CS 2401 – Elementary data structures and algorithms
Fall 2022: Lab 8

Due Date: Friday, November 4, 2022 – end of the day.

Objective: The goal of this assignment is to practice **sorting** array of objects. You will implement bubble sort, selection sort, and merge sort to sort an array of Box objects in ascending order of their volumes.

Background: El Paso Packaging and Supply Co. has contracted you to create a software to summarize their inventory. The client saves dimensions of boxes in the inventory in a text file. It is possible to write a program that will read from the file and create a summary of the inventory. Each line in the file contains the **width, height, and length** of a box. The dimensions are separated by spaces. A sample file is shown below.

```
67.8 41.5 56.1
3.2 2.5 9.1
5.0 4.8 2.5
101.2 32.5 105.0
20.5 4.5 80.75
15.5 44.2 20.30
1.0 1.0 1.0
14.23 7.45 10.5
6.0 5.0 10.2
7.5 7.5 7.5
```

Assignment: Your program should assume that the content is written in a file named input.txt. Each box must be considered an object. To achieve this, you must write a class named `Box`. The Box class is provided below and you are not allowed to change the **Box** class. The Box.java file is provided with the assignment too.

```java
public class Box {
  private double width, height, length;

  Box(double w, double h, double l){
    width=w;
    height=h;
    length=l;
  }

  private double getVolume(){
    return width*height*length;
  }

  public int compareTo(Box o){
    double myVol = this.getVolume();
    double thatVol = o.getVolume();
    if (myVol>thatVol)
      return 1;
    else if (myVol<thatVol)
      return -1;
    else
      return 0;
  }

  public String toString(){
    return "Width: "+width+
           "\theight: "+height+
            "\tlength: "+length+
            "\tVolume: "+getVolume();
  }

}
```

The partially written program file (the Java file that contains the main method) is provided in a file named Lab8.java. **The instructions are provided as comments in the code. The** Lab8.java file with the following content is provided with the assignment too.

```
import java.io.File;
import java.util.Scanner;

public class Lab8 {
  public static void main(String[] args){
    String fileName = "input.txt";
    Box[] allBoxes = getBoxArrayFromDataFile(fileName);

    if (allBoxes!=null){
      System.out.println("Number of boxes in the array: "+
            allBoxes.length);
    }else{
      System.out.println("No array constructed. Array is null. ");
    }

    System.out.println("The boxes are as follows:");
    displayAllBoxes(allBoxes);
    System.out.println("Sorting the array using bubbleSort");

    long start = System.nanoTime();
    bubbleSort(allBoxes);
    long end = System.nanoTime();
    long bubbleSortTime=(end-start);

    System.out.println("The array after bubble sort:");
    displayAllBoxes(allBoxes);

    System.out.println("Re-constructing the array from the input file.");
    allBoxes = getBoxArrayFromDataFile(fileName);
    if (allBoxes!=null){
      System.out.println("Number of boxes in the array: "+
            allBoxes.length);
    }else{
      System.out.println("No array constructed. Array is null. ");
    }
    System.out.println("The boxes after re-reading the input file");
    displayAllBoxes(allBoxes);
    System.out.println("Sorting the array using selectionSort");
    start = System.nanoTime();
    selectionSort(allBoxes);
    end = System.nanoTime();
    long selectionSortTime=(end-start);

    System.out.println("The array after selection sort:");
    displayAllBoxes(allBoxes);

    System.out.println("Re-constructing the array from the input file.");
    allBoxes = getBoxArrayFromDataFile(fileName);
    if (allBoxes!=null){
      System.out.println("Number of boxes in the array: "+
            allBoxes.length);
    }else{
      System.out.println("No array constructed. Array is null. ");
    }
    System.out.println("The boxes after re-reading the input file");
    displayAllBoxes(allBoxes);
    System.out.println("Sorting the array using mergeSort");
    start = System.nanoTime();
    mergeSort(allBoxes);
    end = System.nanoTime();
    long mergeSortTime=(end-start);

    System.out.println("The array after merge sort:");
    displayAllBoxes(allBoxes);

    System.out.println("********* Runtime summary: ***********");
    System.out.println("Time taken by bubble sort: "+bubbleSortTime+" ns");
    System.out.println("Time taken by selection sort: "+selectionSortTime+" ns");
    System.out.println("Time taken by merge sort: "+mergeSortTime+" ns");
  }

  /**
   * Change the body of this method to arrange the Box
   * objects in the array parameter in ascending order of
   * their volumes.
   * The method must use bubble sort.
   * @param theBoxes
   */
  static void bubbleSort(Box[] theBoxes) {

    // You are not allowed to change the header.
    // Change the body of this method.

    System.out.println("I have not yet implemented bubble sort");

  }

  /**
   * Change the body of this method to arrange the Box
   * objects in the array parameter in ascending order of
   * their volumes.
   * The method must use selection sort.
   * @param theBoxes
   */
  static void selectionSort(Box[] theBoxes) {
    // You are not allowed to change the header.
    // Change the body of this method.

    System.out.println("I have not yet implemented selection sort");

  }

  /**
   * Change the body of this method to arrange the Box
   * objects in the array parameter in ascending order of
   * their volumes.
   * The method must use merge sort.
   * @param theBoxes
   */
  static void mergeSort(Box[] theBoxes) {
    // You are not allowed to change the header.
    // Change the body of this method.

    System.out.println("I have not yet implemented merge sort");

  }


  /**
   * Display width, length, height, volume of
   * each Box object in a sequence they appear
   * in the array of the parameter.
   * @param theBoxes
   */
  static void displayAllBoxes(Box[] theBoxes){
    // You are not allowed to change the header.
    // Change the body of this method.

    System.out.println("I have not yet implemented the display method.");
  }


  static Box[] getBoxArrayFromDataFile (String fileName){
    // You are not allowed to change the header.
    // Change the body of this method.

    System.out.println("I have not yet implemented the method to \n construct the array
from file input.");

    return null;
  }

}
```

