Part I: Identifying Sort Algorithms

https://people.eecs.berkeley.edu/~jrs/61b/lab/MysterySort/

(1) The words in parentheses are the explanations why I think the sort is

#1 Quick sort (presorted array is slow but the random one is fast)

#2 Selection sort , Heap sort (find the largest item from unsorted list)

#3 Merge sort (divide and conquer)

#4 Insertion sort (presorted array is fast)

(2) One of the algorithms is quicksort. Which element does it choose for the pivot in a partition? (The pivot is not randomly chosen.) How can you tell?

Select the pivot from the largest/smallest index of the value in the unsorted part. (i.e. the parameter q in quickSort(q))

And the data marked in the green line is what pivot is.

Part II: Timing Sort Algorithms

$ javac -g SortPerf.java

$ java SortPerf select 1 50 1000 select.dat

$ java SortPerf insert 1 50 1000 insert.dat

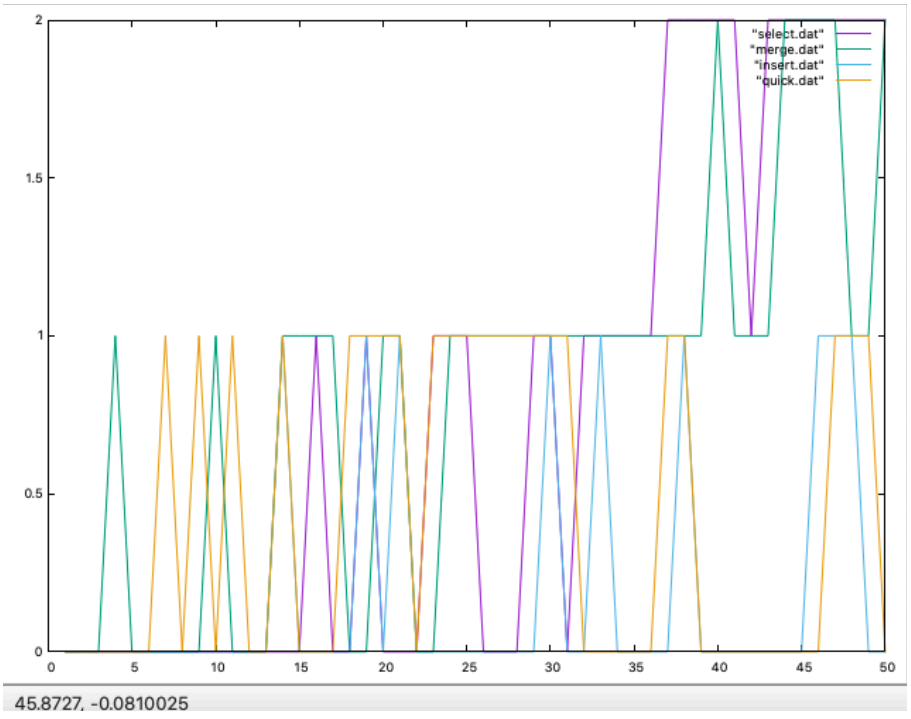
$ java SortPerf merge 1 50 1000 merge.dat

$ java SortPerf quick 1 50 1000 quick.dat

$ brew install gnuplot

$ gnuplot

$ plot "select.dat" with lines, "merge.dat" with lines, "insert.dat" with lines, "quick.dat" with lines



Part III: Building a Better Sort

$ gnuplot

$ plot "select.dat" with lines, "merge.dat" with lines, "insert.dat" with lines, "quick.dat" with lines, "best.dat" with lines

From the diagram in Part II, when length < 50 insertionSort is better than quickSort. However, in the following diagram, when length > 50, insertionSort has terrible results between 80 and 100.

Hence, my solution is

```
if(A.length < 50) {
Sort.insertionSort(A);
} else {
Sort.quicksort(A);
}
```