Assignmnet # 2, Data Strcutures

Overview

In this assignment you will be implementing and testing all three sort algorithms: Bubble Sort , Selection Sort , and Insertion Sort .

In additions, you will also be writing a driver to test the search algorithms and you will be measuring the run times of each search.

You will also be using the **RunTime** class that you created for Homework 1.

Finally, you will be analysing and comparing the performance of the three sort algorithms based on the type of array that was being sorted and the run times you computed.

----------------------------------- Details -----------------------------------

Details

1 **RunTime Class** You will copy the **RunTime** class that you created in Homework 1 to the project you are using for this assignment.

2 **BubbleSort Class** You will write the **BubbleSort.java** class which will **inherit** from **RunTime.java** and **implement** the Sort Interface using the Bubble Sort algorithm. The interface may be downloaded from SortInterface.java Please note that your **sort** method must measure the run time and add it to the **RunTime** class by using the **addRunTime()** method.

3 **SelectionSort Class** You will write the **SelectionSort.java** class which will **inherit** from **RunTime.java** and **implement** the Sort Interface using the Selection Sort algorithm. The interface may be downloaded from SortInterface.java Please note that your **sort** method must measure the run time and add it to the **RunTime** class by using the **addRunTime()** method.

4 **InsertionSort Class** You will write the **InsertionSort.java** class which will **inherit** from **RunTime.java** and **implement** the Sort Interface using the Insertion Sort algorithm. The interface may be downloaded from SortInterface.java Please note that your **sort** method must measure the run time and add it to the **RunTime** class by using the

addRunTime() method.

5 **Driver Class** You will write the **Driver.java** class which will **implement** the [Driver Interface](). The interface may be downloaded from [DriverInterface.java]()

6 **Output From Driver Main Method** Please note that, in addition to implementing the **DriverInterface**, you are also required to write your own **public static main(String[] args)** method in **Driver.java**. Your **main()** method will have to call the **runSort()** method to sort each of the following array types **ten** times for each sort algorithm:

1 1,000 equal **Integer**s.

2 1,000 random **Integer**s.

3 1,000 increasing **Integer**s.

4 1,000 decreasing **Integer**s.

5 1,000 increasing and random **Integer**s.

6 10,000 equal **Integer**s.

7 10,000 random **Integer**s.

8 10,000 increasing **Integer**s.

9 10,000 decreasing **Integer**s.

10 10,000 increasing and random **Integer**s.

7 For each call to the **runSort()** method to sort an **ArrayType** using a **SortType** **ten** times, your **main()** method will produce the following output: **SortType**, **ArrayType**, **Array Size**

8 ---

9 *runTime1 runTime2 runTime3 runTime4 runTime5 runTime6 runTime7 runTime8 runTime9 runTime10 --- Average runTime*

10

11 **Analysis:** Using the run time tables you created by running Driver.main(), copy your results into a Microsoft Word document and answer the following questions using 1-3 complete sentences for each question:

1 Which sort worked best on data in constant or increasing order (ie already sorted data)? Why do you think this sort worked best?

2 Did the same sort do well on the case of mostly sorted data? Why or why not?

3 In general, did the ordering of the incoming data affect the performance of the sorting algorithms? Please answer this question by referencing specific data from your table to support your answer.

4 Which sort did best on the shorter (ie n=1,000) data sets? Did the same one do better on the longer (ie n=10,000) data sets? Why or why not? Please use specific data from your table to support your answer.

5 In general, which sort did better? Give a hypothesis as to why the difference in performance exists.

Are there results in your table that seem to be inconsistent? (ex. If I get run times for a sort that look like this {1.3, 1.5, 1.6, 7.0, 1.2, 1.6, 1.4, 1.8, 2.0, 1.5] the 7.0 entry is not consistent with the rest). Why do you think this happened?

Link for the Assignment :
http://comet.lehman.cuny.edu/sfakhouri/teaching/cmp/cmp338/s17/hw/hw2.html