CHAPTER 5: Graphs and Trees

Students usually like to work with graphs because "it's fun to draw pictures." While the sheer volume of graph terminology at the beginning of the chapter is a little overwhelming, Section 5.1 on the whole goes rather easily. Graphs provide a nice way to introduce the concept of isomorphism, again because it is easy to "see" the correspondences, and check whether adjacency of nodes is preserved. This gentle beginning paves the way for later discussions on isomorphic Boolean algebras (Chapter 7) and isomorphic groups (Chapter 8). Graph planarity is an easily-understood idea that students can play with for a bit, and then Euler's formula gives some theoretical underpinnings; also the proof of Euler's formula provides a nice example of induction. The topic of graph coloring is treated in a series of exercises (68-76) at the end of Section 5.1.

It is worth pointing out that although a computer can store a picture of a graph as an image file, this is not a form that allows us to work with the graph data (note that the formal definition of a graph is not even given in visual terms). What the computer representation must capture is enough information to fulfill the definition of the graph, which would allow a visual representation to be reconstructed if desired. Adjacency matrix and linked-list graph representations are discussed in Section 5.1.

Trees and their representations are introduced in Section 5.2. Simple tree traversal algorithms - inorder, preorder, postorder - provide witness to the power of recursion, and results about trees provide still more practice with inductive proofs.

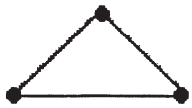
Applications of trees in Sections 5.3 and 5.4 include decision trees that give lower bounds on the work required for searching and sorting, binary search trees, and Huffman encoding. These topics are certainly optional, especially as the students may see them later in a data structures course.

EXERCISES 5.1

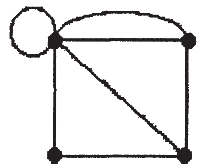
1. g(a) = (1, 2)
   g(b) = (1, 3)
   g(c) = (2, 3)
   g(d) = (2, 2)

2. a. yes b. no c. yes d. 3, a_5, 5, a_6, 6; 3, a_3, 4, a_4, 5, a_6, 6
   e. 3, a_3, 4, a_4, 5, a_5, 3 f. a_3, a_4, or a_5 g. a_1, a_2, a_6, or a_7

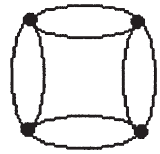
*3. a.



b. For example,

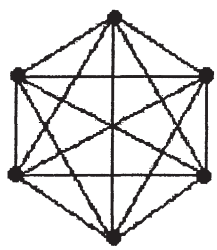


c.

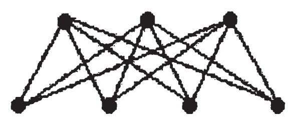


*4. a. 4, 5, 6
b. length 2
c. for example (naming the nodes), 1-2-1-2-2-1-4-5-6

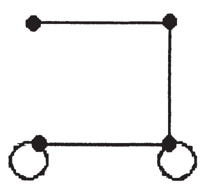
5. a.



b.

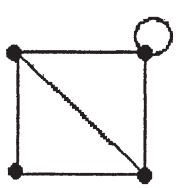


6. a. For example



b. Does not exist; the node of degree 4 would have to have arcs going to 4 distinct other nodes, since no loops or parallel arcs are allowed, but there are not 4 distinct other nodes.

c.



d. Does not exist; in such a graph, the sum of all the degrees would be 11, but the sum of all the degrees is the total number of arc ends, which must be twice the number of arcs, i.e., an even number.

*7. (b), because there is no node of degree 0.

8. (d), because the two nodes of degree 3 are not adjacent.

*9. f_1: $1 \to a$
$2 \to b$
$3 \to c$
$4 \to d$

f_2: $a_1 \to e_2$
$a_2 \to e_7$
$a_3 \to e_6$
$a_4 \to e_1$
$a_5 \to e_3$
$a_6 \to e_4$
$a_7 \to e_5$

10. Not isomorphic; graph in (b) has a loop, graph in (a) does not.

11. f: $1 \to a$
$2 \to d$
$3 \to b$
$4 \to e$
$5 \to c$

12. f: $1 \to b$
$2 \to d$
$3 \to c$
$4 \to e$
$5 \to f$
$6 \to a$

*13. Not isomorphic; graph in (b) has a node of degree 5, graph in (a) does not.

14. f: $1 \to a$
$2 \to c$
$3 \to e$
$4 \to g$
$5 \to f$
$6 \to h$
$7 \to b$
$8 \to d$

15. a. There cannot be a bijection between the two node sets if they are not the same size.

b. For isomorphic graphs there is a bijection from one arc set to the other, either explicitly or, in the case of simple graphs, implicitly by means of the endpoints; this cannot happen if the arc sets are not the same size.

c. If arcs a_1 and a_2 in one graph both have endpoints x-y, then their image arcs in the second graph must have the same endpoints, which cannot happen if the second graph has no parallel arcs.

d. If an arc in one graph has endpoints x-x, then its image arc in the second graph must have endpoints f(x)-f(x), which is not possible if the second graph has no loops.

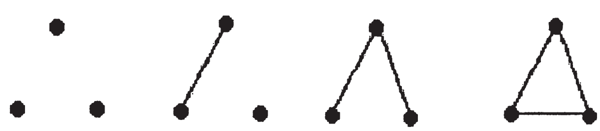
e. A node of degree k in one graph serves as an endpoint to k arcs; its image in the second graph must serve as an endpoint to the images of those k arcs, which implies it will have degree k also.

f. If there is a path $n_1, a_1, n_2, a_2, \ldots, n_k$ between two nodes in one graph, then $f(n_1), f(a_1), f(n_2), f(a_2), \ldots, f(n_k)$ is a path in the second graph. Two nodes in the second graph are the images of nodes in the first graph; if the first graph is connected, there is a path between these nodes and hence there is a path between the two nodes in the second graph.

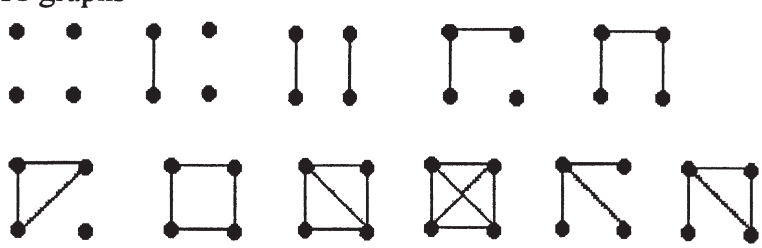
g. By the answer to part f, paths map to paths, so cycles map to cycles.

*16. 2 graphs

17. 4 graphs



18. 11 graphs

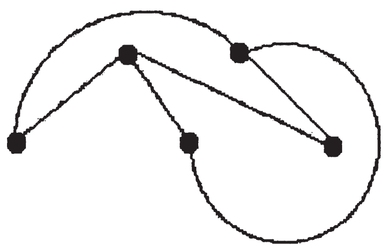


19. $\frac{n(n-1)}{2}$

The number of arcs is the number of ways to select 2 nodes out of n, C(n, 2) = $\frac{n(n-1)}{2}$. (Other proof methods include induction on the number of nodes.)

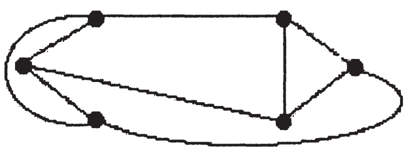
20. n = 13, a = 19, r = 8 and n - a + r = 2

*21.



$K_{2,3}$

22.



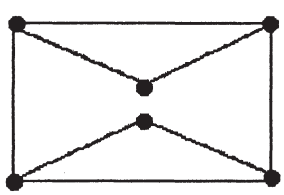
*23. 5 (by Euler's formula)

24. 8 (by Euler's formula)

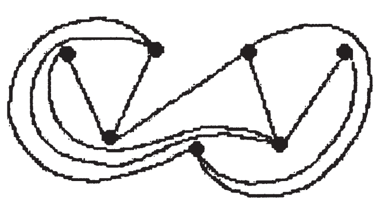
25. The proof for Euler's formula does not depend on the graph being simple, so the result still holds for nonsimple graphs, but this is not true for inequalities (2) and (3).

26. In elementary subdivisions, the inserted node must be a new node, so once v is made a node from the first subdivision, it cannot be a node in the second subdivision.

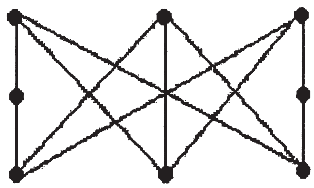
*27. Planar



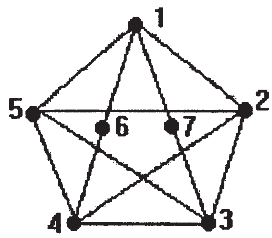
28. Planar



*29. Nonplanar - subgraph below can be obtained from $K_{3,3}$ by elementary subdivisions.



30. Nonplanar - the representation of the original graph shown below indicates that it can be obtained from K_5 by elementary subdivisions.



