

Nesneye Yönelik Programlama

CENG 102 LAB 10

28 Nisan 2019

1 AMAÇ

- Polymorphismi öğrenme,
- Interface kullanımı öğrenme,

2 YAPILACAKLAR

Sorting.java dosyası selection sort ve insertion sortu, bir liteyi artan sırada sıralayacak şekilde implement etmektedir. Bu lab'da Sorting sınıfını çeşitli nesneleri sıralamak için kullanacaksınız.

- 1. Numbers.java dosyası bir tamsayı arrayi oluşturup selection sort algoritmasını çağırır. Sorting.java ve Numbers.java'yı açtığınız projenin içerisine ekleyiniz. Numbers.java bu haliyle derlenemeyecektir. Bunun neden böyle olduğunu bularak düzeltiniz.
- 2. Numbers.java'ya benzer şekilde Strings.java programı yazınız. String array oluşturmalı ve onları sıralamalı.
- 3. insertionSort algoritmasını sırayı azalan şekilde yapacak şekilde değiştiriniz. Numbers.java ve Strings.java'yı insertionSortu çağırarak şekilde güncelleyiniz. İkisini de çalıştırarak yeni sıralamanızı kontrol ediniz.
- 4. Salesperson.java bir satış elemanı için oluşturulmuş bir sınıftır. Bu sınıftaki compareTo metodunu tamamlayın. Kıyaslama toplam satışlar üzerine dayanmalıdır. Çalışan nesne karşılaştırma için parametre ile gönderilenden daha az satış yaptıysa -1, daha yüksek satış yaptıysa +1, eşit satış yapıldıysa kıyaslamaları ismine göre yapmalıdır.
- 5. WeeklySales sınıfı compareTo metodu için bir sürücü niteliğindedir. Bu dosyayı derleyip çalıştırın. compareTo metodunun doğru çalıştığından emin olunuz.

Listing 1: Sorting.java

```
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;
    }
}

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;
    }
}
}

```

Listing 2: Numbers.java

```

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

```

Listing 3: Salesperson.java

```

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
    // (last, then first) breaking a tie. //-
    public int compareTo(Object other)
    {
        int result;
        return result;
    }
    public String getFirstName()
    {
        return firstName;
    }
    //-----
    // Last name accessor.
    //-----
    public String getLastName()
    {
        return lastName;
    }
    //-----
    // Total sales accessor.
    //-----
    public int getSales()
    {
        return totalSales;
    }
}

```

Listing 4: WeeklySales.java

```
public class WeeklySales
{
    public static void main(String[] args)
    {
        Salesperson[] salesStaff = new Salesperson[10];
        salesStaff[0] = new Salesperson("Jane", "Jones", 3000); salesStaff[1] = new Sa
        Sorting.insertionSort(salesStaff);
        System.out.println ("\nRanking_of_Sales_for_the_Week\n");
        for (Salesperson s : salesStaff)
            System.out.println (s);
    } }
```