Requirements: Your task is to change the code in Lab8.java as instructed in the comments to receive the necessary output. On the way, you must follow the following requirements**.**

1. You must NOT change the main method.
2. Do not use any package other than the ones already imported in Lab8.java.
3. Do NOT change any of the method headers provided in Lab8.java.
4. Feel free to write as many new methods as you need in Lab8.java.
5. You must NOT change Box.java at all.

The output of a correctly written code for the input provided earlier is below.

```
Number of boxes in the array: 10
The boxes are as follows:
(The box dimensions are displayed below)
---------------------------------
Width: 67.8     height: 41.5     length: 56.1     Volume: 157848.57
Width: 3.2      height: 2.5      length: 9.1      Volume: 72.8
Width: 5.0      height: 4.8      length: 2.5      Volume: 60.0
Width: 101.2    height: 32.5     length: 105.0    Volume: 345345.0
Width: 20.5     height: 4.5      length: 80.75    Volume: 7449.1875
Width: 15.5     height: 44.2     length: 20.3     Volume: 13907.53
```

```
Width: 1.0      height: 1.0     length: 1.0     Volume: 1.0
Width: 14.23    height: 7.45    length: 10.5    Volume: 1113.14175
Width: 6.0      height: 5.0     length: 10.2    Volume: 306.0
Width: 7.5      height: 7.5     length: 7.5     Volume: 421.875
---------------------------------

Sorting the array using bubbleSort
The array after bubble sort:
(The box dimensions are displayed below)
---------------------------------
Width: 1.0      height: 1.0     length: 1.0     Volume: 1.0
Width: 5.0      height: 4.8     length: 2.5     Volume: 60.0
Width: 3.2      height: 2.5     length: 9.1     Volume: 72.8
Width: 6.0      height: 5.0     length: 10.2    Volume: 306.0
Width: 7.5      height: 7.5     length: 7.5     Volume: 421.875
Width: 14.23    height: 7.45    length: 10.5    Volume: 1113.14175
Width: 20.5     height: 4.5     length: 80.75   Volume: 7449.1875
Width: 15.5     height: 44.2    length: 20.3    Volume: 13907.53
Width: 67.8     height: 41.5    length: 56.1    Volume: 157848.57
Width: 101.2    height: 32.5    length: 105.0   Volume: 345345.0
---------------------------------

Re-constructing the array from the input file.
Number of boxes in the array: 10
The boxes after re-reading the input file
(The box dimensions are displayed below)
---------------------------------
Width: 67.8     height: 41.5    length: 56.1    Volume: 157848.57
Width: 3.2      height: 2.5     length: 9.1     Volume: 72.8
Width: 5.0      height: 4.8     length: 2.5     Volume: 60.0
Width: 101.2    height: 32.5    length: 105.0   Volume: 345345.0
Width: 20.5     height: 4.5     length: 80.75   Volume: 7449.1875
Width: 15.5     height: 44.2    length: 20.3    Volume: 13907.53
Width: 1.0      height: 1.0     length: 1.0     Volume: 1.0
Width: 14.23    height: 7.45    length: 10.5    Volume: 1113.14175
Width: 6.0      height: 5.0     length: 10.2    Volume: 306.0
Width: 7.5      height: 7.5     length: 7.5     Volume: 421.875
---------------------------------

Sorting the array using selectionSort
The array after selection sort:
(The box dimensions are displayed below)
---------------------------------
Width: 1.0      height: 1.0     length: 1.0     Volume: 1.0
Width: 5.0      height: 4.8     length: 2.5     Volume: 60.0
Width: 3.2      height: 2.5     length: 9.1     Volume: 72.8
Width: 6.0      height: 5.0     length: 10.2    Volume: 306.0
