D1 – Binary Search Experiments

Output:

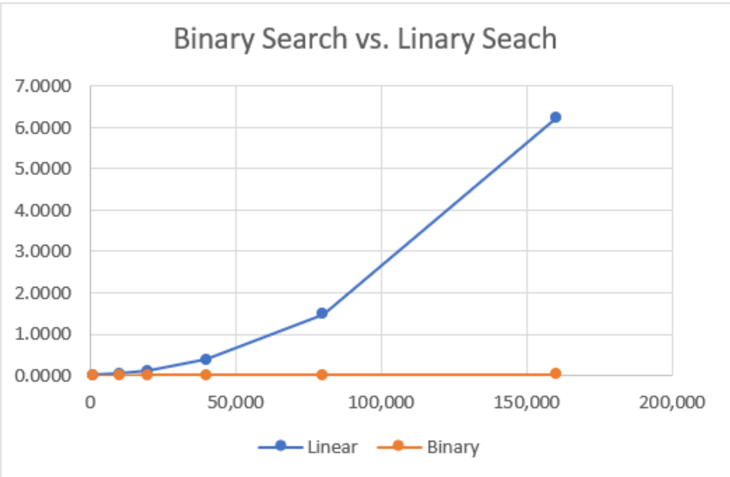
	N = 1000			N = 10000			N = 20000		
	Test 1	Test 2	Test 3	Test 1	Test 2	Test 3	Test 1	Test 2	Test 3
Binary Search	0.005	0.001	0.002	0.003	0.004	0.003	0.005	0.004	0.004
Linear Search	0.009	0.008	0.007	0.05	0.042	0.049	0.116	0.109	0.117
		Binary total:	0.008		*Binary total:*	0.010		*Binary total:*	0.013
		Linear total:	0.024		*Linear total:*	0.141		*Linear total:*	0.342

	N = 40000			N = 80000			N = 160000		
	Test 1	Test 2	Test 3	Test 1	Test 2	Test 3	Test 1	Test 2	Test 3
Binary Search	0.007	0.007	0.007	0.012	0.013	0.012	0.024	0.024	0.024
Linear Search	0.382	0.369	0.414	1.467	1.48	1.497	6.212	6.205	6.249
		Binary total:	0.021		*Binary total:*	0.037		*Binary total:*	0.072
		Linear total:	1.165		*Linear total:*	4.444		*Linear total:*	18.666

Binary search total (all tests) :	**0.161**
Linear search total (all tests) :	**24.782**

time (seconds)

n	Linear	Binary
1,000	0.0080	0.0027
10,000	0.0470	0.0033
20,000	0.1140	0.0043
40,000	0.3883	0.0070
80,000	1.4813	0.0123
160,000	6.2220	0.0240



Conclusion:

i) How long would you estimate that both methods use when N=320000? Validate your estimate.

- Linear search:

  It seems that the time it takes for every larger sample size, the time increases dramatically. My estimation would be that for a sample size of n=320.000 that the search time would be around 24 - 28 seconds.

  The reason for this being that from a sample size of n=20.000 increased to n=40.000 the time multiplier was 3,4. And from n=40.000 to n=80.000 the time multiplier was 3,8 (increase of 0,4). And for the final test, when the sample size of n=80.000 was changed to n=160.000 the time multiplier between sample sizes was 4,2 (increase of 0,4 again). Therefore, my estimation is that the increase from n=160.000 to a sample size of n=320.000 would be:

  Low) 6,2 seconds * 4 = 24,8 seconds

  High) 6,2 seconds * (4,2+0,4) = 28,5 seconds

- Binary search:

  My estimation would be that for a sample size of n=320.000 that the search time would be around 0,04 - 0,05 seconds.

  The reason being that the increase from a sample size of n=80.000 to a sample size of n=160.000 for a binary search, the time multiplier was approximately twice as much as the previous example (and around 1,8 when sample size increased from n=40.000 to a sample size of n=80.000).

  Therefore, my conclusion would be that the time it would take for a binary search with a sample size of n=320.000 would be:

  0,024 seconds * 2 = 0,048 seconds

ii) ii) What does this say about the complexity of the two methods?

- Linear search:

  For linear search, where one item is search at a time, the time complexity is Big-Oh of n^2, or $O(n^2)$ as we have two for-loops in the function. Therefore, the time factor increases very fast for larger sample sizes.

- Binary search:

  For binary search, the time does not increase so dramatically with larger sample sizes because in binary search, the sample size is cut in half for each step of the search (when list being searched is sorted). Therefore, the time complexity increases with larger sample sizes, as there is an increase in searches being made, but not in such drastically way as in linear search however. For binary search, the time complexity is Big-Oh of log n, or $O(\log n)$.