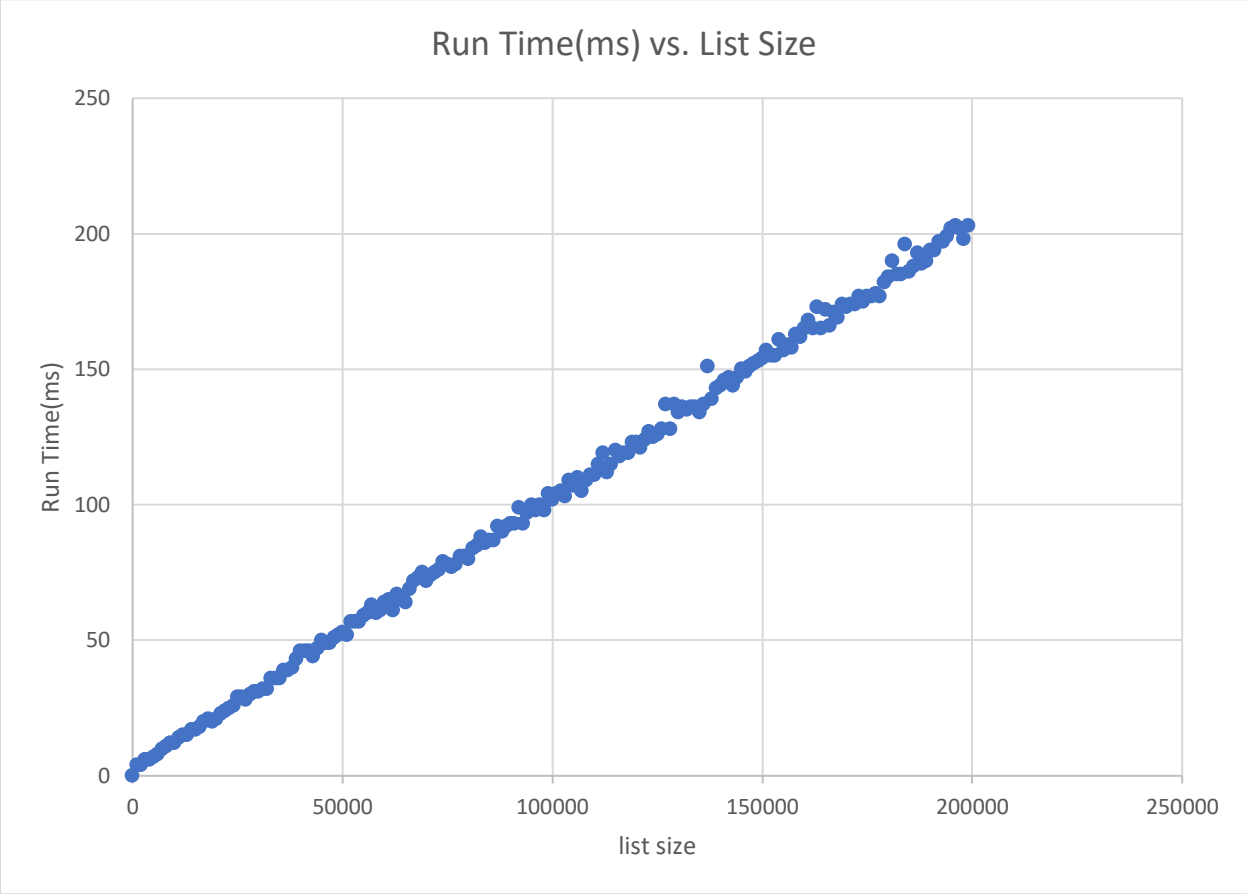
Experiment 1

1. This experiment is testing the time needed for the topKSort to run 10 times with K = 500 and input list size varying from 0 to 200000 with step of 1000.

2.