*31. $\begin{bmatrix} 1 & 1 & 0 & 0 & 2 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 2 & 1 & 0 & 0 & 0 \end{bmatrix}$

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

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

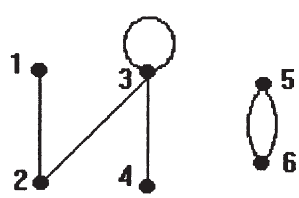
34. $\begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 \end{bmatrix}$

*35. $\begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$

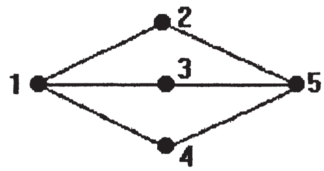
36. $\begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$

37.

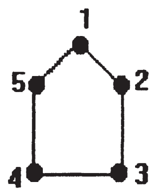
38.



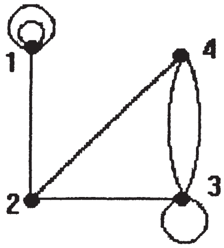
*39.



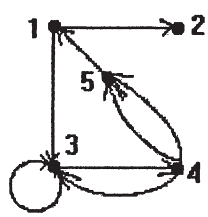
40.



*41.



42.

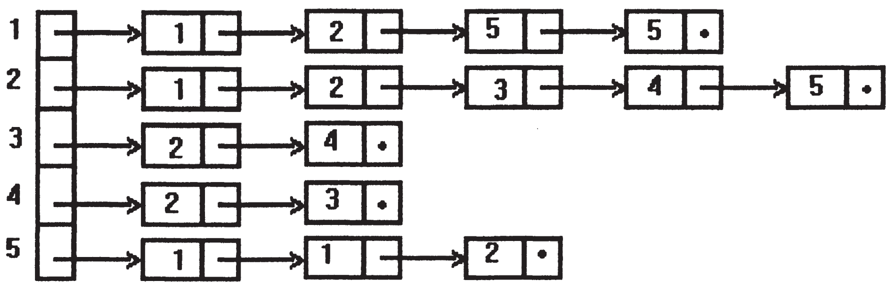


*43. The graph consists of n disconnected nodes with a loop at each node.

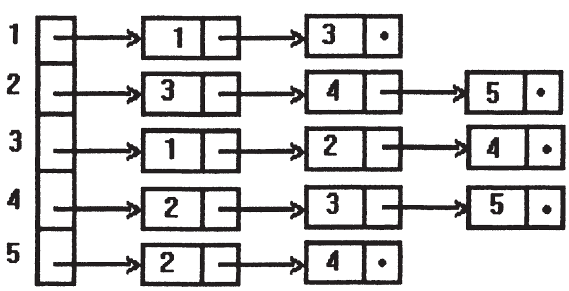
44. The $n \times n$ matrix with 0s down the main diagonal and 1s elsewhere.

45. The directed graph represented by A^T will look like the original graph with all the arrows reversed.

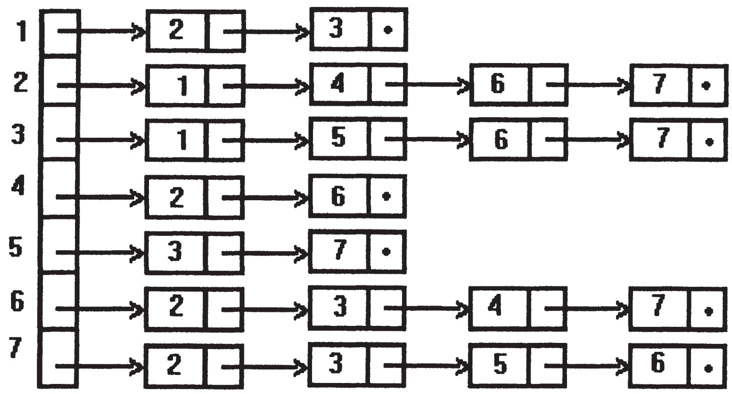
46.



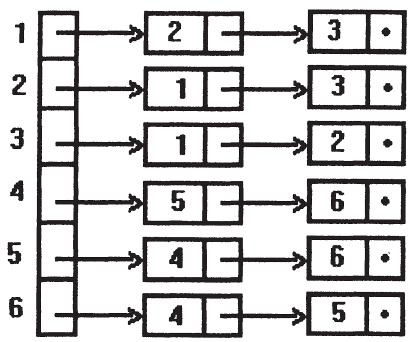
47.



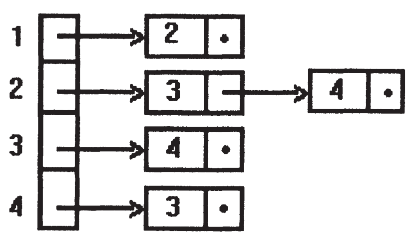
*48.



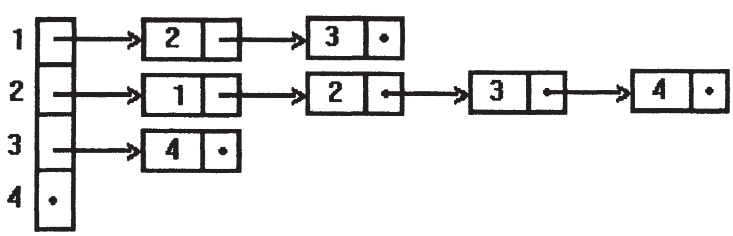
49.



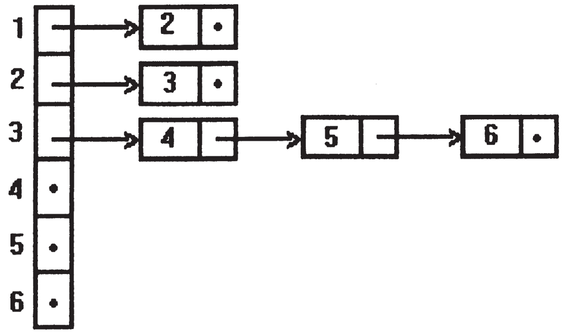
50.



51.

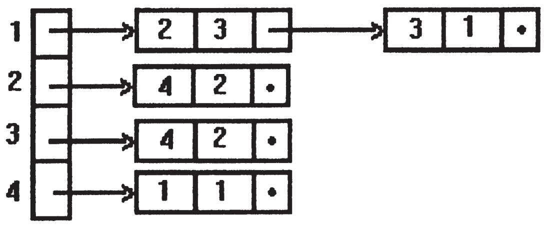


52. a.



b. 16 c. 36

*53.



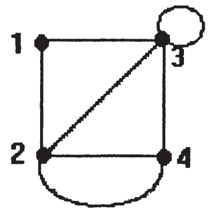
*54.

	Node	Pointer
1		5
2		7
3		11
4		0
5	2	6
6	3	0
7	1	8
8	2	9
9	3	10
10	4	0
11	4	0

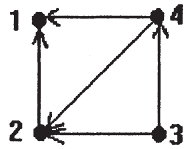
55.

	Node	Weight	Pointer
1			5
2			7
3			8
4			9
5	2	3	6
6	3	1	0
7	4	2	0
8	4	2	0
9	1	1	0

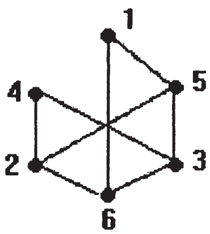
56.



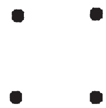
57.



*58.



59.

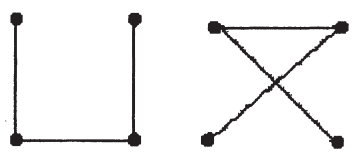


60. By the definition of isomorphic graphs, nodes x-y are adjacent in G_1 if and only if their images are adjacent in G_2. Therefore nodes are not adjacent in G_1 (and therefore are adjacent in G_1') if and only if their images are not adjacent in G_2 (and therefore are adjacent in G_2'). Thus the same function f makes the complement graphs isomorphic.

61. If G is isomorphic to G', then there is a bijection between the arcs of G and the arcs of G', so each has the same number of arcs. Thus considering all the arcs in the complete graph K_n, exactly half belong to G and half to G', so the number of arcs in K_n must be even, say 2m. There are n(n - 1)/2 arcs in K_n, so n(n - 1)/2 = 2m or n(n - 1) = 4m. In the product n(n - 1), one factor is even and the other odd, so the factor of 4 in the product must come either from n, in which case n = 4k for some k, or from n - 1, in which case n - 1 = 4k for some k, or n = 4k + 1.

*62. a. If G is not connected then G consists of two or more connected subgraphs that have no paths between them. Let x and y be distinct nodes. If x and y are in different subgraphs, there is no x-y arc in G, hence there is an x-y arc in G', and a path exists from x to y in G'. If x and y are in the same subgraph, then pick a node z in a different subgraph. There is an arc x-z in G' and an arc z-y in G', hence there is a path from x to y in G'.

b.



