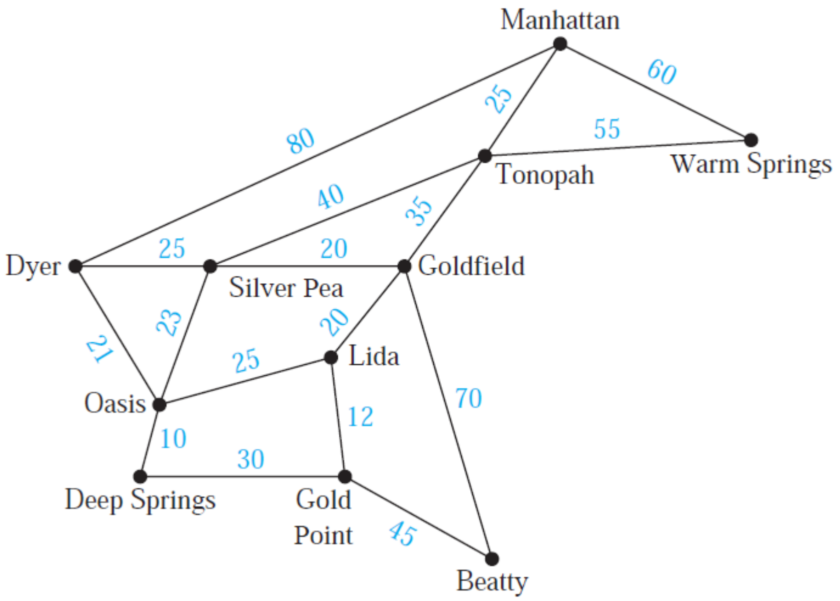
The minimal spanning tree is

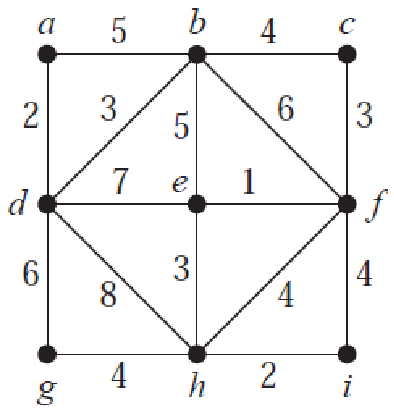


PRACTICE PROBLEMS



The roads represented by this graph are all unpaved. The lengths of the roads between pairs of towns are represented by edge weights. Which roads should be paved so that there is a path of paved roads between each pair of towns so that a minimum road length is paved?





