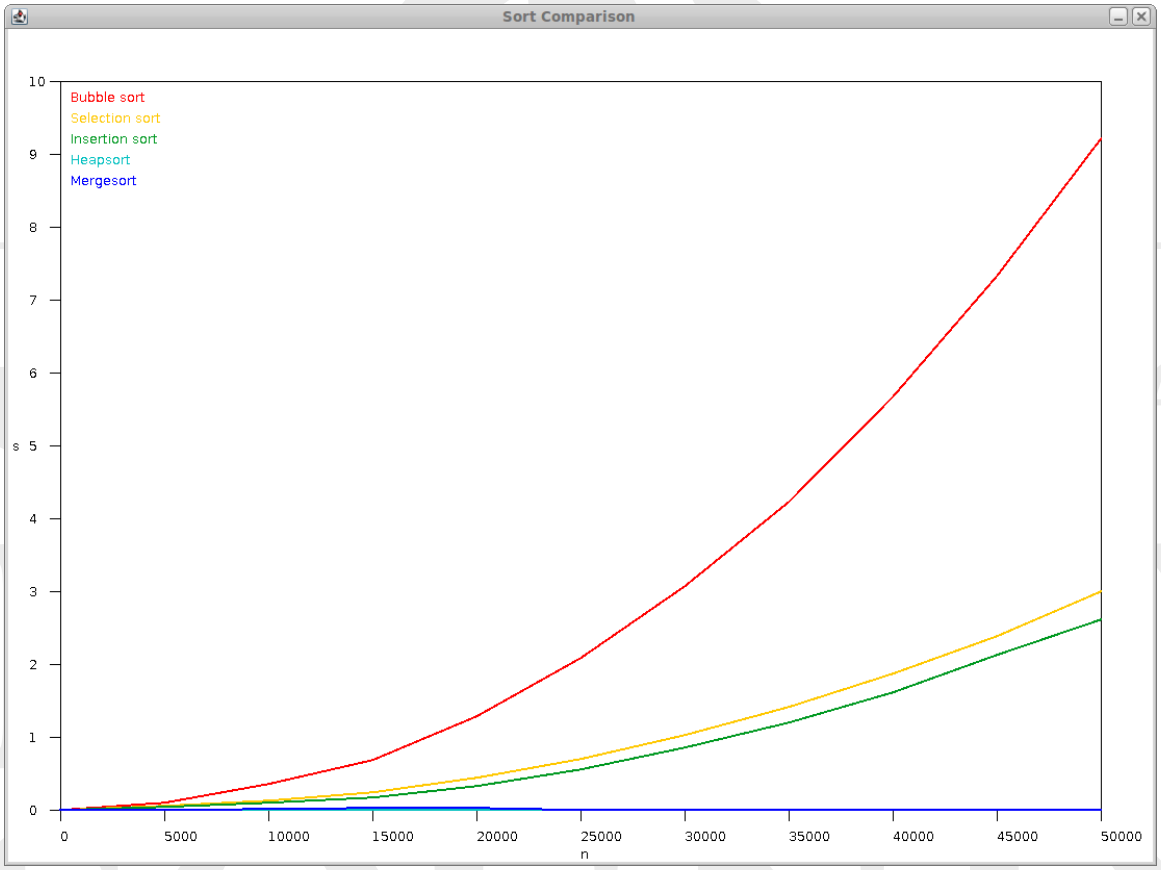
Lab 7: Comparing & Graphing Sorts

In this lab we will be comparing the `Sorter`s you have already created. You should have a working `BubbleSorter`, `InsertionSorter`, `SelectionSorter`, `HeapSorter`, and `MergeSorter`. If you are still missing one, we will still accept this lab. If you're missing more than one, you should spend your time working on that instead. (Remember that you can always download your already-submitted code from Moodle.)

Your final program should be called `GraphSorts`. Its output should be a graph of array size vs. the length of time taken, such as shown here:



The graph you print out should be a line graph, containing curves for each sort. Size will be on the x-axis, and time (in seconds) on the y-axis. Each sort should be graphed in a different color, and it should be clear which one is which (such as with a key like that shown above). The point (0,0) should be in the lower-left corner. The maximum time should be 10 seconds, while the maximum size should be 50000. You will need to plot a point for every size increase of 5000. You should use the `timeSort()` method contained within the `Sorter` class to obtain the proper numbers.

The exact size of the output is up to you. However, be sure to use constants in your code, to avoid any hard-coding. One good technique is to have a constants called `GRAPH_WIDTH` and `GRAPH_HEIGHT`, and `MARGIN`. Then, you can calculate `WINDOW_WIDTH` and `WINDOW_HEIGHT` from these.

You will have to spend some time translating window coordinates to graph coordinates. Remember that window coordinates have (0,0) in the upper-left corner. It will probably pay off to make a private function called `drawCurve()`, that takes a `Color`, a `Graphics2D` ("pen"), and 2 arrays of doubles–one for the x coordinates, and one for the y coordinates–and then plots a nice curve using the `Graphics2D`'s `drawLine()` method.

If you wish to draw thicker lines, you will have to use the `Graphics2D`'s `setStroke()` method. It takes one argument, of a `Stroke` class. It is easiest to make a constant stroke, that is equal to `new BasicStroke(2.0f)`. (The `f` indicates that this number is a `float`, rather than a `double`.)

Decide if you want to work on the graphics or the sorting first. If you want to work on the graphics first, you should just make some arrays of fake values to plot. If you want to work on the sorting first, you can make a program that compares the `Sorter`s via a text output. You should only move on to the second half when you're *sure* that the first half is working properly.

When you turn in your files, be sure to turn in every `Sorter` that you made with it. (You don't need to turn in the abstract `Sorter` class that we provided, though.)