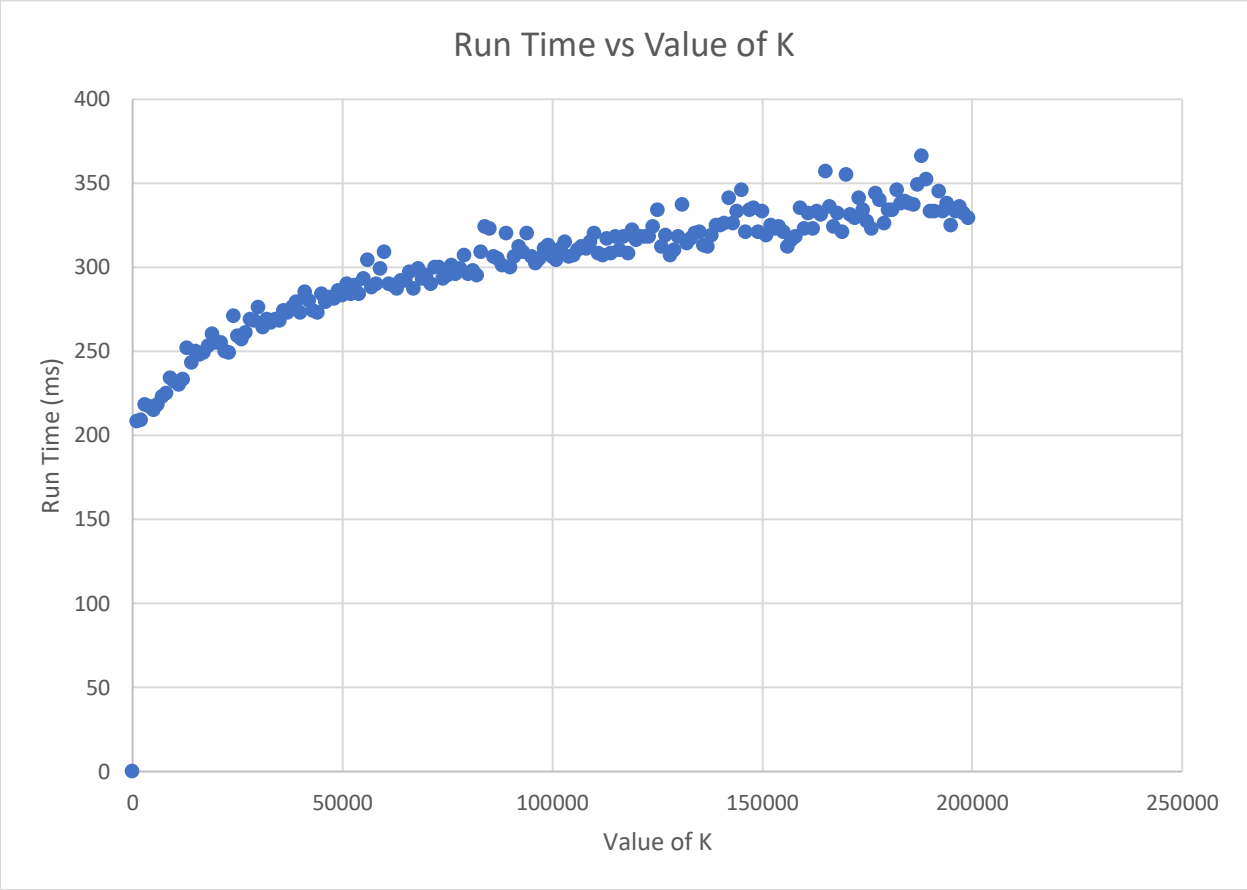


3. The result matches with my prediction of a big-O of n runtime. Since our topKSort is designed to be big-O of nlog(K), and since K is a constant value, the runtime would have a linear relationship with the size of the list which is n.

Experiment 2

1. This experiment is testing the time needed for the topKSort to run 10 times with fixed input list size of 200000 and K varying from 0 to 200000 with step of 1000.

2.

