Sorting

Dr. Andrew Rosen

Abstract

As we have learned, sorting algorithms perform two operations: comparisons, to check to see if two pieces of data are in so kind of order, and exchanges, which swap to pieces of data. In this lab you will implement various sorting algorithms and compare the amount of time it takes for them to run.

1 Premise

Implement Insertion Sort, Quicksort, and one other sorting algorithm from the list below.[1]

- Shellsort
- Mergesort
- Heapsort
- Timsort (+7 points EC)

Modify the sorting algorithms to keep track of

- The runtime of the algorithm
- The number of comparisons.
  - Here, a comparison is whenever we are checking to see if two items are out of order. It does *not* include checking to see if we're out of bounds or anything index related.
- The number of exchanges.
  - Whenever two items get swapped or an item gets copied or moved, that's an exchange.

[1] Remember to cite if you use code from the book or class.

2 Testing and Collecting Data

Test your algorithms on differently sized lists or arrays of integers. A good test will have many data points of varying magnitudes and should test each algorithm at each size multiple times. For example, a decent test might test lists that have sizes that are powers of 10s (10, 100, 1000, 10000 ...), and each data point/line in your output would be the average of the results of 100 runs on an array of a specific size.

Output your data to a file. It may be useful for you to output your data into a `.csv` file if you plan on using Excel in the next portion.

2.1 Implementation Hint

SIncve you can only ever return one value in Java, you will need to work around that to gather the number of comparisons and exchanges performed. I reccomend one (only one) of the following:

- Creating a Data object, which represents the data collected for a single sample (a single run on sorting algorithm) and thus contains fields for exchanges and comparisons. Modify the sorting algorithms to hold these.
- Using static variables to keep track of exchanges and comparisons and reset them for every sample.

Also, this is one assigment you are actually free to lookup the algorithm from the book or online, then modify it, **but you must cite your source.**

3 Presenting Your Data

Your objective is to present your data in such a way that a non-programmer can tell which algorithms perform better. We will give you a fair amount of freedom here. Some options include:

- A well formatted table.
- Graphing your data (using a program such as Excel).

4 Grading

10 points Insertion Sort

30 points Quicksort

20 points Third sorting algorithm of student's choice.

20 points Data is collected.

15 points Data is presented in a meaningful way.

5 points Data presentation is aesthetically pleasing.

7 points Extra Implement Timsort as the third algorithm.