```
// Example 4.4 joi/examples/TreeMapDemo.java
//
//
// Copyright 2003 Bill Campbell and Ethan Bolker

import java.util.TreeMap;
import java.util.Iterator;
import java.util.Set;
import java.util.Collection;
import java.util.Map;

// A class illustrating the use of TreeMap.  A typical run:
//
// %> java TreeMapDemo
// Store 3 wrapped ints, keys "one", "two", "three".
// The wrapped int stored for "two" is 2
//
// Iterate over keys, get each value.
// Note that key order is aphabetical:
// The value for key one is 1
// The value for key three is 3
// The value for key two is 2
//
// Iterate over the values:
// 1
// 3
// 2
//
// Iterate over the key-value pairs:
// The value for the entry with key one is 1
// The value for the entry with key three is 3
// The value for the entry with key two is 2
//
// How a TreeMap represents itself as a String:
// {one=1, three=3, two=2}
//
// Store a different value at key "two"
// {one=1, three=3, two=2222}
//
// Store map.get( "one" ) at key "two"
// {one=1, three=3, two=1}
//
// A TreeMap with Integer keys mapping to String values
// {1=I, 2=II, 3=III}
// %>

public class TreeMapDemo
{
    public static void main( String[] args )
    {
        Terminal terminal = new Terminal(); // for input and output

        TreeMap map = new TreeMap();

        // Put in some ints (each wrapped up as an Integer object)
        terminal.println(
```



```
          "Store 3 wrapped ints, keys \"one\", \"two\", \"three\".");
        map.put("one",   new Integer(1) );
        map.put("two",   new Integer(2) );
        map.put("three", new Integer(3) );

        // get the value associated with a key;
        // notice the required cast.
        Integer wrappedInt = (Integer) map.get( "two" );

        // And print the wrapped int
        terminal.println( "The wrapped int stored for \"two\" is "
                          + wrappedInt);


        // The set of keys.
        Set keys = map.keySet();
        // The iterator over this "set" of keys will return
        // the keys in key-order.
        terminal.println( "\nIterate over keys, get each value." );
        terminal.println( "Note that key order is aphabetical:" );
        Iterator keysIterator = keys.iterator();
        while ( keysIterator.hasNext() ) {
            String key = (String) keysIterator.next();
            terminal.println( "The value for key " + key + " is "
                              + ((Integer) map.get( key)) );
        }

        // Iterate over the collection of values;
        // notice the order is the same (ie the key-order).
        terminal.println( "\nIterate over the values:" );
        Iterator valuesIterator = map.values().iterator();
        while ( valuesIterator.hasNext() ) {
            terminal.println( ((Integer) valuesIterator.next()));
        }

        // The set of Map.Entry objects (key-value pairs);
        // Map.Entry is an inner class of Map.

        // Iterate over the entries.
        terminal.println( "\nIterate over the key-value pairs:" );
        Iterator entriesIterator = map.entrySet().iterator();
        while ( entriesIterator.hasNext() ) {
            Map.Entry entry = (Map.Entry) entriesIterator.next();
            terminal.println( "The value for the entry with key "
                      + entry.getKey() + " is "
                      + ((Integer) entry.getValue()));
        }

        // how a TreeMap represents itself as a String:
        terminal.println(
            "\nHow a TreeMap represents itself as a String:");
        terminal.println(map.toString());
        terminal.println();

        // We can overwrite the value stored under a key
        terminal.println(
```

```
113             "Store a different value at key \"two\"");
114         map.put("two", new Integer(2222));
115         terminal.println(map.toString());
116         terminal.println();
117
118         // We can store the same value under two keys
119         terminal.println(
120             "Store map.get( \"one\" ) at key \"two\"");
121         map.put("two", map.get( "one" ) );
122         terminal.println(map.toString());
123         terminal.println();
124
125         // And keys don't necessarily have to be Strings;
126         // Here's a TreeMap mapping Integers to strings.
127         terminal.println(
128             "A TreeMap with Integer keys mapping to String values");
129         map = new TreeMap();
130         map.put( new Integer( 1 ), "I" );
131         map.put( new Integer( 2 ), "II" );
132         map.put( new Integer( 3 ), "III" );
133         terminal.println(map.toString());
134         terminal.println();
135     }
136 }
```