CSSE230: Sorting Races

# Name(s): Austin Swatek

# Part 1: Data

Table 1 shows the runtimes of 6 sorts for at least 4 different types of arrays:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Name | SelectionSort | InsertionSort | TreeSort | HeapSort | QuickSort |
| Runtime | 162ms | 359ms | 258ms | 438ms |  |
| Size | 1,000,000 | 1,000,000 | 1,000,000 | 1,000,000 | 1,000,000 |

# Part 2: Discussion

Include your discussion of the runtimes in Table 1, as described in the specification.

SelectionSort – N^2

InsertionSort – N^2

TreeSort – N\*log(N)

HeapSort – N\*log(N)

I thought that HeapSort was supposed to be one of the better sorts, but when compared to Java’s built-in sort for arrays, it is much consistently slower. I am not sure if that is because of a bad way I implemented HeapSort, or if the regular sorts are getting lucky with the data sets. With one million items, there were no memory issues, but this was all with integers. With other types of data, there could be bigger issues with spacing I guess.