G G'

63. The matrix for G' will have 1s where A had 0s and 0s where A had 1s except for diagonal elements, which remain 0s.

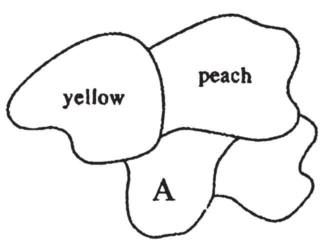
64. In a simple connected planar graph G with n nodes, $n \geq 3$, and a arcs, $a \leq 3n - 6$ (Theorem on the Number of Nodes and Arcs). The number of arcs in G' is the number of arcs in the complete graph, $n(n - 1)/2$, minus a. If G' is also planar, then $n(n - 1)/2 - a \leq 3n - 6$, or $n(n - 1)/2 \leq 3n - 6 + a \leq 3n - 6 + 3n - 6 = 6n - 12$. This says that $n(n - 1) \leq 12n - 24$, or $n^2 - 13n + 24 \leq 0$. But a proof by induction shows that for $n \geq 11$, $n^2 - 13n + 24 > 0$, so not both G and G' can be planar.

65. The maximum number of arcs occurs in a complete graph; the maximum is $C(n,2) = n(n - 1)/2$, therefore $a \leq n(n - 1)/2$ or $2a \leq n^2 - n$.

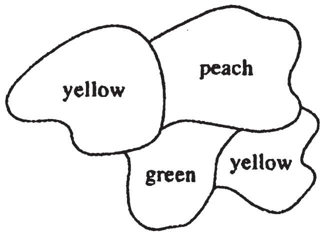
*66. Let a simple connected graph have n nodes and m arcs. The original statement is $m \geq n - 1$, which says $n \leq m + 1$, or that the number of nodes is at most $m + 1$. For the base case, let $m = 0$. The only simple connected graph with 0 arcs consists of a single node, and the number of nodes, 1, is $\leq 0 + 1$. Now assume that any simple connected graph with r arcs, $0 \leq r \leq k$, has at most $r + 1$ nodes. Consider a simple connected graph with $k + 1$ arcs, and remove one arc. If the remaining graph is connected, it has k arcs and, by the inductive hypothesis, the number n of nodes satisfies $n \leq k + 1$. Therefore in the original graph (with the same nodes), $n \leq k + 1 < (k + 1) + 1$. If the remaining graph is not connected, it consists of two connected subgraphs with r_1 and r_2 arcs, $r_1 \leq k$ and $r_2 \leq k$, $r_1 + r_2 = k$. By the inductive hypothesis, one subgraph has at most $r_1 + 1$ nodes and the other has at most $r_2 + 1$ nodes. The original graph therefore had at most $r_1 + 1 + r_2 + 1 = (k + 1) + 1$ nodes.

67. Let G be a simple graph with n nodes, $n \geq 2$, and m arcs, $m > C(n-1,2) = (n - 1)(n - 2)/2$, and suppose that G is not connected. By Exercise 62, G' is connected. By Exercise 66, the number of arcs in G' is at least $n - 1$. Therefore the number m of arcs in G is $n(n - 1)/2$ - (the number of arcs in G') $\leq n(n - 1)/2 - (n - 1) = (n - 1)(n/2 - 1) = (n - 1)(n - 2)/2$, which is a contradiction.

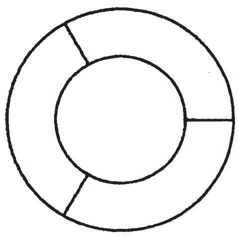
68. At least three colors are required because of the overlapping boundaries. Once the following assignment has been made, the country marked A must be a third color.



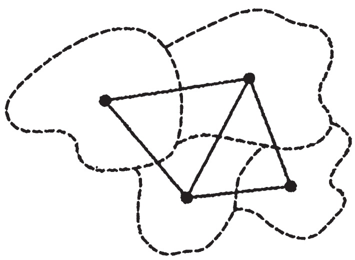
Three colors are sufficient:



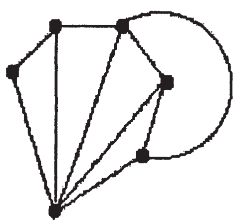
69. For example,



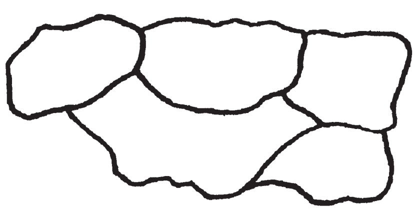
70. a.



b.



c.



71. *a. 3 b. 4

72. The four-color conjecture is equivalent to the statement that the chromatic number for any simple, connected, planar graph is at most 4.

*73. If this result is not true, then every node in such a graph has degree greater than 5, that is, degree 6 or higher. The total number of arc ends in the graph is then at least 6n, where n is the number of nodes. But the number of arc ends is exactly twice the number a of arcs, so

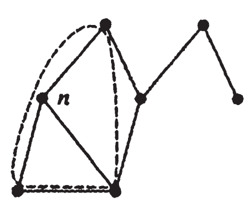
$$6n \leq 2a$$

By inequality (2) of the theorem on the number of nodes and arcs, $a \leq 3n - 6$ or $2a \leq 6n - 12$. Combining these inequalities, we obtain

$$6n \leq 6n - 12$$

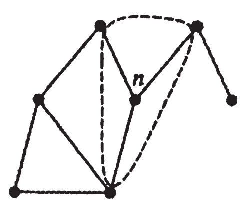
which is a contradiction.

74. The proof is by mathematical induction on the number of nodes in the graph. For the basis step of the induction process, it is clear that five colors are sufficient if the number of nodes is less than or equal to 5. Now assume that any simple, connected, planar graph with $\leq k$ nodes can be colored with five colors, and consider such a graph with $k + 1$ nodes. We can assume that $k + 1$ is at least 6 because 5 or fewer nodes are taken care of. By Exercise 73, at least one node n of the graph has degree less than or equal to 5; temporarily removing n (and its adjoining arcs) from the graph will leave a collection of one or more simple, connected, planar subgraphs, each with no more than k nodes (Figures a and b). By the inductive hypothesis, each subgraph has a coloring with no more than five colors (use the same palette of five colors for each subgraph). Now look at the original graph again. If n has degree less than 5 or if the 5 nodes adjacent to n do not use five different colors, there is a fifth color left to use for n. Thus, we assume that n is adjacent to 5 nodes, n_1, n_2, n_3, n_4, and n_5, arranged clockwise around n and colored, respectively, colors 1, 2, 3, 4, and 5 (Figure c).



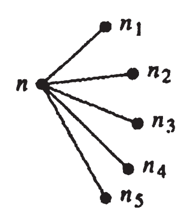
One subgraph remaining when *n* is removed

(a)



Two subgraphs remaining when *n* is removed

(b)



(c)

Now pick out all the nodes in the graph colored 1 or 3. Suppose there is no path, using just these nodes, between n_1 and n_3. Then, as far as nodes colored 1 and 3 are concerned, there are two separate sections of graph, one section containing n_1 and one containing n_3. In the section containing n_1 interchange colors 1 and 3 on all the nodes. Doing this does not violate the (proper) coloring of the subgraphs, it colors n_1 with 3, and it leaves color 1 for n.

Now suppose there is a path between n_1 and n_3 using only nodes colored 1 or 3. In this case we pick out all nodes in the original graph colored 2 or 4. Is there a path, using just these nodes, between n_2 and n_4? No, there is not. Because of the arrangement of nodes n_1, n_2, n_3, n_4, and n_5, such a path would have to cross the path connecting n_1 and n_3. Because the graph is planar, these two paths would have to meet at a node, which would then be colored 1 or 3 from the n_1-n_3 path and 2 or 4 from the n_2-n_4 path, an impossibility. Thus, there is no path using only nodes colored 2 or 4 between n2 and n4, and we can rearrange colors as in the previous case. This completes the proof.

75. *a. A graph with parallel arcs would contain

Temporarily create small countries at the nodes of parallel arcs, giving

which is a simple graph. Any coloring satisfactory for this graph is satisfactory for the original graph (let the new regions shrink to points).

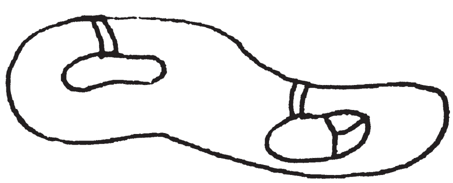
b. By Euler's formula, in such a graph $n - a + r = 2$, but r is the total number of regions, including the one external region, so $r = R + 1$, therefore $n - a + R = 1$.

c. The total number of region edges in the graph is 2a, which is greater than or equal to the number of edges of enclosed regions, which in turn is greater than or equal to 6R. Therefore $2a \geq 6R = 6(1 - n + a)$ (from part (b)) $= 6 - 6n + 6a$, from which $2a \leq 3n - 3$.

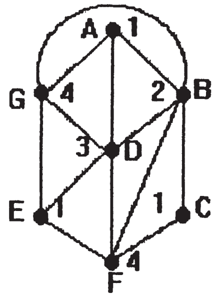
d. Assume that every enclosed region has at least six adjacent edges. The total number of arc ends in the graph $= 2a \geq 3n > 3n - 3 \geq 2a$ (by part (b)). Contradiction.

e. Use induction on the number of enclosed regions. Six colors are sufficient for any map with ≤ 6 enclosed regions. Assume 6 colors are sufficient for any map with k enclosed regions, and consider a map with $k + 1$ enclosed regions. By part (d), there is at least one enclosed region with ≤ 5 edges. Shrink this region to a point. The remaining regions have a proper coloring by the inductive hypothesis; expand the region back and use the sixth color for it.

