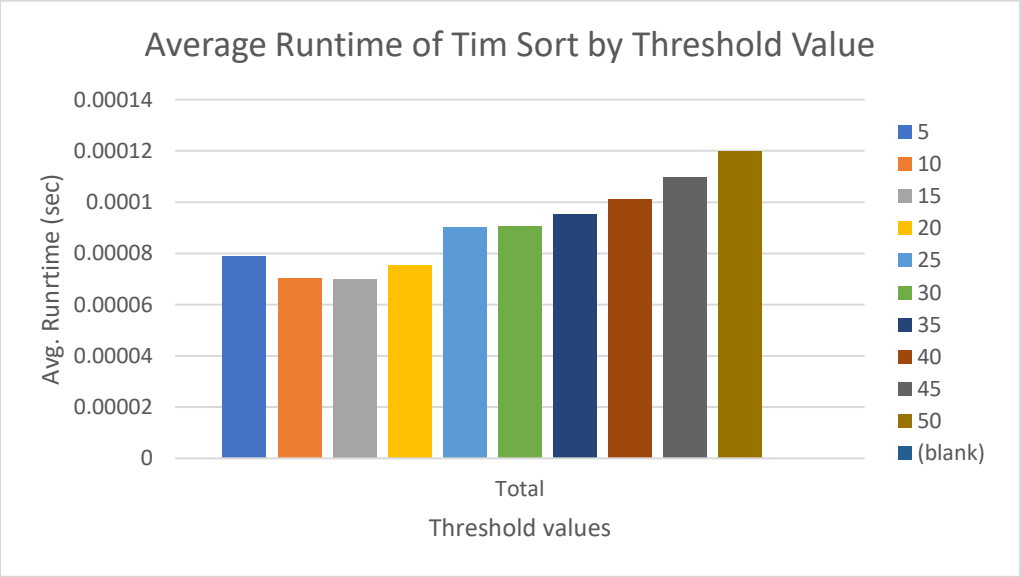
Alex Holt – A57959647

Hypothesis: For my hybrid sort, I predict that the optimal k value will be around 30 due to my findings in part 1 of this assignment.

Methods: To solve this problem, I wrote the code at the following link: https://github.com/Holtster2000/cse431_hw4/blob/master/q2.py

I used my implementation of merge and insertion sort from question 1 to implement tim_sort. I then used a random generated list of integers for multiple values of N and k. I used thresholds from 5 to 50 in increments of 5 and for each k, used values of N from 10 to 100 in increments of 10. Each test was ran 1000 times to account for random variations.

Results: From the output of the program, I created a graph showing the average runtime for each threshold value k.



As shown, the best runtimes are at a threshold value of 15 for tim_sort.

Discussion: I was surprised to see the optimal threshold to be lower than the result I previously found in question 1. However, since we are implementing merge sort on top of the insertion sort, I wonder if the expected value of 30 was cut in half to 15. Since merge sort performs the same as insertion sort at 30 elements, we only need to switch to insertion sort when below 30 elements and merge sort splits the list in half each recursion. This could explain why see 15 being the optimal threshold.

Conclusions: Under the conditions tested, the best threshold value k for tim_sort, a hybrid merge sort and insertion sort, is 15.