CZ2002 LAB 4

OBJ ORIENTED DES & PROG

SS5 GROUP 6

Zou Zeren U2022422H

Contents

1 Numbers.java Issue **2**

2 Numbers.java Errors **2**

3 String.java **3**

4 Decending Insertion Sort **3**

5 SalesPerson Class **5**

6 WeeklySales **6**

7 Calculate Surface Area of a Figure **6**

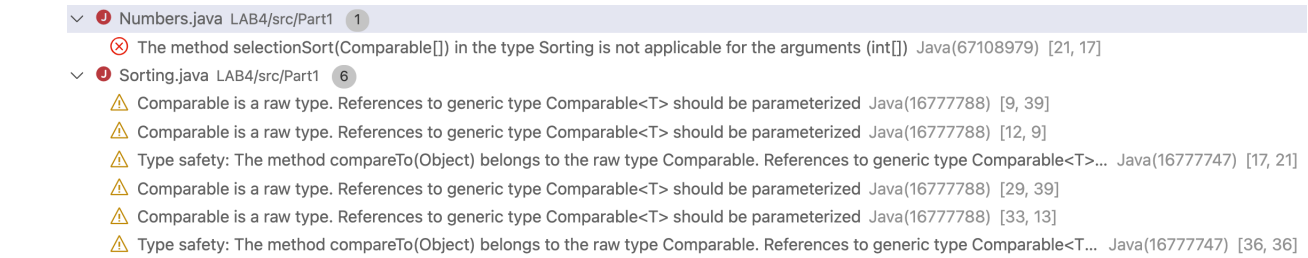
1 Numbers.java Issue

The file Numbers.java (in attachment) 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.

Numbers.java have to use functions from the Sorting.java.
Int is a primitive type, and therefore cannot implement any interface. That's why int[] cannot be passed to a method that expects Comparable[]. To overcome this error, change intList to be an array of Integer (i.e Integer[]), since Integer implements Comparable.

2 Numbers.java Errors

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 (or higher) will take care of most conversions from int to Integer). You are to do research in the internet and understand better autoboxing.




```
Run | Debug
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 ();
}
}
```

```
Run | Debug
public static void main (String[] args)
{
    Integer[] 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 Integer[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 ();
}
}
```

```
How many integers do you want to sort? 4

Enter the numbers...
3
4
2
1

Your numbers in sorted order...
1 2 3 4
anthonyzeren@Zous-MacBook-Pro CZ2002-OODP-NTU %
```

3 String.java

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.

```java
package Part1;

import java.util.Scanner;
public class Numbers
{
    // -----------------------------------------
// Reads in an array of integers, sorts them,
// then prints them in sorted order.
// -----------------------------------------
    Run | Debug
    public static void main (String[] args)
    {
        Integer[] 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 Integer[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 ();
        scan.close();
    }
}
```

```
How many string do you want to sort? 4

Enter the strings...
apple
pear
water melon
banana

Your strings in sorted order...
 apple banana pear water melon
```

4 Decending Insertion Sort

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. Decending Insertion Sort

```java
    //-----------------------------------------------------------------
    // 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)
            while (position > 0 && key.compareTo(list[position-1]) > 0)
            {
                list[position] = list[position-1];
                position--;
            }
            list[position] = key;
        }
    }
}
```

Tested on Numbers

```java
package part1;

import java.util.Scanner;
public class Numbers
{
    // -----------------------------------------
// Reads in an array of integers, sorts them,
// then prints them in sorted order.
// -----------------------------------------
    Run | Debug
    public static void main (String[] args)
    {
        Integer[] 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 Integer[size];
        System.out.println ("\nEnter the numbers...");
        for (int i = 0; i < size; i++)
            intList[i] = scan.nextInt();
       // Sorting.selectionSort(intList);
        Sorting.insertionSort(intList);
        System.out.println ("\nYour numbers in sorted order...");
        for (int i = 0; i < size; i++)
            System.out.print(intList[i] + " ");
        System.out.println ();
        scan.close();
    }
}
```

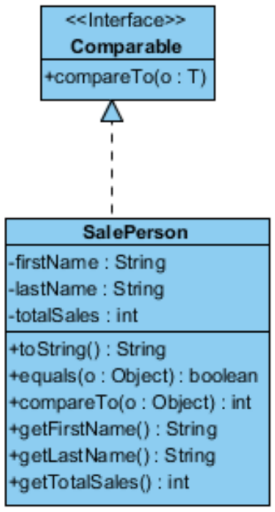
```
How many integers do you want to sort? 3

Enter the numbers...
2
1
3

Your numbers in sorted order...
3 2 1
```

Tested on Strings

```java
package part1;

import java.util.Scanner;
public class Strings {
    // --------------------------------------------
// Reads in an array of integers, sorts them,
// then prints them in sorted order.
// --------------------------------------------
Run | Debug
public static void main (String[] args)
{
    String[] strList;
    int size;
    Scanner scan = new Scanner(System.in);
    System.out.print ("\nHow many string do you want to sort? ");
    size = scan.nextInt()+1;
    strList = new String[size];
    System.out.println ("\nEnter the strings...");
    for (int i = 0; i < size; i++)
        strList[i] = scan.nextLine();
    //Sorting.selectionSort(strList);
    Sorting.insertionSort(strList);
    System.out.println ("\nYour strings in sorted order...");
    for (int i = 0; i < size; i++)
        System.out.print(strList[i] + " ");
    System.out.println ();
    scan.close();
}
}
```

```
How many string do you want to sort? 3

Enter the strings...
apple
banana
cherry

Your strings in sorted order...
cherry banana apple
```

5 SalesPerson Class



5. The class diagram on the right defines the SalePerson class that represents a sale person. The sale person has a first name, last name, and a total number of sales (an int).