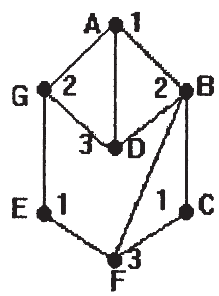
f. For countries with holes, put slits to open up the holes, making sure that a slit doesn't end at a boundary. Considering the slits as regions, the new map has no holes and can be colored by part (e). Then shrink the slits.



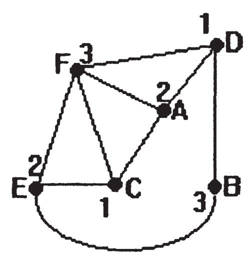
76. Let the members of Congress be the nodes of the graph, and include an arc x-y if any lobbyist must see both x and y. A time slot (instead of a color) is to be assigned to each node; adjacent nodes must have different time-slots. The resulting graph requires four time slots.



If lobbyist 3 does not need to see B and lobbyist 5 does not need to see D, the graph changes to one that requires only 3 time slots.

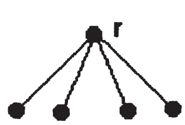


77. Let the processors be the nodes of a graph, and include an arc x-y if processors x and y both write to the data store at the same time. A data store block (instead of a color) is to be assigned to each node; adjacent nodes must use different blocks. The resulting graph requires three blocks.

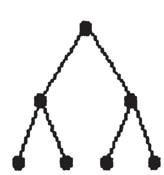


EXERCISES 5.2

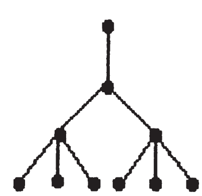
*1. a.



b.

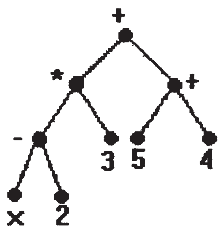


c.

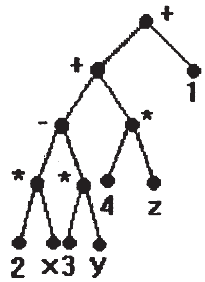


2. a. yes b. no c. yes d. b e. does not exist f. 2 g. 3

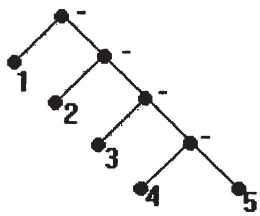
3.



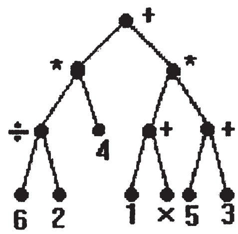
*4.



5.



6.



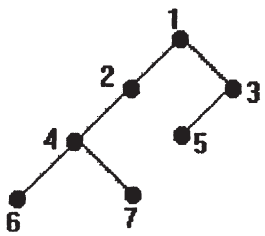
*7.

	Left child	Right child
1	2	3
2	0	4
3	5	6
4	7	0
5	0	0
6	0	0
7	0	0

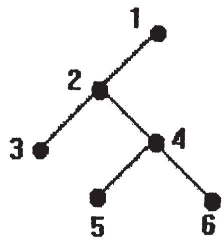
8.

	Left child	Right child
1	2	3
2	4	5
3	6	7
4	8	9
5	10	11
6	12	13
7	14	15
8	0	0
9	0	0
10	0	0
11	0	0
12	0	0
13	0	0
14	0	0
15	0	0

*9.



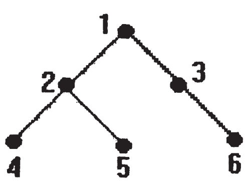
10.



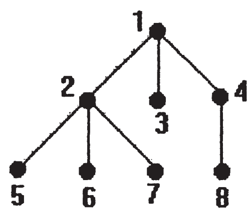
11.

	Name	Left child	Right child
1	All	0	2
2	Gaul	3	4
3	divided	0	0
4	is	5	6
5	into	0	0
6	three	7	0
7	parts	0	0

*12.



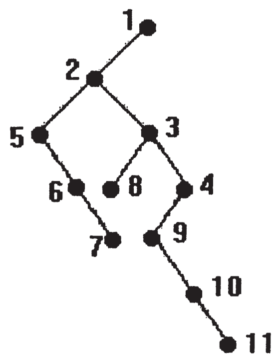
13.



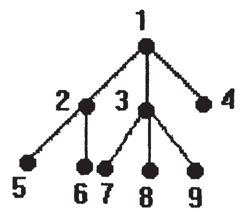
14. a.

	Left child	Right sibling
1	2	0
2	5	3
3	8	4
4	9	0
5	0	6
6	0	7
7	0	0
8	0	0
9	0	10
10	0	11
11	0	0

b.



15.



*16. preorder: a b d e h f c g
inorder: d b h e f a g c
postorder: d h e f b g c a

17. preorder: a b d g e c f h
inorder: g d b e a h f c
postorder: g d e b h f c a

18. preorder: a b e c f j g d h i
inorder: e b a j f c g h d i
postorder: e b j f g c h i d a

19. preorder: a b e f c g h d i
inorder: e b f a g c h i d
postorder: e f b g h c i d a

*20. preorder: a b c e f d g h
inorder: e c f b g d h a
postorder: e f c g h d b a

21. preorder: a b d h i c e f g
inorder: h d i b a e c f g
postorder: h i d b e f g c a

*22. prefix: + / 3 4 - 2 y
postfix: 3 4 / 2 y - +

23. prefix: * + * x y / 3 z 4
postfix: x y * 3 z / + 4 *

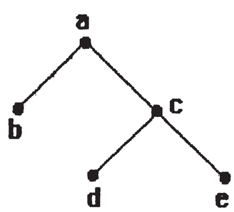
24. infix: ((2 + 3) * (6 * x)) - 7
postfix: 2 3 + 6 x * * 7 -

25. infix: ((x - y) + z) - w
postfix: x y - z + w -

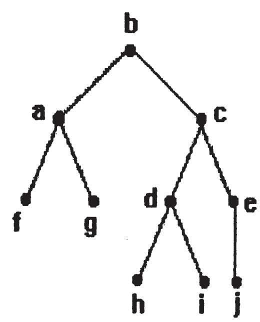
*26. prefix: + * 4 - 7 x z
infix: (4 * (7 - x)) + z

27. prefix: / x - + 2 w * y z
infix: x / ((2 + w) - (y * z))

28.



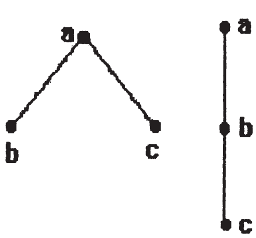
29.



*30.

both inorder and postorder traversal give d c b a

31.



both preorder traversals give a b c

32. If the root has no left child and no right child, return 0 as the height, else invoke the algorithm on the left child if it exists, invoke the algorithm on the right child if it exists, return the maximum of those two values plus 1.

33. If the root has no left child and no right child, return 1, else invoke the algorithm on the left child if it exists, invoke the algorithm on the right child if it exists, return the sum of those two values plus 1.

*34. Consider a simple graph that is a nonrooted tree. A tree is an acyclic and connected graph, so for any two nodes x and y, a path from x to y exists. If the path is not unique, then the two paths diverge at some node n_1 and converge at some node n_2, and there is a cycle from n_1 through n_2 and back to n_1, which is a contradiction.

Now consider a simple graph that has a unique path between any two nodes. The graph is clearly connected. Also, there are no cycles because the presence of a cycle produces a nonunique path between two nodes on the cycle. The graph is thus acyclic and connected and is a nonrooted tree.

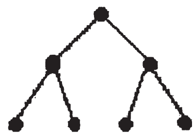
35. If G is a nonrooted tree, then G is connected. Suppose we remove an arc *a* between n_1 and n_2 and G remains connected. Then there is a path from n_1 to n_2. Adding *a* to this path results in a cycle from n_1 to n_1, which contradicts the definition of a tree. Suppose G is connected and removing any single arc makes G unconnected. Then there is a unique path between any two nodes and the graph is a nonrooted tree (Exercise 34).

36. Let G be a nonrooted tree and add an arc *a* between n_1 and n_2. Because G was originally connected, there was a path between n_1 and n_2; adding *a* to this path results in a cycle from n_1 to n_1. If adding one arc to G results in a graph with exactly one cycle, then the original graph was acyclic and connected, a nonrooted tree.

37. Using induction on n, a tree with 2 nodes has 2 nodes of degree one. Assume that a k-node tree has ≥ 2 nodes of degree one. Consider any tree with k + 1 nodes. Remove a leaf x (and its arc); the result is a tree with k nodes and, by the inductive hypothesis, $\geq$ 2 nodes of degree one. Putting node x back to obtain the original tree, node x has degree one but the parent of x may go from degree one to degree two; therefore the number of degree one nodes either increases by one or stays the same, so is still ≥ 2.

