As you can see, the `contains`, `add`, and `remove` methods take, in the worst case, an amount of time proportional to the height of the tree. When the tree is reasonably "bushy," this is $O(\log n)$, where n is the number of nodes in the tree.

> **In our simplified implementation, if the values to be added to the BST arrive not in random order but close to ascending or descending order, the tree may lose its shape and deteriorate into an almost linear structure.**

A more sophisticated implementation would use algorithms that keep the tree "balanced" (but such algorithms are outside the scope of this book).

As a lab exercise, fill in the blanks in the `removeRoot` method in `JM\Ch23\BST\MyTreeSet.java`. For "extra credit," make `MyTreeSet Iterable` by implementing an `iterator` method and a `MyTreeSetIterator` class. See Section <...> for an example. Your iterator should perform inorder traversal of the tree: left subtree, root, right subtree. Use a `Stack<TreeNode>` in your `MyTreeSetIterator` class. Do not implement the `remove` method for your iterator.

23.6 *Lab:* Morse Code

Morse Hall, the Mathematics Department building at Phillips Academy in Andover, Massachusetts, is named after Samuel F. B. Morse, who graduated from the academy in 1805.

In 1838, Samuel Morse devised a signaling code for use with his electromagnetic telegraph. The code used two basic signaling elements: the "dot," a short-duration electric current, and the "dash," a longer-duration signal. The signals lowered an ink pen mounted on a special arm, which left dots and dashes on the strip of paper moving beneath. Morse's code gained wide acceptance and, in its international form, is still in use. (Samuel Morse also achieved distinction as an artist, particularly as a painter of miniatures, and between 1826 and 1845 served as the first president of the National Academy of Design.)

In 1858 Queen Victoria sent the first transatlantic telegram of ninety-eight words to congratulate President James Buchanan of the United States. The telegram started a

Draft. Copyright © 2006 by Skylight Publishing.

new era of "instant" messaging — it took only sixteen and a half hours to transmit via the brand new transatlantic telegraph cable.

In this project, we will simulate a telegraph station that encodes messages from text to Morse code and decodes Morse code back to plain text. The encoding is accomplished simply by looking up a symbol in a `TreeMap<Character, String>` that associates each symbol with its Morse code string. The decoding is implemented with the help of a binary "decoding" tree of our own design. The Morse code for each letter represents a path from the root of the tree to some node: a "dot" means go left, and a "dash" means go right. The node at the end of the path contains the symbol corresponding to the code.

The "Telegraph" is implemented in two classes: `Telegraph` and `MorseCode`. In addition, `MorseCode` uses the `TreeNode` class described in Section 23.2. The `Telegraph` class opens two windows on the screen, "London" and "New York," and handles the text entry fields and GUI events in them. We have written this class for you. The `MorseCode` class implements encoding and decoding of text. All the methods in this class are static. The `start` method initializes the encoding map and the decoding tree; the private method `treeInsert` inserts a given symbol into the decoding tree, according to its Morse code string; the public `encode` and `decode` methods convert plain text into Morse code and back, respectively. Your task is to supply all the missing code in the `MorseCode` class.

The `Telegraph` class and the unfinished `MorseCode` class are in `JM\Ch23\MorseCode`. `TreeNode.java` is also provided in the same folder. Put together a project with `Telegraph`, `MorseCode`, and `TreeNode` and test your program.

23.7 Case Study and Lab: Java Messenger

In 1996, America Online introduced its subscribers to the "Buddy List," which allowed AOL members to see when their friends were online. A year later, AOL introduced the *AOL Instant Messenger* (*AIM*). In this case study we implement our own instant "messaging" application, *Java Messenger*. In our application, the same user logs in several times under different screen names, and messages are sent from one window to another on the same screen. Also, in this toy version, all other logged-in users are considered "buddies" of a given user.

Draft. Copyright © 2006 by Skylight Publishing.