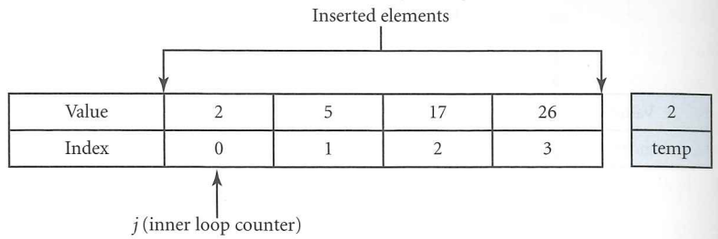

COMMON ERROR TRAP

When looping through an array, be careful not to access an element outside the bounds of the array. Your code will compile, but will generate an *ArrayIndexOutOfBoundsException* at run time.

The outer loop counter (i) is incremented, and its value is 4, which causes the outer loop to terminate. All the elements have been inserted; the array is now sorted.

Example 8.18 shows our *Sorter* class with the Insertion Sort algorithm implemented in lines 43–63.

```
1 /* Sort Utility Class
2    Anderson, Franceschi
3 */
4
5 public class Sorter
6 {
7   /**  Performs a Selection Sort on
8    *       an integer array
9    *    @param the array to sort
10  */
11  public static void selectionSort( int [ ] array )
12  {
13     int temp; // temporary location for swap
14     int max;  // index of maximum value in subarray
15
16     for ( int i = 0; i < array.length; i++ )
17     {
18       // find index of largest value in subarray
19       max = indexOfLargestElement( array, array.length - i );
20
21       // swap array[max] and array[array.length - i - 1]
22       temp = array[max];
23       array[max] = array[array.length - i - 1];
24       array[array.length - i - 1] = temp;
25     }
26   }
27
```



```
28   /**  Finds index of largest element
29   *    @param     size  the size of the subarray
30   *    @return    the index of the largest element in the subarray
31   */
32   private static int indexOfLargestElement( int [ ] array, int size )
33   {
34      int index = 0;
35      for( int i = 1; i < size; i++ )
36      {
37          if ( array[i] > array[index] )
38             index = i;
39      }
40      return index;
41   }
42
43   /**  Performs an Insertion Sort on an integer array
44   *    @param array   array to sort
45   */
46   public static void insertionSort( int [ ] array )
47   {
48     int j, temp;
49
50     for ( int i = 1; i < array.length; i++ )
51     {
52        j = i;
53        temp = array[i];
54
55        while ( j != 0 && array[j - 1] > temp )
56        {
57           array[j] = array[j - 1];
58           j--;
59        }
60
61        array[j] = temp;
62     }
63   }
64 }
```

EXAMPLE 8.18 Sorter Class with Insertion Sort

Example 8.19 shows a client program that instantiates an integer array, fills it with random values, and then prints the array before and after performing the Insertion Sort. Figure 8.24 shows a sample run, using the Insertion Sort algorithm to sort an array of integers.