Width: 7.5      height: 7.5     length: 7.5     Volume: 421.875
Width: 14.23    height: 7.45    length: 10.5    Volume: 1113.14175
Width: 20.5     height: 4.5     length: 80.75   Volume: 7449.1875
Width: 15.5     height: 44.2    length: 20.3    Volume: 13907.53
Width: 67.8     height: 41.5    length: 56.1    Volume: 157848.57
Width: 101.2    height: 32.5    length: 105.0   Volume: 345345.0
---------------------------------

Re-constructing the array from the input file.
Number of boxes in the array: 10
```

```
The boxes after re-reading the input file
(The box dimensions are displayed below)
---------------------------------
Width: 67.8     height: 41.5    length: 56.1    Volume: 157848.57
Width: 3.2      height: 2.5     length: 9.1     Volume: 72.8
Width: 5.0      height: 4.8     length: 2.5     Volume: 60.0
Width: 101.2    height: 32.5    length: 105.0   Volume: 345345.0
Width: 20.5     height: 4.5     length: 80.75   Volume: 7449.1875
Width: 15.5     height: 44.2    length: 20.3    Volume: 13907.53
Width: 1.0      height: 1.0     length: 1.0     Volume: 1.0
Width: 14.23    height: 7.45    length: 10.5    Volume: 1113.14175
Width: 6.0      height: 5.0     length: 10.2    Volume: 306.0
Width: 7.5      height: 7.5     length: 7.5     Volume: 421.875
---------------------------------

Sorting the array using mergeSort
The array after merge sort:
(The box dimensions are displayed below)
---------------------------------
Width: 1.0      height: 1.0     length: 1.0     Volume: 1.0
Width: 5.0      height: 4.8     length: 2.5     Volume: 60.0
Width: 3.2      height: 2.5     length: 9.1     Volume: 72.8
Width: 6.0      height: 5.0     length: 10.2    Volume: 306.0
Width: 7.5      height: 7.5     length: 7.5     Volume: 421.875
Width: 14.23    height: 7.45    length: 10.5    Volume: 1113.14175
Width: 20.5     height: 4.5     length: 80.75   Volume: 7449.1875
Width: 15.5     height: 44.2    length: 20.3    Volume: 13907.53
Width: 67.8     height: 41.5    length: 56.1    Volume: 157848.57
Width: 101.2    height: 32.5    length: 105.0   Volume: 345345.0
---------------------------------

********* Runtime summary: ***********
Time taken by bubble sort: 10816 ns
Time taken by selection sort: 31059 ns
Time taken by merge sort: 17667 ns
```

Comments:

The Runtime summary will vary between computers. For small number of Box information in input.txt, it is likely that merge sort will take longer time. If you put a few hundred (or, maybe thousands) box information in the input.txt file, then merge sort's runtime will be significantly lower than bubble sort and selection sort. Also, you will probably observe that selection sort is taking smaller amount of time than the bubble sort.

Deliverables: You are expected to submit only one Java files (`Lab8.java`) via Blackboard. Please do not submit the input.txt file or the Box.java file.

Grading Criteria:

- [20 points] The Program **compiles and runs**.
- [10 points] The program is **indented** and **documented** properly.
- [10 points] The program uses the correct **variable types** and **names**.

To Chegg: Please do not provide solution if this document is uploaded. Please remove this document from Chegg, if uploaded.

    - [60 points] The sorting algorithms are implemented correctly.

- Late submission: [**-10**] points for every 24 hours after the deadline.

If you need any clarification, please ask your TA for further details.