

# Polymorphic Sorting

The file *Sorting.java* contains the *Sorting* class from Listing 9.9 in the text. This class implements both the selection sort and the insertion sort algorithms for sorting any array of *Comparable* objects in ascending order. In this exercise, you will use the *Sorting* class to sort several different types of objects.

1. The file *Numbers.java* reads in an array of integers, invokes the selection sort algorithm to sort them, and then prints the sorted array. Save *Sorting.java* and *Numbers.java* to your directory. *Numbers.java* won't compile in its current form. Study it to see if you can figure out why.
2. Try to compile *Numbers.java* and see what the error message is. The problem involves the difference between primitive data and objects. Change the program so it will work correctly (note: you don't need to make many changes - the autoboxing feature of Java 1.5 will take care of most conversions from *int* to *Integer*).
3. Write a program *Strings.java*, similar to *Numbers.java*, that reads in an array of *String* objects and sorts them. You may just copy and edit *Numbers.java*.
4. Modify the *insertionSort* algorithm so that it sorts in descending order rather than ascending order. Change *Numbers.java* and *Strings.java* to call *insertionSort* rather than *selectionSort*. Run both to make sure the sorting is correct.
5. The file *Salesperson.java* partially defines a class that represents a sales person. This is very similar to the *Contact* class in Listing 9.10. However, a sales person has a first name, last name, and a total number of sales (an *int*) rather than a first name, last name, and phone number. Complete the *compareTo* method in the *Salesperson* class. The comparison should be based on total sales; that is, return a negative number if the executing object has total sales less than the other object and return a positive number if the sales are greater. Use the name of the sales person to break a tie (alphabetical order).
6. The file *WeeklySales.java* contains a driver for testing the *compareTo* method and the sorting (this is similar to Listing 9.8 in the text). Compile and run it. Make sure your *compareTo* method is correct. The sales staff should be listed in order of sales from most to least with the four people having the same number of sales in reverse alphabetical order.
7. OPTIONAL: Modify *WeeklySales.java* so the salespeople are read in rather than hardcoded in the program.

```
//*****
//  Sorting.java          Author: Lewis/Loftus
//
//  Demonstrates the selection sort and insertion sort algorithms.
//*****

public class Sorting
{
    //-----
    //  Sorts the specified array of objects using the selection
    //  sort algorithm.
    //-----
    public static void selectionSort (Comparable[] list)
    {
        int min;
        Comparable temp;

        for (int index = 0; index < list.length-1; index++)
        {
            min = index;
            for (int scan = index+1; scan < list.length; scan++)
                if (list[scan].compareTo(list[min]) < 0)
```

```

        min = scan;

        // Swap the values
        temp = list[min];
        list[min] = list[index];
        list[index] = temp;
    }
}

//-----
//  Sorts the specified array of objects using the insertion
//  sort algorithm.
//-----
public static void insertionSort (Comparable[] list)
{
    for (int index = 1; index < list.length; index++)
    {
        Comparable key = list[index];
        int position = index;

        // Shift larger values to the right
        while (position > 0 && key.compareTo(list[position-1]) < 0)
        {
            list[position] = list[position-1];
            position--;
        }

        list[position] = key;
    }
}
}

// *****
//  Numbers.java
//
//  Demonstrates selectionSort on an array of integers.
//  *****

import java.util.Scanner;

public class Numbers
{
    // -----
    //  Reads in an array of integers, sorts them,
    //  then prints them in sorted order.
    //  -----
    public static void main (String[] args)
    {
        int[] intList;
        int size;

        Scanner scan = new Scanner(System.in);

        System.out.print ("\nHow many integers do you want to sort? ");
        size = scan.nextInt();
        intList = new int[size];

        System.out.println ("\nEnter the numbers...");
        for (int i = 0; i < size; i++)
            intList[i] = scan.nextInt();

        Sorting.selectionSort(intList);
    }
}

```

```

        System.out.println ("\nYour numbers in sorted order...");
        for (int i = 0; i < size; i++)
            System.out.print(intList[i] + " ");
        System.out.println ();
    }
}

// *****
// Salesperson.java
//
// Represents a sales person who has a first name, last
// name, and total number of sales.
// *****

public class Salesperson implements Comparable
{
    private String firstName, lastName;
    private int totalSales;

    //-----
    // Constructor: Sets up the sales person object with
    // the given data.
    //-----
    public Salesperson (String first, String last, int sales)
    {
        firstName = first;
        lastName = last;
        totalSales = sales;
    }

    //-----
    // Returns the sales person as a string.
    //-----
    public String toString()
    {
        return lastName + ", " + firstName + ": \t" + totalSales;
    }

    //-----
    // Returns true if the sales people have
    // the same name.
    //-----
    public boolean equals (Object other)
    {
        return (lastName.equals(((Salesperson)other).getLastName()) &&
            firstName.equals(((Salesperson)other).getFirstName()));
    }

    //-----
    // Order is based on total sales with the name
    // (last, then first) breaking a tie.
    //-----
    public int compareTo(Object other)
    {
        int result;

        return result;
    }

    //-----
    // First name accessor.
    //-----
    public String getFirstName()
    {

```

```

        return firstName;
    }

    //-----
    //  Last name accessor.
    //-----
    public String getLastName()
    {
        return lastName;
    }

    //-----
    //  Total sales accessor.
    //-----
    public int getSales()
    {
        return totalSales;
    }
}

// *****
//  WeeklySales.java
//
//  Sorts the sales staff in descending order by sales.
//  *****

public class WeeklySales
{
    public static void main(String[] args)
    {
        Salesperson[] salesStaff = new Salesperson[10];

        salesStaff[0] = new Salesperson("Jane", "Jones", 3000);
        salesStaff[1] = new Salesperson("Daffy", "Duck", 4935);
        salesStaff[2] = new Salesperson("James", "Jones", 3000);
        salesStaff[3] = new Salesperson("Dick", "Walter", 2800);
        salesStaff[4] = new Salesperson("Don", "Trump", 1570);
        salesStaff[5] = new Salesperson("Jane", "Black", 3000);
        salesStaff[6] = new Salesperson("Harry", "Taylor", 7300);
        salesStaff[7] = new Salesperson("Andy", "Adams", 5000);
        salesStaff[8] = new Salesperson("Jim", "Doe", 2850);
        salesStaff[9] = new Salesperson("Walt", "Smith", 3000);

        Sorting.insertionSort(salesStaff);

        System.out.println ("\nRanking of Sales for the Week\n");

        for (Salesperson s : salesStaff)
            System.out.println (s);
    }
}

```