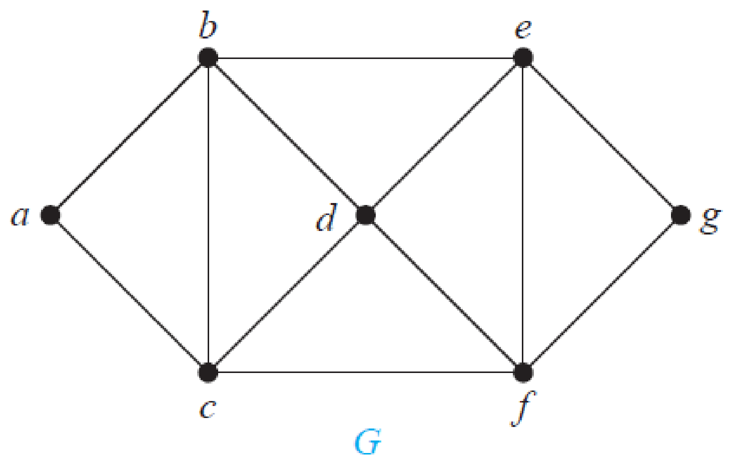
Graph Colouring, Chromatic Number

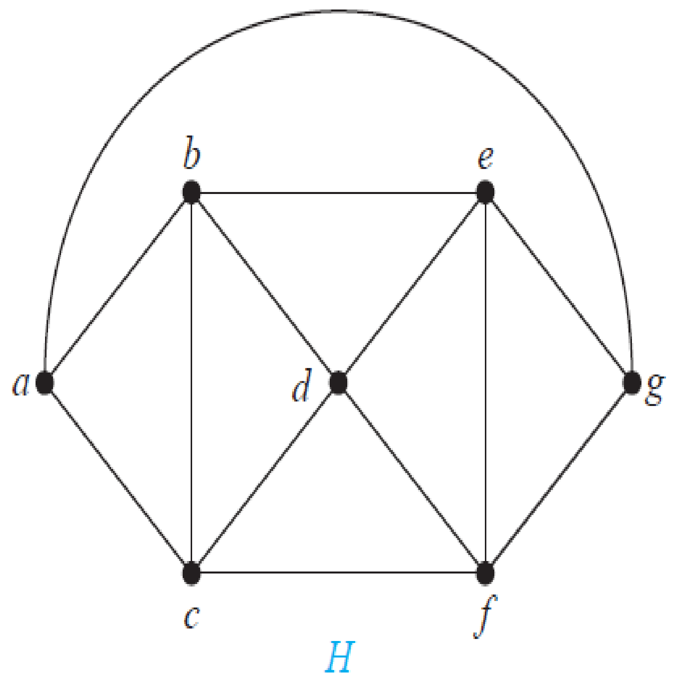
A *coloring* of a simple graph is the assignment of a color to each vertex of the graph so that no two adjacent vertices are assigned the same color.

The *chromatic number* of a graph is the least number of colors needed for a coloring of this graph. The chromatic number of a graph G is denoted by $\chi(G)$. (Here χ is the Greek letter *chi*.)

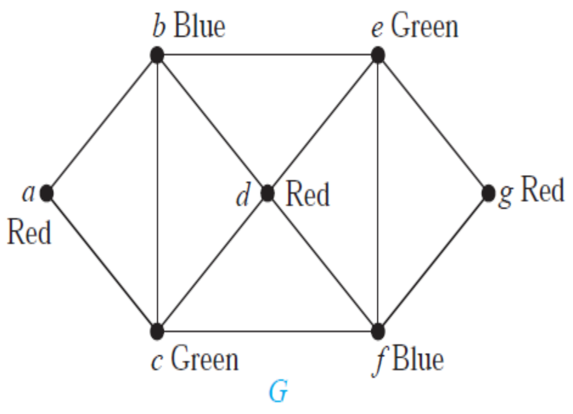
THE FOUR COLOR THEOREM The chromatic number of a planar graph is no greater than four.

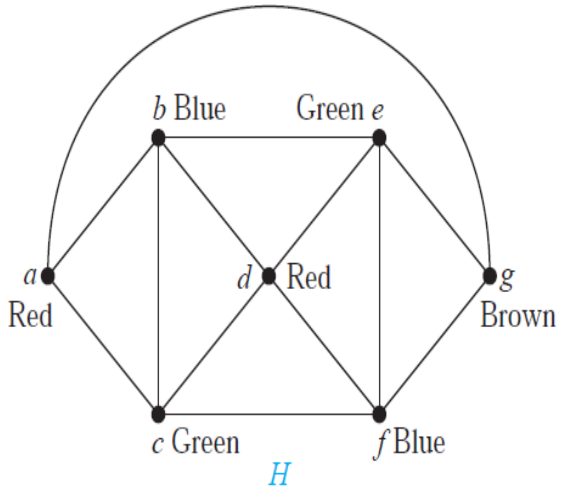
What are the chromatic numbers of the graphs G and H





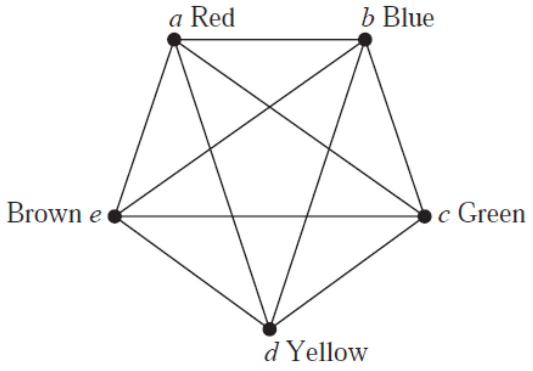
Solution:
No.of colours required for G =3
No.of colours required for H=4





