Group 9-Sorting Competition 2015

Tyler Lemke
Ryan McArthur

Results

- Overall 16th place.
- Among the class 9th place.
- Small data results -

  634, 645, 633 with median = 634
- Big data results -

  2160, 2180, 2190 with median = 2180
- Sum of medians = 2814

Our Methodology Through the Competition

- Began by implementing counting sort and arrays.sort on hashmaps of longs.
- using hashmaps came with a couple of issues.
- Our implementation took too long to process the data before we even started
- Converting all the strings to longs took a long time especially with Long.parselong().
- We decided to scrap it and start anew.

Our Final Implementation

- We used the built in Arrays.sort to sort the initial 9 digits as strings. We then used an implementation of counting sort on ints to sort upon the sum of the first 4 digits, in reverse order.
- We decided to use this because it took a lot less time to process the data in a way that followed the guidelines than it did using the hashmap method.
- We were working on getting rid of Arrays.sort and trying to sort on something other than strings, but with the time wasted on trying out the hashmap approach we never got to it.

Running Time of Our Sort

- Since our method implements a form of counting sort and TimSort, at the worst case we get **Θ**(nlog(n)).
- Our expected case is also **Θ**(nlog(n)).
- With our counting sort we traverse the array 3 times.

Data Storage

- Original Array of strings.
- Arrays.sort dealt with that array.
- Our counting sort actually created a separate array of ints that we sorted and just made index changes on the original array of strings so no comparisons of strings had to happen. Other than the changes happening to both arrays the implementation of counting sort was pretty much by the book.

- We planned to do this with the first sort too but ran out of time due to the failed first attempt.