Wanner HernandezR

CMSC451 Project 1

Professor Didulo, Dennis

June 13, 2021

**Project 1  
CMSC 451 Project 1**

The first project involves benchmarking the behavior of Java implementations of one of the following sorting algorithms, bubble sort, selection sort, insertion sort, Shell sort, merge sort, quick sort or heap sort. You must post your selection in the "Ask the Professor" conference. No more than five students may select any one algorithm.

You must write the code to perform the benchmarking of the algorithm you selected. Your program must include both an iterative and recursive version of the algorithm. You do not have to write the sorting algorithms yourself, you may take them from some source, but you must reference your source.

You must identify some critical operation to count that reflects the overall performance and modify each version so that it counts that operation. In addition to counting critical operations you must measure the actual run time in nanoseconds.

In addition, you should examine the result of each call to verify that the data has been properly sorted to verify the correctness of the algorithm. If the array is not sorted, an exception should be thrown.

It should also randomly generate data to pass to the sorting methods. It should produce 50 data sets for each value of *n*, the size of the data set and average the result of those 50 runs. The exact same data must be used for the iterative and the recursive algorithms. It should also create 10 different sizes of data sets. Choose sizes that will clearly demonstrate the trend as *n* becomes large. Be sure that the data set sizes are evenly spaced so this data can be used to generate graphs in project 2

This project should consist of two separate programs. The first of those programs should perform the benchmarking described above and generate two data files, one containing the results from the iterative algorithm and the one containing the results of the recursive algorithm.

The benchmarking program must be written to conform to the following design:

page1image3246818880Diagram

Description automatically generated

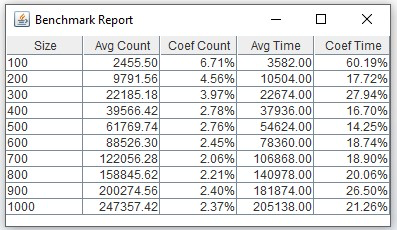
The output files should contain 10 lines that correspond to the 10 data set sizes. The first value on each line should be the data set size followed by 50 pairs of values. Each pair represents the critical element count and the time in nanoseconds for each of the 50 runs of that data set size.

The second program should produce the report. It should allow the user to select the input file using JFileChooser. The report should contain one line for each data set size and five columns

and should be displayed using a JTable. The first column should contain the data set size the

second the average of the critical counts for the 50 runs and the third the coefficient of variance of those 50 values expressed as a percentage. The fourth and fifth column should contain similar data for the times. The coefficient of variance of the critical operation counts and time measurement for the 50 runs of each data set size provide a way to gauge the data sensitivity of the algorithm.

Shown below is an example of how the report should look:



You must research the issue of JVM warm-up necessary for properly benchmarking Java programs and ensure that your code performs the necessary warm-up so the time measurements are accurate.

Project 1 Part 1

Files name:

* BenchmarkSorts.java
* Main.java
* SortInterface.java
* UnsortedException.java
* YourSort.java

Project 1 Part 2

File name:

* Main.java

**Test cases:**

Part 1

Test case 1: This test case will show that the program is perfectly working and it will save the file in the CMSC451Project1Part1folder.

Graphical user interface, text, application, email

Description automatically generated

Test case 2: This Test case show where the files are been store.

Graphical user interface, application

Description automatically generated

Test case 3: This Test case will show the file information and what is been inputted. As well what is the information that is been display.

Graphical user interface, text, application

Description automatically generated

Part 2

Test case: This will display the second part of the project information by selecting a file form the folders file recursive.txt and iterative.txt and it will display the information on the file.

Graphical user interface, text, application

Description automatically generated