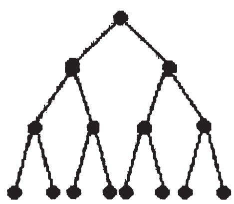
*38. Proof is by induction on d. For d = 0, the only node is the root, and $2^0 = 1$. Assume that there are at most 2^d nodes at depth d, and consider depth d + 1. There are at most two children for each node at depth d, so the maximum number of nodes at depth d + 1 is $2 \cdot 2^d = 2^{d+1}$.

39. a.



7 nodes

b.



15 nodes

c. $2^{h+1} - 1$

d. Proof is by induction on h. For h = 0, the tree consists only of the root, so the number of nodes = 1 = $2^{0+1} - 1$. Assume that a full binary tree of height h has $2^{h+1} - 1$ nodes. Consider a full binary tree of height h + 1. A full tree has the maximum number of leaves which, by Exercise 38, will be 2^{h+1}. Removing the leaves and associated arcs gives a full binary tree of height h, with $2^{h+1} - 1$ nodes by the inductive assumption. The total number of nodes in the original tree is

$$2^{h+1} + 2^{h+1} - 1 = 2 \cdot 2^{h+1} - 1 = 2^{h+2} - 1$$

Or, since by Exercise 36 there are 2^d nodes at each level, the total number of nodes is

$$1 + 2 + 2^2 + \ldots + 2^h = 2^{h+1} - 1$$

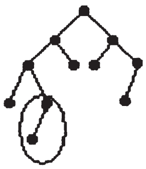
by Example 15 of Section 2.2. This approach does not avoid induction, however, because this result was obtained by induction.

40. a. In a full binary tree, all internal nodes have two children, so the total number of "children nodes" is 2x; the only "non-child" node is the root, so there is a total of 2x + 1 nodes.

b. From part a, there are 2x + 1 total nodes, x of which are internal, leaving 2x + 1 - x = x + 1 leaves.

c. Consider a full binary tree with n nodes; let x be the number of internal nodes. From part a, $n = 2x + 1$. Therefore, $x = (n - 1)/2$. From part b, the number of leaves $= x + 1 = (n - 1)/2 + 1 = (n + 1)/2$.

41. Let x be the number of nodes with two children; we want to show there are $x + 1$ leaves; one proof is by induction on x. If $x = 0$, then the tree must be a "chain," i.e., it looks like this and there is only 1 leaf; $1 = x + 1$. Assume in any binary tree with x nodes having two children that there are $x + 1$ leaves. Now consider a binary tree where $x + 1$ nodes have two children. Remove the subtree rooted by one of the children of a node with two children at the maximum depth at which such a node occurs, for example:



This subtree has no nodes with two children, and exactly one leaf. The remaining tree has x nodes with two children and, by the inductive hypothesis, $x + 1$ leaves. Thus the original tree had $x + 2$ leaves.

Or, do induction on the number of leaves and "grow" the tree. There are two cases: adding to a leaf produces no new leaves, and adding a second child increments both x and the number of leaves.

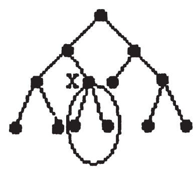
Still another proof parallels that of Exercise 40. Let x = the number of nodes with two children, let y = the number of nodes with one child. Then the total number of nodes is$2x + y + 1$, and the number of internal nodes is $x + y$, so the number of leaves is $2x + y + 1 - (x + y) = x + 1$.

*42. By Exercise 39, a full tree of height $h - 1$ has $2^h - 1$ nodes. When $n = 2^h$, this is the beginning of level h. The height h remains the same until $n = 2^{h+1}$, when it increases by 1. Therefore for $2^h \leq n < 2^{h+1}$, the height of the tree remains the same, and is given by $h = \lfloor \log n \rfloor$. For example, for $2^2 \leq n < 2^3$, that is, n = 4, 5, 6, or 7, the tree height is 2, and $2 = \lfloor \log n \rfloor$.

43. Let x = the number of nodes with 2 children, let y = the number of nodes with 1 child. Then the total number n of nodes is $2x + y + 1$ (1 is for the root, the only non-child node), and the number of leaves is $x + 1$ (Exercise 41). Each leaf contributes 2 null pointers and each node with one child contributes one. Therefore the total number of null pointers is $2(x + 1) + y = 2x + 2 + y = n + 1$.

44. The proof is by induction on i, $i \geq 0$. For $i = 0$, a tree with no internal nodes consists only of the single root node. In this case $E = 0$, $I = 0$, and $i = 0$, so the equation $E = I + 2i$ is true. Assume that in any tree with k internal nodes, all of which have two children, $E_k = I_k + 2k$.

Now consider a tree with k + 1 internal nodes, all of which have two children. We want to show that $E_{k+1} = I_{k+1} + 2(k + 1)$. In such a tree, pick an internal node x whose two children are leaves, and delete the two leaves(see figure). Now x is a leaf. Denote the path length to x by m. The new tree has k internal nodes (since we reduced



the number of internal nodes by 1). By the inductive hypothesis,

$$E_k = I_k + 2k \qquad (1)$$

Also,

$I_k = I_{k+1} - m$ (subtract the path to x because x is no longer an internal node)

$E_k = E_{k+1} - 2(m + 1) + m$ (subtract the two paths to x's children, add the path to x)

Substituting in Equation (1),

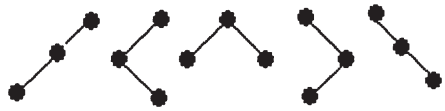
$$E_{k+1} - 2(m + 1) + m = I_{k+1} - m + 2k$$
$$E_{k+1} - m - 2 = I_{k+1} - m + 2k$$
$$E_{k+1} = I_{k+1} + 2k + 2 = I_{k+1} + 2(k + 1)$$

45. a. There is only one binary tree with one node, so B(1) = 1. For a binary tree with n nodes, n > 1, the "shape" of the tree is determined by the "shape" of the left and right subtrees; the two subtrees have a total of n – 1 nodes. Let the left subtree have k nodes; the right subtree then has n – 1 – k nodes; k can range from 0 to n – 1. For each value of k, there are B(k) ways to form the left subtree, then B(n – 1 – k) ways to form the right subtree, so by the Multiplication Principle, there are B(k)B(n – 1 – k) different trees.

b. $B(0) = 1$
$B(1) = 1$
$$B(n) = = \sum_{k=0}^{n-1} B(k)B(n-1-k) = = \sum_{k=1}^{n} B(k-1)B(n-k)$$
which is the same as the Catalan sequence, so by Exercise 82 of Section 3.4,

$$B(n) = \frac{1}{n+1} C(2n, n)$$

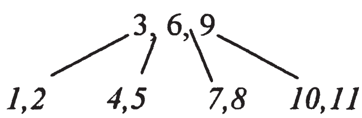
c. $B(3) = \dfrac{1}{3+1}C(6,3) = 5$. The 5 distinct binary trees are:

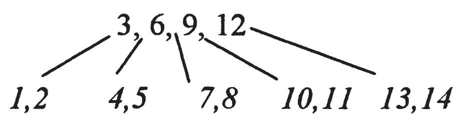


d. $B(6) = \dfrac{1}{6+1}C(12,6) = 132$

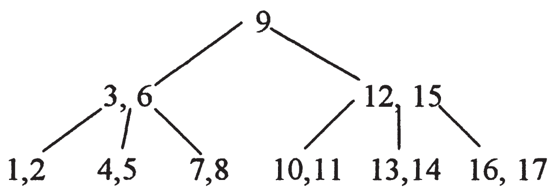
46. *a. 17. The growth of the tree proceeds as follows, building on the tree with 8 nodes:

Insert 9, 10, 11: Insert 12, 13, 14:

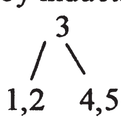




Insert 15, 16, 17:



b. Proof is by induction on n. When n = 2, the tree is



and the bottom level has 2 nodes. $2 = 2\cdot 3^{2-2}$, true. Assume that when the tree has just achieved k levels, the bottom level has $2\cdot 3^{k-2}$ nodes. Now let the tree grow just to achieve level k + 1. By the inductive assumption, the level above the new bottom level has $2\cdot 3^{k-2}$ nodes, each of which has 3 links to child nodes. Hence the number of nodes in the bottom level is $2\cdot 3^{k-2}\cdot 3 = 2\cdot 3^{k-1}$.

c. From part (b), the number of nodes when the tree has attained n levels is

1 (the root) $+ 2\cdot 3^0 + 2\cdot 3^1 + \ldots + 2\cdot 3^{n-2}$

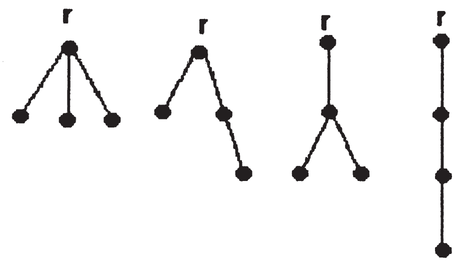
The root node contains one data value; all other nodes contain two data values, so the total number of nodes is

