Printed by: longa@nku.edu. Printing is for personal, private use only. No part of this book may be reproduced or transmitted without publisher's prior permission. Violators will be prosecuted.

EXAMPLE 31 Sometimes a clever observation can take the place of an inductive proof. Every arc of a tree connects one node to its parent. Each node of the tree except the root has a parent, and there are $n - 1$ such nodes, therefore $n - 1$ arcs. Each arc has two arc ends, so there are $2(n - 1)$ arc ends. ●

SECTION 6.2 REVIEW

TECHNIQUES

- Construct expression trees.
- Construct array and pointer representations for binary trees.
- Conduct preorder, inorder, and postorder traversals of a tree.

MAIN IDEAS

- Binary trees can be represented by arrays and by linked structures.
- Recursive procedures exist to systematically visit every node of a tree.

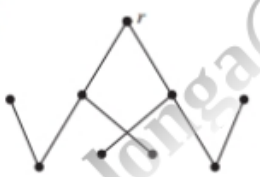
EXERCISES 6.2

1. Which of the following graphs are trees with root r? If a graph is a tree, draw it in a more conventional way. If not, say what property fails.

a.

c. 

b. 

d. 

2. Which of the following graphs are binary trees with root r? If the graph is not a binary tree, say what property fails.

a. 

b. 

c. 

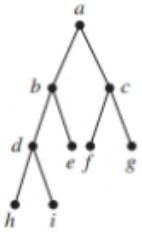
d. 

3. Sketch a picture of each of the following trees.
   a. Tree with five nodes and depth 1
   b. Full binary tree of depth 2
   c. Tree of depth 3 where each node at depth i has $i + 1$ children

DO NOT COPY longa@nku.edu

Printed by: longa@nku.edu. Printing is for personal, private use only. No part of this book may be reproduced or transmitted without publisher's prior permission. Violators will be prosecuted.

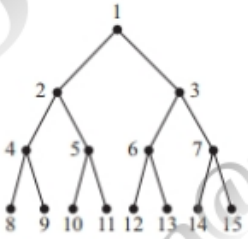
4. Answer the following questions about the accompanying graph with node a as the root.
   a. Is it a binary tree?
   b. Is it a full binary tree?
   c. Is it a complete binary tree?
   d. What is the parent of e?
   e. What is the right child of e?
   f. What is the depth of g?
   g. What is the height of the tree?



In Exercises 5–8, draw the expression tree.

5. $[(2 * x - 3 * y) + 4 * z] + 1$
6. $[(x - 2) * 3] + (5 + 4)$
7. $1 - (2 - [3 - (4 - 5)])$
8. $[(6/2) * 4] + [(1 + x) * (5 + 3)]$
9. Write the left child–right child array representation for the binary tree in the figure.

10. Write the left child–right child array representation for the binary tree in the figure.



11. Draw the binary tree corresponding to the left child–right child representation that follows. (1 is the root.)

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

DO NOT COPY longa@nku.edu

Printed by: longa@nku.edu. Printing is for personal, private use only. No part of this book may be reproduced or transmitted without publisher's prior permission. Violators will be prosecuted.

12. Draw the binary tree corresponding to the left child–right child representation that follows. (1 is the root.)

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

13. Write the left child–right child array representation for the binary search tree that is created by processing the following list of words: "All Gaul is divided into three parts" (see Exercise 43 of Section 5.1). Also store the name of each node.
14. Write the left child–right child array representation for the binary search tree that is created by processing the following list of words: "We hold these truths to be self-evident, that all men are created equal" (see Exercise 43 of Section 5.1). Also store the name of each node.
15. In the following binary tree representation, the left child and parent of each node are given. Draw the binary tree. (1 is the root.)

	Left child	Parent
1	2	0
2	4	1
3	0	1
4	0	2
5	0	2
6	0	3

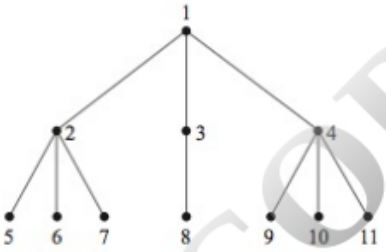
16. The following represents a tree (not necessarily binary) where, for each node, the leftmost child and the closest right sibling of that node are given. Draw the tree. (1 is the root.)

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

DO NOT COPY longa@nku.edu

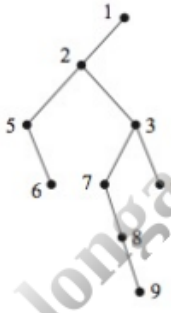
Printed by: longa@nku.edu. Printing is for personal, private use only. No part of this book may be reproduced or transmitted without publisher's prior permission. Violators will be prosecuted.