- The *toString* method will return the name of the sale person and total sales in the formal :
  <lastName> , <firstName> : <totalSales>
- The *equals* method will check whether the first and last names of Object are the same as the current sale person.
- The *compareTo* method make the comparison 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's last name to break a tie (in ascending alphabetical order).***

- ***Create and Write the SalePerson class***

SalePerson.java M

LAB4 > src > sales > SalePerson.java > SalePerson > SalePerson(String, String, int)

```java
package sales;

public class SalePerson implements Comparable<SalePerson> {
  private String firstName;
  private String lastName;
  private int totalSales;

  public SalePerson(String firstName, String lastName, int totalSales) {
    this.firstName = firstName;
    this.lastName = lastName;
    this.totalSales = totalSales;
  }

  public String getFirstName() {
    return firstName;
  }

  public String getLastName() {
    return lastName;
  }

  public int getTotalSales() {
    return totalSales;
  }
```

```java
@Override
public int compareTo(SalePerson p) {
  if (this.totalSales < p.getTotalSales())
    return -1;
  else if (this.totalSales > p.getTotalSales())
    return 1;
  else {
    if (this.lastName.compareTo(p.getLastName()) > 0)
      return -1;
    else
      return 1;
  }
}
```

```java
@java.lang.Override
public java.lang.String toString() {
  return lastName + ", " + firstName + ":" + totalSales;
}

public boolean equals(Object object) {
  if (this == object)
    return true;
  if (object == null || getClass() != object.getClass())
    return false;
  if (!super.equals(object))
    return false;
  SalePerson that = (SalePerson) object;
  return totalSales == that.totalSales && java.util.Objects.equals(firstName, that.firstName)
      && java.util.Objects.equals(lastName, that.lastName);
}
}
```

6 WeeklySales

The file WeeklySales.java (in attachment) contains a driver for testing the compareTo method and the sorting . Compile and run it. Make sure your compareTo method is correct. The sales staff should be listed in the order of sales from most to least. If the sale staffs have the same number of sales, they are listed in ascending alphabetical order of their last names.'

WeeklySales.java M

LAB4 > src > sales > WeeklySales.java > WeeklySales > main(String[])

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

public class WeeklySales {
    Run | Debug
    public static void main(String[] args) {
        SalePerson[] salesStaff = new SalePerson[10];
        salesStaff[0] = new SalePerson("Jane", "A", 3000);
        salesStaff[1] = new SalePerson("Daffy", "Duck", 4935);
        salesStaff[2] = new SalePerson("James", "B", 3000);
        salesStaff[3] = new SalePerson("Dick", "Lemon", 2800);
        salesStaff[4] = new SalePerson("Don", "Salt", 1570);
        salesStaff[5] = new SalePerson("Jane", "C", 3000);
        salesStaff[6] = new SalePerson("Harry", "Tea", 7300);
        salesStaff[7] = new SalePerson("Andy", "Carrot", 5000);
        salesStaff[8] = new SalePerson("Jim", "Donnut", 2850);
        salesStaff[9] = new SalePerson("Walt", "D", 3000);

        Sorting.insertionSort(salesStaff);

        System.out.println("\nRanking of Sales for the Week\n");
        for (SalePerson s : salesStaff) System.out.println(s);
    }
}
```

```
Ranking of Sales for the Week

Tea, Harry:7300
Carrot, Andy:5000
Duck, Daffy:4935
A, Jane:3000
B, James:3000
C, Jane:3000
D, Walt:3000
Donnut, Jim:2850
Lemon, Dick:2800
Salt, Don:1570
anthonyzeren@Zous-MacBook-Pro CZ2002-OODP-NTU %
```

7 Calculate Surface Area of a Figure

By using the concepts of inheritance and polymorphism, you are required to design a program that calculate the total surface area of a figure. The following are the requirements an constraints :

You should have a Class/Interface called Shape and decide its appropriate attributes and behaviours

You should have basic shapes like Square, Rectangle, Circle and Triangle.

The program will request the user to :

- *enter the total number of shapes*
- *choose the shape and enter the required dimension/s for the selected shape*
- *choose the type of calculation (for now, we will just calculate Area, with future plan to calculate Volume as well).*

The calculation/s should be done upon user's request and NOT when dimensions are entered.

1. *For a start, use the 2-D figure on the right to verify your program. The figure consists of a Circle (radius=10), a Triangle (height=25, base =20) and a Rectangle (length=50, breadth = 20) . Calculate the total area of the 2- D figure. (You will create an Application class Shape2DApp.java for this purpose)*

2. *We will now expand and extend your design to cater to 3-D figures. Imagine the figure on the right is turn into a 3-D figure – Circle becomes Sphere, Triangle becomes a square-based Pyramid and the Rectangle is a cubiod. Calculate the total surface area of the 3- D figure. [Note : You need to think whether 'is a' or 'has a' relationship is more appropriate and relevant for between 2D and 3D shapes. (You will create an Application class Shape3DApp.java for this purpose)*

3. *We will include more Shapes. The square-based Pyramid will be replaced with a Cone and the Cubiod is replaced with a Cylinder. Calculate the total surface area of the new 3- D figure. (You will reuse the Application class Shape3DApp.java with appropriate selection)*