$1 + 4(3^0 + 3^1 + \ldots + 3^{n-2}) = 1 + 4\left(\dfrac{3^n - 3}{6}\right)$ (this equality can be proved by induction)

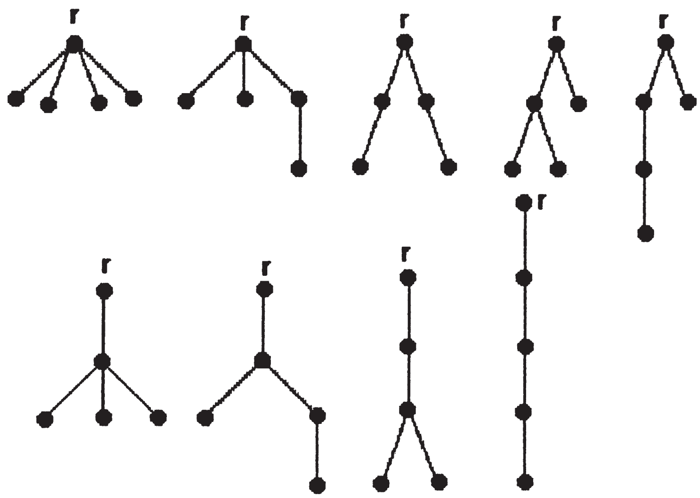
This expression gives 5 when n = 2 and 17 when n = 3, which agrees with earlier results.

47. The chromatic number of a tree is 2; all the nodes at an odd level can be one color, and all the nodes at an even level can be the other color.

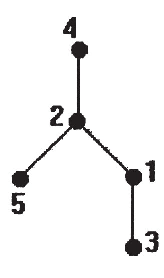
*48.



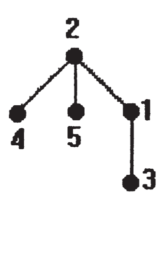
49.



50.

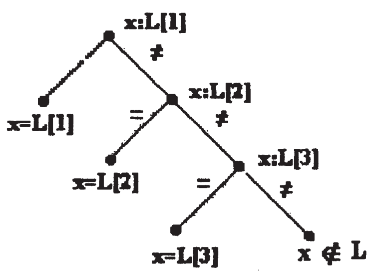


51.

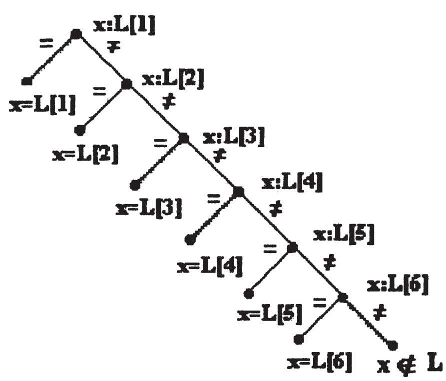


EXERCISES 5.3

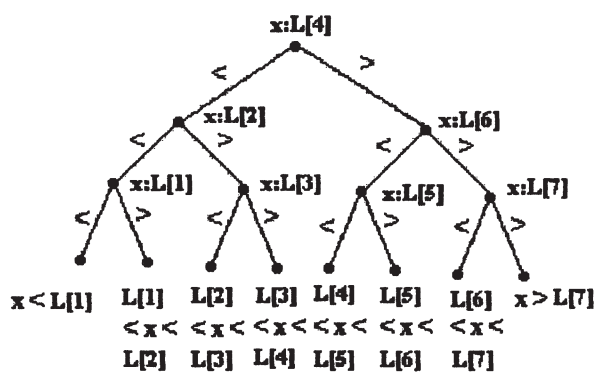
*1.



2.

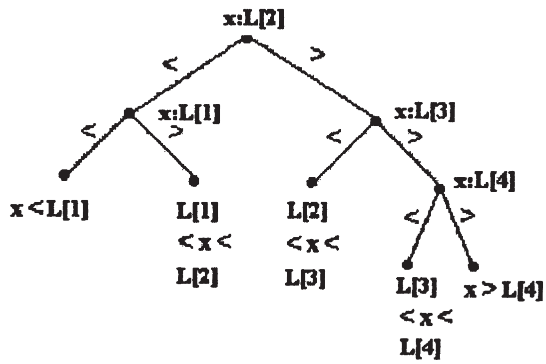


3.



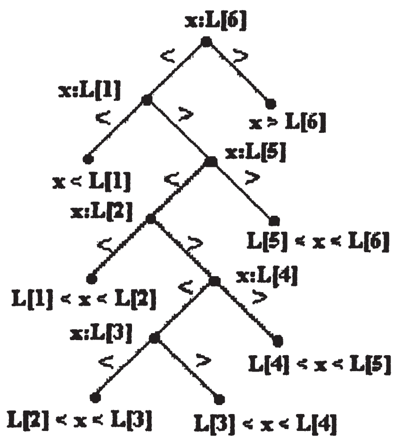
depth = 3 = 1 + $\lfloor \log 7 \rfloor$

4.



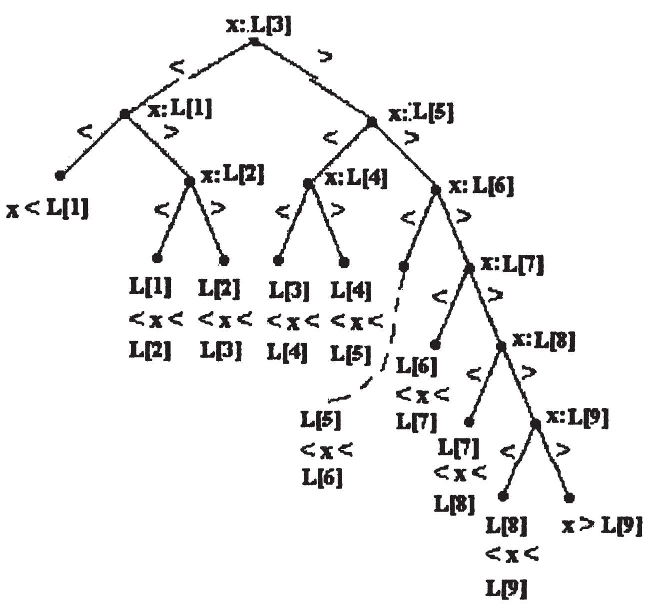
depth = 3 = 1 + $\lfloor \log 4 \rfloor$

*5.



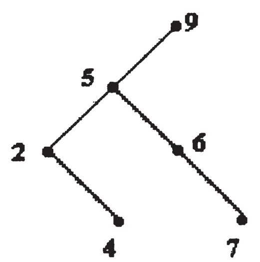
depth = 6; algorithm is not optimal

6.



depth = 6; algorithm is not optimal

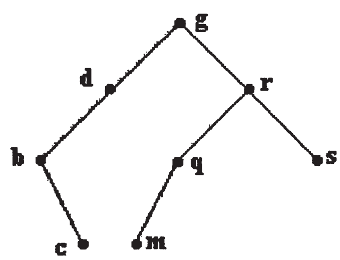
*7. a.



depth = 3

b. 2.83

8. a.

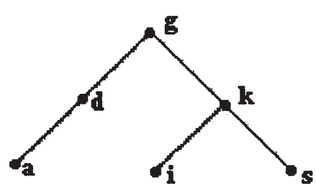


depth = 3

b. 2.75

9. a. $\lfloor \log 6 \rfloor + 1 = 3$

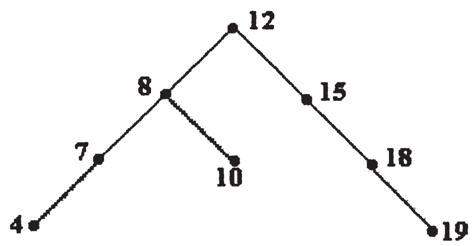
b. For example: g, d, a, k, i, s



depth = 2 (Note that a binary search tree of depth 2 implies 3 compares in the worst case.)

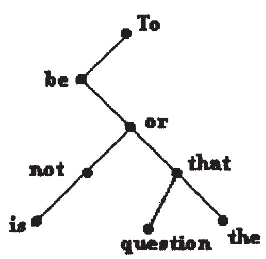
10. a. $\lfloor \log 9 \rfloor + 1 = 4$

b. For example: 12, 8, 10, 15, 18, 19, 7, 4



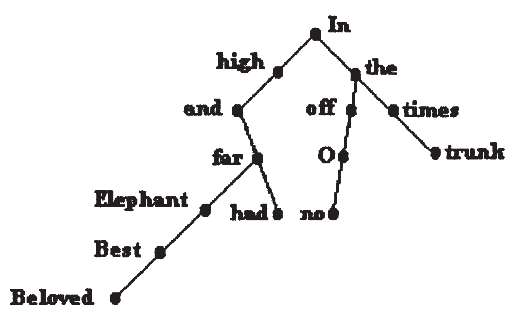
depth = 3 (Note that a binary search tree of depth 3 implies 4 compares in the worst case.)