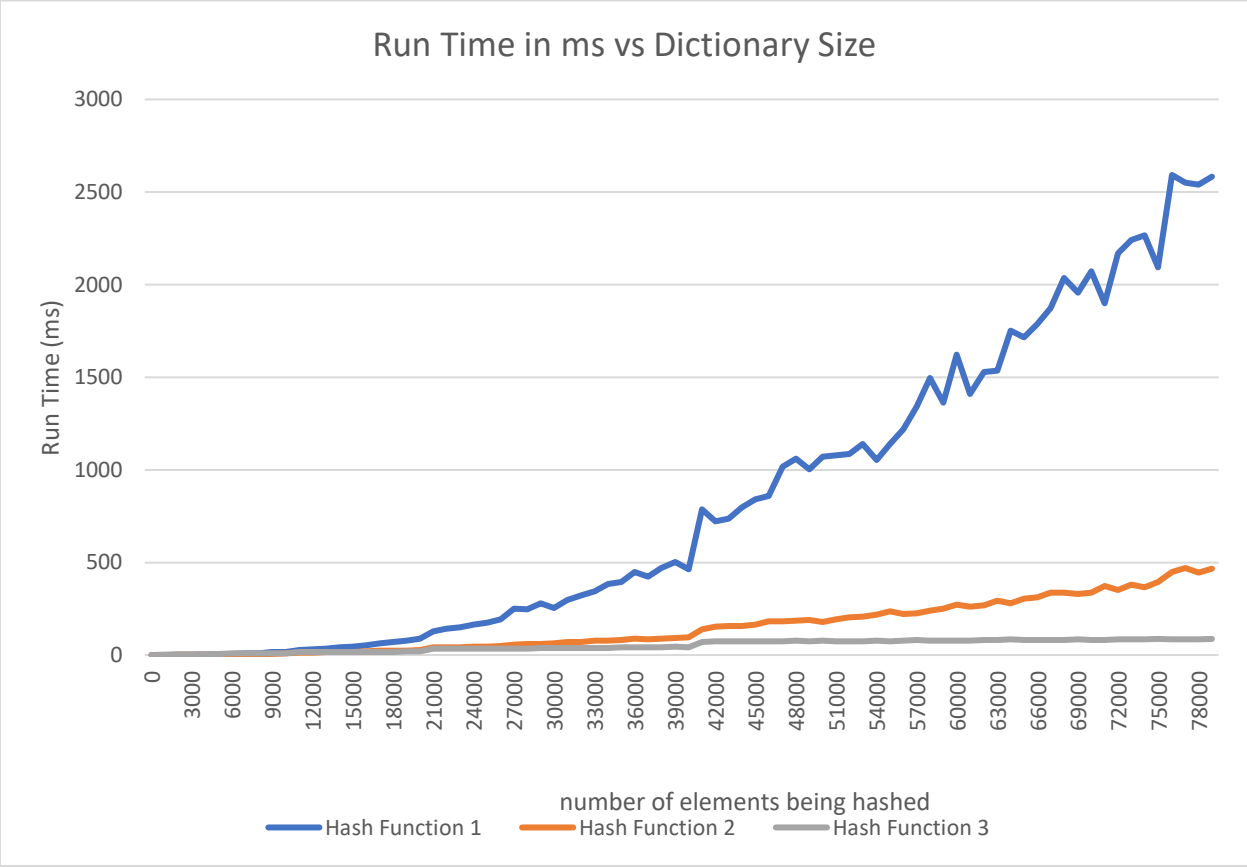


3. The result matches with my prediction of a big-O of log(k) runtime. Since the topKSort has runtime of big-O of nlog(k), and the list size n is constant in this case, so the relationship for run time vs. the value of k would be in log(k).

Experiment 3

1. It's calculating the time needed for putting in a list of random characters into a Chained Hash Dictionary which size varies from 0 to 8000 with step of 1000. Each test cases in the experiment uses a different method to calculate the hash code.

2.



3. The result matches with my prediction. The first test case is using hash code that "makes the chainedhashmap take a hideously long amount of time", thus having the greatest runtime. The second test case is using a normal method to generate the hash code. And the third test case is including prime number 31 in generating the hash code which reduces the chance for each char to be stored in same bucket, and therefore time needed to traverse inside each bucket would be less than the other two cases. Hence, the third test case is the fastest.