EX 13.9 In terms of the final state of a stack, does it matter how the pop operations are intermixed with the push operations? Does it matter how the push operations are intermixed among themselves? Explain using examples.

EX 13.10 Would a tree data structure be a good choice to represent a family tree that shows lineage? Why or why not? Would a binary tree be a better choice? Why or why not?

EX 13.11 What data structure would be a good choice to represent the links between various Web sites? Give an example.

**Programming Projects**

PP 13.1 Consistent with the example from Chapter 8, design and implement an application that maintains a collection of DVDs using a linked list. In the main method of the driver class, add various DVDs to the collection and print the list when complete.

PP 13.2 Modify the MagazineRack program presented in this chapter by adding delete and insert operations into the MagazineList class. Have the Magazine class implement the Comparable interface, and base the processing of the insert method on calls to the compareTo method in the Magazine class that determines whether one Magazine title comes before another alphabetically. In the driver, exercise various insertion and deletion operations. Print the list of magazines when complete.

PP 13.3 Design and implement a version of selection sort (from Chapter 10) that operates on a linked list of nodes that each contain an integer.

PP 13.4 Design and implement a version of insertion sort (from Chapter 10) that operates on a linked list of nodes that each contain an integer.

PP 13.5 Design and implement an application that simulates the customers waiting in line at a bank. Use a queue data structure to represent the line. As customers arrive at the bank, customer objects are put in the rear of the queue with an enqueue operation. When the teller is ready to service another customer, the customer object is removed from the front of the queue with a dequeue operation. Randomly determine when new customers arrive at the bank and when current customers are finished at the teller window. Print a message each time an operation occurs during the simulation.
PP 13.6 Modify the solution to the PP 13.5 so that it represents eight tellers and therefore eight customer queues. Have new customers go to the shortest queue. Determine which queue had the shortest waiting time per customer on average.

PP 13.7 Design and implement an application that evaluates a postfix expression that operates on integer operands using the arithmetic operators +, −, *, /, and %. We are already familiar with infix expressions, in which an operator is positioned between its two operands. A postfix expression puts the operators after its operands. Keep in mind that an operand could be the result of another operation. This eliminates the need for parentheses to force precedence. For example, the following infix expression:

\[(5 + 2) \ast (8 - 5)\]

is equivalent to the following postfix expression:

\[5 \ 2 \ + \ 8 \ 5 \ - \ \ast\]

The evaluation of a postfix expression is facilitated by using a stack. As you process a postfix expression from left to right, you encounter operands and operators. If you encounter an operand, push it on the stack. If you encounter an operator, pop two operands off the stack, perform the operation, and push the result back on the stack. When you have processed the entire expression, there will be one value on the stack, which is the result of the entire expression.

You may want to use a StringTokenizer object to assist in the parsing of the expression. You can assume the expression will be in valid postfix form.

PP 13.8 Design and implement a program that prompts the user to enter a string and then performs two palindrome tests. The first should use a single stack to test whether the string is a palindrome. The second should use two stacks to test whether the string is a palindrome when capitalization, spaces, punctuation, and other non-alphanumeric characters are ignored. The program should print the results of both tests.

PP 13.9 Design and implement a class named StringTree, a binary tree for storing String objects in alphabetic order. Each node in the tree should be represented by a Node class, which stores the string value and pointers to the right and left child nodes. For any node value in the tree, the value of its left child should