*11.



be is not or question that the To

12.



and Beloved Best Elephant far had high in no O off the times trunk

*13. a. $\lceil \log 4! \rceil = \lceil \log 24 \rceil = 5$

b. $\lceil \log 8! \rceil = \lceil \log 40320 \rceil = 16$

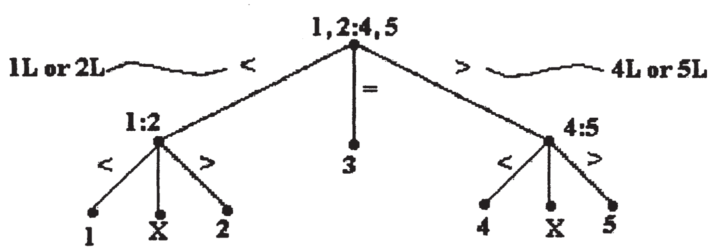
c. $\lceil \log 16! \rceil = \lceil \log 2.09 \text{ x } 10^{13} \rceil = 45$

14.

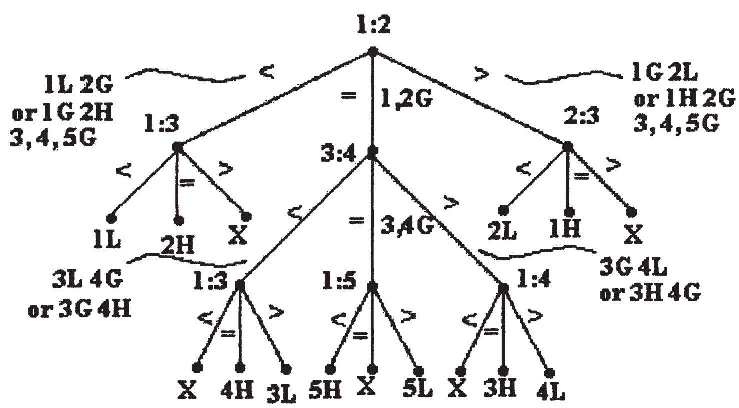
n	Exer. 13	Sel. sort	Mergesort
4	5	6	5
8	16	28	17
16	45	120	49

Mergesort is close to optimal.

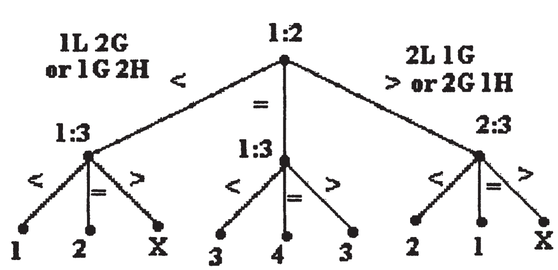
15. a. 5 (any of the five coins can be the light one)
    b. 2 (the minimum depth for a ternary tree with 5 leaves)
    c.



*16. a. 10 (any of the five coins can be heavy or light)
    b. 3 (the minimum depth for a ternary tree with 10 leaves)
    c.

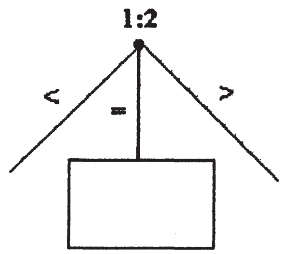


17. a. 4 (any of the four coins can be the counterfeit one)
    b. 2 (the minimum depth for a ternary tree with 4 leaves)
    c.



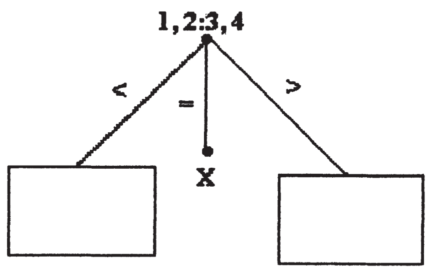
18. a. 8 (any of the four coins can be heavy or light)
    b. 2 (the minimum depth for a ternary tree with 8 leaves)
    c. Suppose that a decision tree of depth 2 can solve the problem.

    Case 1: first compare involves 2 coins. Then the decision tree has the form shown:



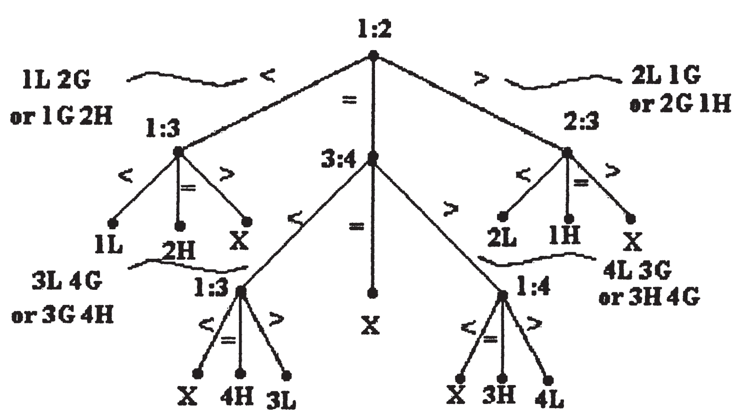
If coins 1 and 2 balance, then within the box, the four possible outcomes of 3L, 3H, 4L, 4H must be determined, but there can be at most 3 leaves produced here.

Case 2: first compare involves 4 coins. Then the decision tree has the form shown (the four coins cannot balance because one of them is bad):



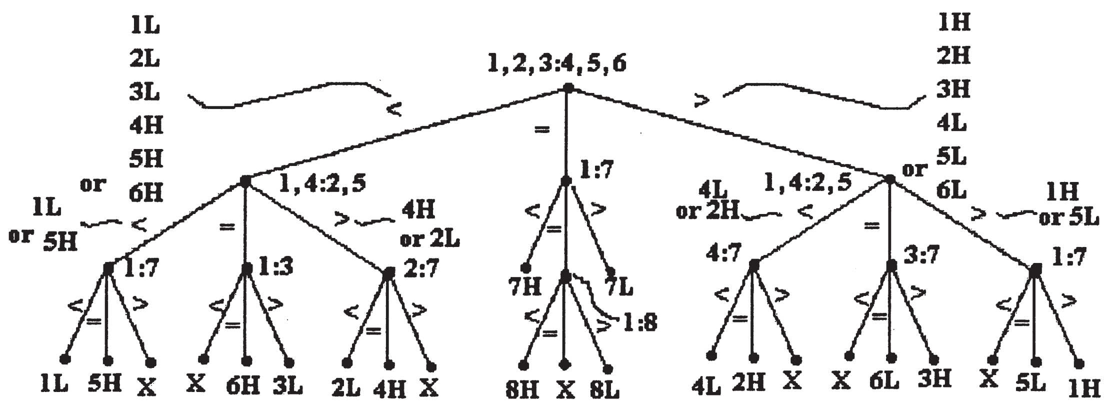
Then within the two boxes, the eight possible outcomes must be determined, but there can be at most 6 leaves produced here.

19.



20. a. 16 (any of the eight coins can be heavy or light)
    b. 3 (the minimum depth for a ternary tree with 16 leaves)

c.

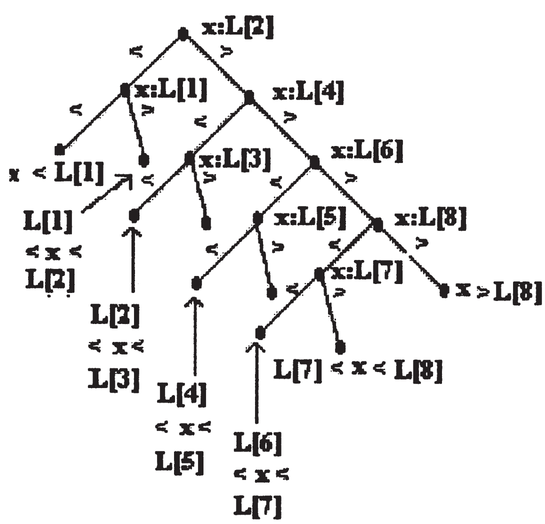


*21. The three-way comparison would be done something like

if (x = node element)
  write "found";
else
  if (x < node element)
  ....
  else
  ...

so that in the worst case, where x is not equal to the node element, 2 comparisons are done at the node. The number of comparisons in the worst case for binary search of n elements is therefore 2 times the depth of the tree, or $2*(1 + \lfloor \log n \rfloor)$.

22. a. (For simplicity, not all leaves are labeled)



b. 10 comparisons (for example, if $L(7) < x < L(8)$). As explained in the solution to Exercise 21, two comparisons are performed at each internal node.

c. The longest branch of the decision tree is approximately n/2 nodes in length, with 2 comparisons at each node, so the search is $\Theta(n)$. This is a higher order of magnitude than the original binary search algorithm, and is equivalent to sequential search.