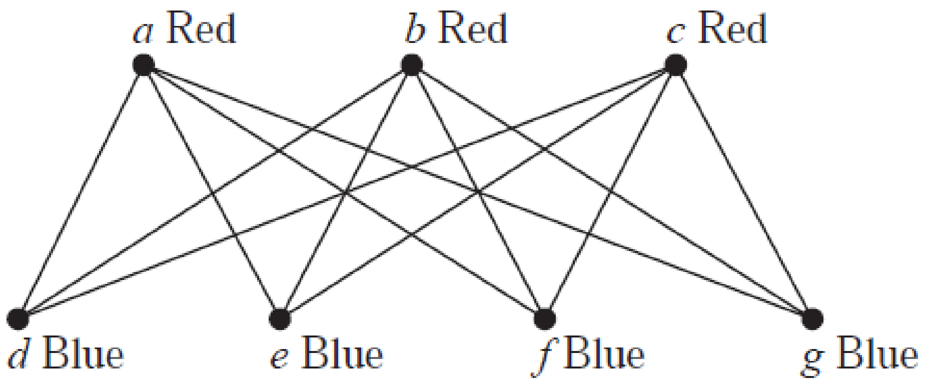
What is the chromatic number of K_n?

Solution: A coloring of K_n can be constructed using n colors by assigning a different color to each vertex. Is there a coloring using fewer colors? The answer is no. No two vertices can be assigned the same color, because every two vertices of this graph are adjacent. Hence, the chromatic number of K_n is n. That is, $\chi(K_n) = n$. (Recall that K_n is not planar when $n \geq 5$, so this result does not contradict the four color theorem.) A coloring of K_5 using five colors is

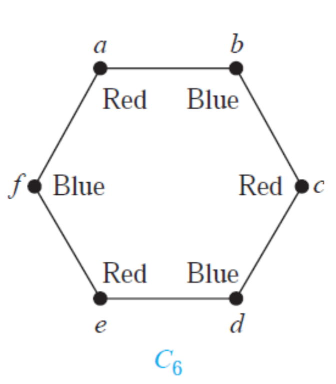


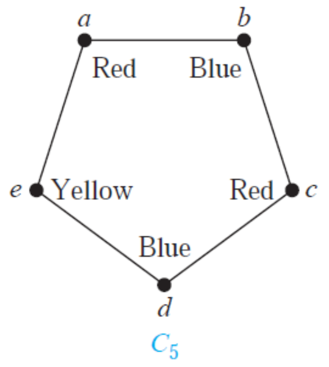
What is the chromatic number of the complete bipartite graph $K_{m,n}$, where m and n are positive integers?

$$\chi(K_{m,n}) = 2.$$



What is the chromatic number of the graph C_n, where $n \geq 3$? (Recall that C_n is the cycle with n vertices.)



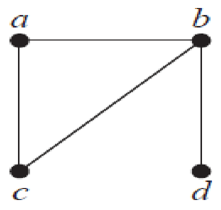


We have shown that $\chi(C_n) = 2$ if n is an even positive integer with $n \geq 4$ and $\chi(C_n) = 3$ if n is an odd positive integer with $n \geq 3$.

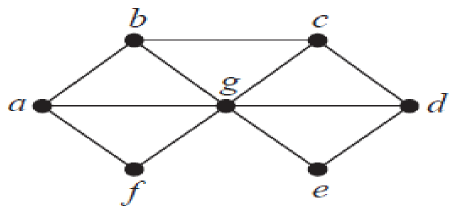
PRACTICE PROBLEMS

Find the chromatic number of the following graphs.

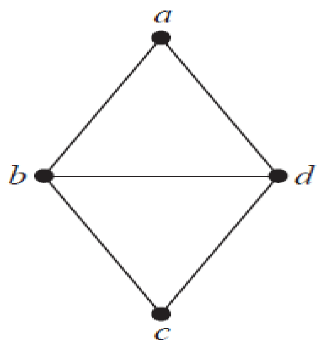
5.



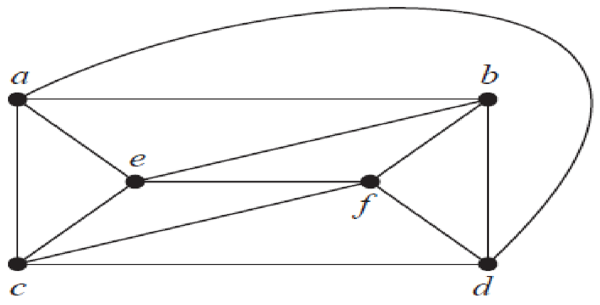
6.



7.



8.



9.