17. a. For the following tree, write the leftmost child–right sibling array representation described in Exercise 16.



b. Now draw the binary tree that results from treating the answer to part (a) as a left child–right child binary tree representation. An arbitrary tree can thus be thought of as having a binary tree representation.

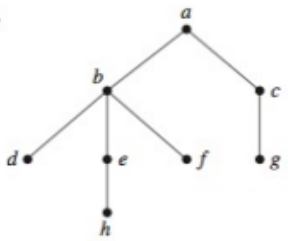
18. The following binary tree is the representation of a general tree (as in part (b) of Exercise 17). Draw the tree.



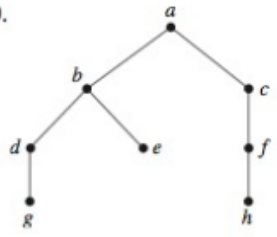
DO NOT COPY longa@nku.edu

For Exercises 19–24, write the list of nodes resulting from a preorder traversal, an inorder traversal, and a postorder traversal of the tree.

19.

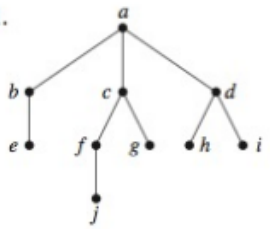


20.

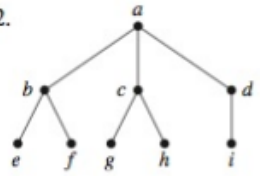


Printed by: longa@nku.edu. Printing is for personal, private use only. No part of this book may be reproduced or transmitted without publisher's prior permission. Violators will be prosecuted.

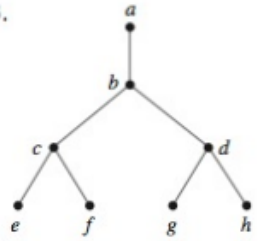
21.



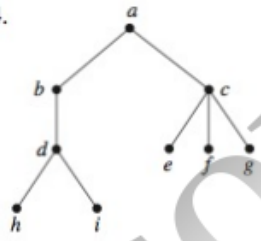
22.



23.



24.



25. Write in prefix and postfix notation: $3/4 + (2 - y)$.
26. Write in prefix and postfix notation: $(x * y + 3/z) * 4$.
27. Write in infix and postfix notation: $- * + 2\ 3 * 6\ x\ 7$.
28. Write in infix and postfix notation: $- + - x\ y\ z\ w$.
29. Write in prefix and infix notation: $4\ 7\ x - * z +$.
30. Write in prefix and infix notation: $x\ 2\ w + y\ z * - /$.
31. Evaluate the postfix expression $8\ 2\ /\ 2\ 3 * +$.
32. Evaluate the postfix expression $5\ 3 +\ 1\ 3 +\ /\ 7 *$.
33. Draw a single tree whose preorder traversal is

$$a, b, c, d, e$$

and whose inorder traversal is

$$b, a, d, c, e$$

34. Draw a single tree whose inorder traversal is

$$f, a, g, b, h, d, i, c, j, e$$

and whose postorder traversal is

$$f, g, a, h, i, d, j, e, c, b$$

35. Find an example of a tree whose inorder and postorder traversals yield the same list of nodes.
36. Find two different trees that have the same list of nodes under a preorder traversal.
37. Informally describe a recursive algorithm to compute the height of a binary tree, given the root node.
38. Informally describe a recursive algorithm to compute the number of nodes in a binary tree, given the root node.
39. Prove that a simple graph is a nonrooted tree if and only if there is a unique path between any two nodes.

DO NOT COPY longa@nku.edu

Printed by: longa@nku.edu. Printing is for personal, private use only. No part of this book may be reproduced or transmitted without publisher's prior permission. Violators will be prosecuted.

40. What is the minimum number of nodes and arcs that need to be deleted to reduce a full binary tree of height ≥ 2 to a forest of 4 binary trees?
41. Let G be a simple graph. Prove that G is a nonrooted tree if and only if G is connected and if the removal of any single arc from G makes G unconnected.
42. Let G be a simple graph. Prove that G is a nonrooted tree if and only if G is connected and the addition of one arc to G results in a graph with exactly one cycle.
43. Prove that a binary tree has at most 2^d nodes at depth d.
44. Prove that a tree with n nodes, $n \geq 2$, has at least two nodes of degree 1.
45. a. Draw a full binary tree of height 2. How many nodes does it have?
    b. Draw a full binary tree of height 3. How many nodes does it have?
    c. Conjecture how many nodes there are in a full binary tree of height h.
46. Prove your conjecture from Exercise 45(c) three different ways.
    a. Use induction on the height h of the full binary tree. (*Hint*: Use Exercise 43.)
    b. Add up the nodes at each level of the tree (*Hint*: Use Exercise 43).
    c. Use structural induction.
47. a. Prove that a full binary tree with x internal nodes has $2x + 1$ total nodes.
    b. Prove that a full binary tree with x internal nodes has $x + 1$ leaves.
    c. Prove that a full binary tree with n nodes has $(n - 1)/2$ internal nodes and $(n + 1)/2$ leaves.
48. Prove that the number of leaves in any binary tree is 1 more than the number of nodes with two children.
49. Find an expression for the height of a complete binary tree with n nodes. (*Hint:* Use Exercise 45.)
50. Prove that in the pointer representation of a binary tree with n nodes there are $n + 1$ null pointers. (*Hint:* Use Exercise 48).
51. Find the chromatic number of a tree (see Section 6.1, Exercise 80).
52. Let E be the external path length of a tree, that is, the sum of the path lengths to all the leaves. Let I be the internal path length, that is, the sum of the path lengths to all the internal nodes. Let i be the number of internal nodes. Prove that in a binary tree where all internal nodes have two children, $E = I + 2i$.
53. Let $B(n)$ represent the number of different binary trees with n nodes.
    a. Define $B(0)$ to have the value 1 (there is one binary tree with 0 nodes). Prove that $B(n)$ is given by the recurrence relation