23. a.
$$\begin{aligned}\log n! &= \log [(n)(n-1)(n-2)\ldots(2)(1)]\\ &= \log n + \log(n-1) + \log(n-2) + \ldots + \log 2 + \log 1\\ &\le \log n + \log n + \log n + \ldots + \log n \quad \text{for } n \ge 1\\ &= n \log n\end{aligned}$$

b.
$$\begin{aligned}\log n! &= \log [(n)(n-1)(n-2)\ldots(2)(1)]\\ &= \log n + \log(n-1) + \log(n-2) + \ldots + \log 2 + \log 1\\ &\ge \log n + \log(n-1) + \ldots + \log\lceil n/2\rceil\\ &\ge \log\lceil n/2\rceil + \log\lceil n/2\rceil + \ldots + \log\lceil n/2\rceil\\ &\ge \lceil n/2\rceil \log\lceil n/2\rceil\\ &\ge \left(\frac{n}{2}\right)\log\left(\frac{n}{2}\right) = \left(\frac{n}{2}\right)(\log n - \log 2) = \left(\frac{n}{2}\right)(\log n - 1)\\ &= \left(\frac{n}{2}\right)\log n - \left(\frac{n}{2}\right) = \left(\frac{n}{4}\right)\log n + \left(\frac{n}{4}\right)\log n - \left(\frac{n}{2}\right) = \left(\frac{n}{4}\right)\log n + \left(\frac{n}{4}\right)(\log n - 2)\\ &\ge \left(\frac{n}{4}\right)\log n \quad \text{because } \log n \ge 2 \text{ for } n \ge 4\end{aligned}$$

EXERCISES 5.4

*1. a. ooue b. iaou c. eee

2. a. bh%% b. wwq c. qhwb

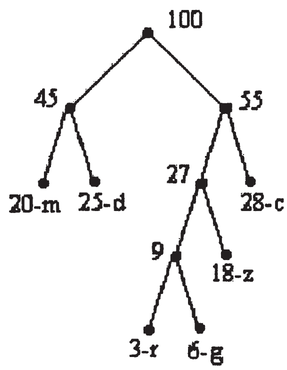
3. a. (pw)a b. paw c. ((a))

4. 319
31771
3175
11119
1111771
111175
139
13771
1375

5. a - 0101
   b - 011
   c - 10
   d - 11

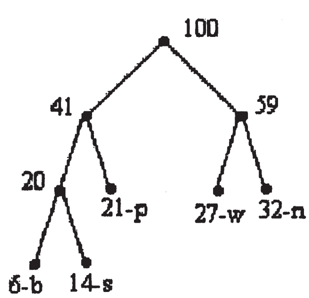
6. r - 000
   s - 001
   t - 01
   u - 1

*7. a.



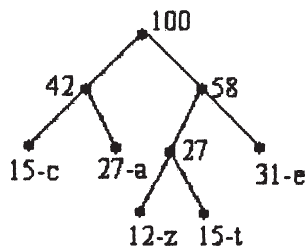
b. c - 11
   d - 01
   g - 1001
   m - 00
   r – 1000
   z - 101

8. a.

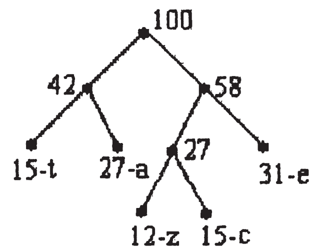


b.
   b – 000
   n - 11
   p - 01
   s - 001
   w - 10

9. a.



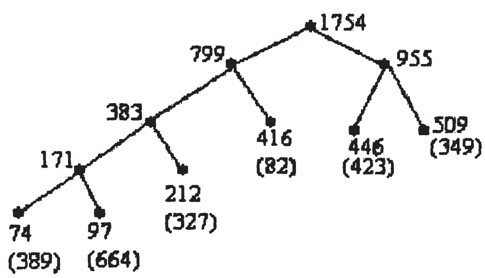
or



b.

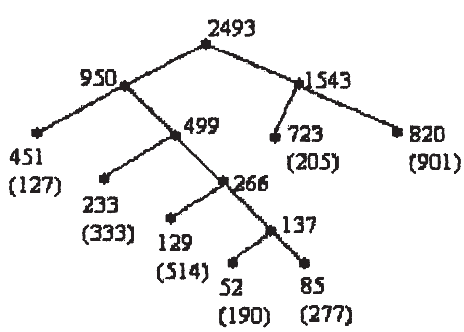
a - 01		a - 01
z - 100		z - 100
t - 101	or	t - 00
e - 11		e - 11
c - 00		c - 101

*10. a.



b. 82 - 01
664 - 0001
327 - 001
349 - 11
423 - 10
389 – 0000

11. a.



b. 190 – 01110
205 – 10
514 – 0110
333 – 010
127 – 00
901 – 11
277 – 01111

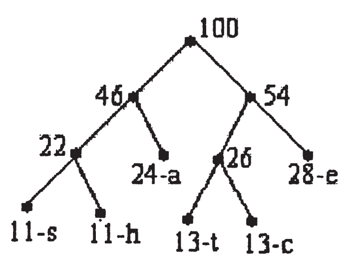
12. Greyscale images contain only luminance information with no color components; the luminance information is more accurately preserved through the various preprocessing steps.

*13. Every single character z has been replaced by the two-character string sh. The frequencies of occurrence of all the characters must be recomputed. For example, if there were 100 characters in the original file, there are now 112. If there were 27 a's per 100 characters in the original file, there are now 27 a's per 112 characters, so the frequency of occurrence of an a is (27/112)*100 = 24.

The new frequencies are:

a	s	h	t	e	c
24	11	11	13	28	13

The new Huffman tree is

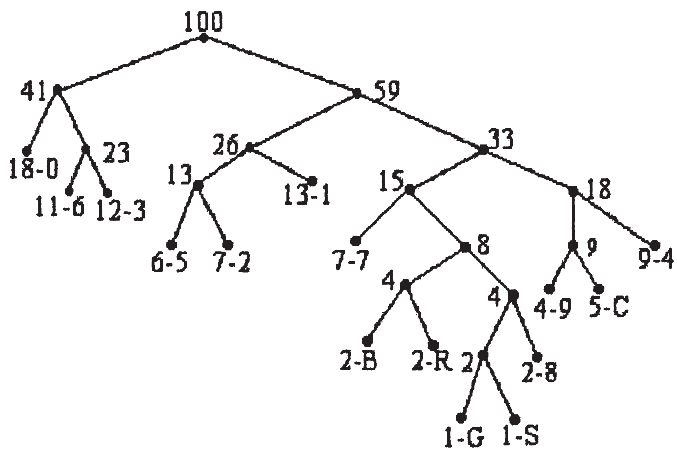


and a new encoding (one of several possibilities) is

s - 000
h - 001
a - 01
t - 100
c - 101
e - 11

14. Aside from punctuation or uppercase characters, the single character that appears least frequently (1 time) is k, so it would have one of the longest codes. The single character that appears most frequently is the blank character separating words, so it would have one of the shortest codes. The next most frequent character (27 times) is t, so it would also have a short code.

15. a. A possible Huffman tree is:



which would lead to the codes:

B - 110100	0 - 01	5 - 1000
C - 11101	1 - 101	6 - 010
G - 1101100	2 - 1001	7 - 1100
R - 110101	3 - 011	8 - 110111
S - 1101101	4 - 1111	9 - 11100

b. The space required is

$9 \times 5 \times 10^8 (.02*6 + .05*5 + .01*7 + .02*6 + .01*7 + .18*2 + .13*3 + .07*4 + .12*3 + .09*4 + .06*4 + .11*3 + .07*4 + .02*6 + .04*5)$

$= 9 \times 5 \times 10^8 (3.55)$

as compared to $9 \times 5 \times 10^8 (8)$. The new file takes 3.55/8 = 44 percent of the space of the original file.

*16. a. Because we are assuming that $f(x) < f(p)$, we can write $f(x) + j = f(p)$ for some positive quantity j. Because x is above p in tree T (Figure 5.58a), we can write $d(x) + k = d(p)$ for some positive integer k. The contributions to E(T) from nodes x and p are given by

$$\begin{aligned} f(x)d(x) + f(p)d(p) &= f(x)d(x) + [(f(x) + j)(d(x) + k)] \\ &= 2f(x)d(x) + jd(x) + kf(x) + jk \end{aligned}$$

In tree T' (Figure 5.58b), the contributions to E(T') from nodes x and p are given by the following (using the original d(x) and d(p) values):

$$\begin{aligned} f(x)d(p) + f(p)d(x) &= f(x)(d(x) + k) + (f(x) + j)d(x) \\ &= 2f(x)d(x) + kf(x) + jd(x) \end{aligned}$$

which is jk smaller than the previous expression.

b. In Figure 5.59d, the contribution to E(B) from the node with frequency f(x) + f(y) is

$$[f(x) + f(y)]*r$$

where r is the depth of that node. The corresponding contribution to E(B') (Figure 5.59c) is

$$\begin{aligned} f(x)(r + 1) + f(y)*(r + 1) &= f(x)*r + f(y)*r + f(x) + f(y) \\ &= [f(x) + f(y)]*r + f(x) + f(y) \end{aligned}$$

so E(B') exceeds E(B) by f(x) + f(y).