**CS 2163 Java ---**Homework 6

**General description for homework 6:** this homework is to be finished in Eclipse, NOT Greenfoot.

In this homework, your program will create an array and then fill it with random values, and then print it out. Then the program will invoke a sort algorithm to sort the array, and then print it to verify that it has been sorted.

**Here are the detail descriptions for homework 6:**

First create a new Eclipse Java project named ***JohnDoeHw6***, and then inside this project, create a new class named ***JohnDoeHw6.java***. Replace ***JohnDoe*** with your first and last name.

At the beginning of ***JohnDoeHw6.java***, add an appropriate program header similar to line 1 ~ 11 of file “CorrectSequence.java” (this file can be found in zip file “UML-classDiagram.zip” in Moodle folder “chap 4”).

In its ***main*** method, provide a line like this as the first line inside the main method, which outputs your name, course meeting time, class and semester info:

System.out.println("Spring 2017, Java, Mon 5:30pm, John Doe, hw6");

No GUI is needed in this homework. No console input is needed, thus no need to import Scanner class.

In the main method, then follow the STEPS below:

1. Declare an **array** of **double** type, with **size 10**.
2. Use a regular for loop to generate **10 random** double type variables and fill them into the array. The range of the random variables is between **0 and 100**, **including 0** and **excluding 100**. Use **method Math.random()**.
3. Invoke a method named ***printArray*** in the same class, to print all elements of this array in one line, with a space separating adjacent array elements, and then output an end-of-line symbol at the end. You should use **printf** method to specify the output to have **only 2 digits after the decimal point** for each double type variable. This method ***printArray*** takes only **one parameter**, which is a **double** type **one-dimensional array**, and the method **return type should be void**. Use a for-each loop in this method to navigate through the array. Please refer to file “LinearSearch.java”, line 44~46 on how to implement a for-each loop.
4. Invoke a method name ***sort*** in the same class, to sort this array in **DECREASING** order. This method takes only **one parameter**, which is a **double type array**, and its **return type should be void**. You must **choose one of these three algorithms**: selection, insertion, or bubble sort. The source code of these three algorithms locates in zip file “search-and-sorting.zip”. You need to ***modify the source code*** of the sorting algorithm you choose, and make it usable in this homework.

**Note**: you **CANNOT** invoke any existing sorting method from any Java library class, such as the sort method in java class Arrays. The purpose of this homework is to let you modify the sorting algorithm source code thus get a better understanding of how a sorting algorithm works internally. But if you just call a sort method in the Java jdk library, you lost the whole purpose of this homework, and you will receive zero point.

1. Invoke method ***printArray*** again to ***print the array to the screen***, to verify that it has been sorted.

**Implementation requirements:**

* In class JohnDoeHw6.java, you should have three methods. One is the ***main*** method, the second method is ***printArray***. , and the third is ***sort***. Do not put more classes or more methods in the homework, o/w, you are in the wrong track.
* **Inside the main method**, you invoke method ***printArray*** and ***sort***.
* The signatures of method ***printArray*** and ***sort*** have already been defined in step 3 and 4 above, and you **cannot** change the signature nor the return type of this method
* Because the required sort order in this homework is decreasing order, you need to make a modification on the sorting algorithm provided in the **file “search-and-sorting.zip” (they are all increasing sort)**. Hint: just change the comparator before the swap operation will change the sort order from increasing to decreasing. Again, as stated in Step 4 above, you can choose which sorting algorithm to use.
* Add some suitable comments in the source code.
* Please read the above implementation details carefully **BEFORE** start coding***.***
* After finishing coding, you need to debug your program and run it multiple times to verify that the random values in the array have been sorted.

This assignment has **20 points with grading components listed in the table below:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **generate random variables** | **Decreasing sort** | **Pass array to method** | **Code indent & align** | **header / code comment** | **Naming convention** | **Submit file format** | **Program logic** |
| **3** | **3** | **3** | **5** | **1** | **2** | **1** | **2** |

The grading components are, but not limited to: code alignment and indentation, variable/method/class naming convention, programmer header, suitable comments, submitted file format, overall program logic.

***REMIND:***

**If the submitted program does not compile or does not run, it will receive ZERO point.**

For Eclipse java file, if there is still any red circle check mark in java source code, you will receive zero point, because red circle check mark in Eclipse IDE means the java source code still has compilation error. You have to fix this compilation error first, before you proceed to the next phase of running java program. Even in some rare scenarios, you can run the program in Eclipse with compilation error, but a compilation error in java source code will result in zero point for the homework.

**What and where to submit:** you need to ***zip JohnDoeHw6.java only*** and submit the zip file ***JohnDoeHw6.zip*** to Moodle homework 6 drop box. After submission, be sure to follow the **VERIFICATION** process: download your .zip file to a local folder in your computer, unzip the zip file and compile then run the Java program in Eclipse. If it works correctly, then you have submitted the .zip file successfully in Moodle. If not, you need to fix the problem and re-submit. DO NOT zip any eclipse project folder into your submission zip file.

Please read “RulesForIndentAndAlignCode.docx” to know the 5 rules in the document. You are required to follow these 5 rules, and have these 5 rules built into your coding habit.