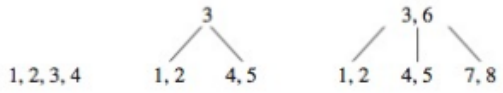
$$B(1) = 1$$

$$B(n) = \sum_{k=0}^{n-1} B(k)B(n - 1 - k)$$

    b. Compare the sequence $B(n)$ to the sequence of Catalan numbers (Exercise 97, Section 4.4). Write the closed-form expression for $B(n)$.
    c. Compute the number of different binary trees with 3 nodes. Draw all these trees.
    d. Compute the number of different binary trees with 6 nodes.
54. In the data structure known as a B-tree of order 5, each node of the tree can contain multiple data values, maintained in sorted order. Between and around the data values at an internal node are arcs that lead to children of the node. New data values are inserted into the leaf nodes of the tree, but when a leaf (or internal node) gets up to five values, it splits in two and the median value pops up to the next level of the tree. The figure shows the tree at various points as the data values 1 through 8 are inserted into an initially empty tree.

DO NOT COPY longa@nku.edu

Printed by: longa@nku.edu. Printing is for personal, private use only. No part of this book may be reproduced or transmitted without publisher's prior permission. Violators will be prosecuted.



a. The minimum number of data values to insert into a B-tree of order 5 to force it to have two levels is 5. Find the minimum number of data values required to force the tree to have three levels.

b. Prove that when a B-tree of order 5 has the minimum number of data values to force it to have n levels, $n \geq 2$, the bottom level contains $2(3^{n-2})$ nodes.

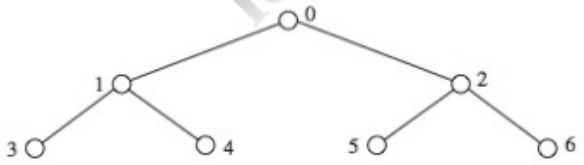
c. Find (and justify) a general expression for the minimum number of data values required to force a B-tree of order 5 to have n levels. $\left(\textit{Hint}: 3^0 + 3^1 + \cdots + 3^{n-2} = \left(\frac{3^n - 3}{6}\right).\right)$

In Exercises 55 and 56, two trees are *isomorphic* if there is a bijection f: $N_1 \rightarrow N_2$, where f maps the root of one tree to the root of the other and where $f(y)$ is a child of $f(x)$ in the second tree when y is a child of x in the first tree. Thus the two trees shown are isomorphic graphs but not isomorphic trees (in part (a) the root has two children and in part (b) it does not). These are the only two nonisomorphic trees with three nodes.

55. Draw all the nonisomorphic trees with four nodes.

56. Draw all the nonisomorphic trees with five nodes.

57. One of the most efficient sorting algorithms is *HeapSort*, which sorts an array of values into increasing order. To understand how the *HeapSort* algorithm works, it is best to imagine that the array elements are stored in level order as the nodes of a binary tree. Thus the values in a 7-element array that is indexed from 0 through 6 would be stored in a binary tree with element 0 at the root, elements 1 and 2 at depth 1, and so on.



A *heap* is a binary tree in which the value at every node is greater than the value at its two child nodes. *HeapSort* is a two-phase process. The first phase is to reorganize the tree elements into a heap (more on this later), and the second is to sort the heap. The key idea is that in a heap, the largest element is the root of the tree; its proper place in the sorted array is at the end of the unsorted section of the array (at the lowest, rightmost tree element not yet in its sorted position). The tree root gets thrown to the last unsorted position, and the element that formerly occupied that position must be inserted back into the unsorted section in such a way as to preserve the heap property. Consider the following binary tree, which is a heap—each node value is larger than the values at the two child nodes.

DO NOT COPY
longa@nku.edu