Result table for linear search algorithm

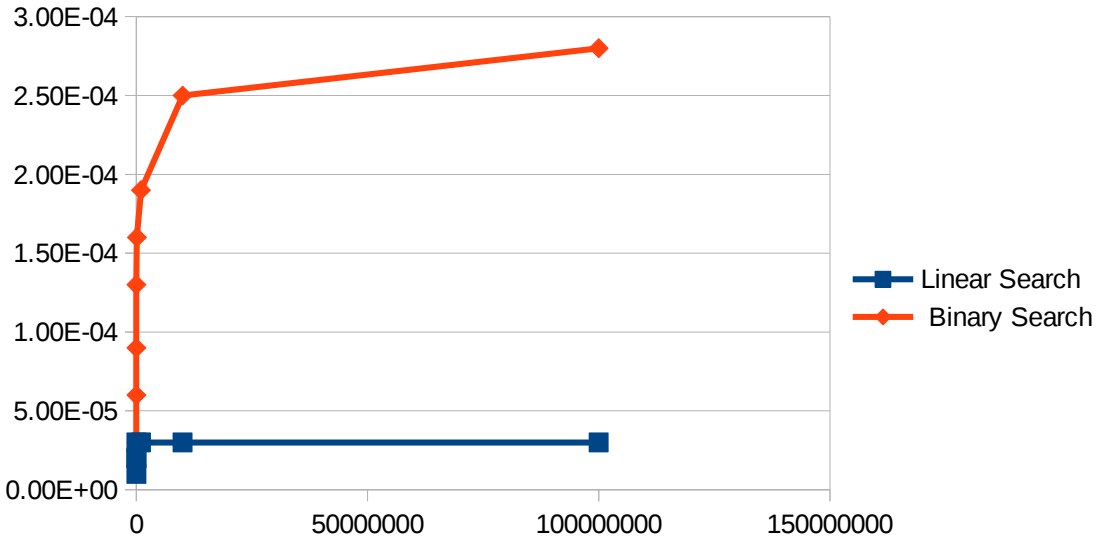
Size / Case & Complexity	(i) O(1)	(ii) O(n)	(iii) O(n)	(iv) O(n)
N = 1	3e-05 milliseconds.	1e-05 milliseconds.	1e-05 milliseconds.	2e-05 milliseconds.
N = 10	2e-05 milliseconds.	3e-05 milliseconds.	5e-05 milliseconds.	5e-05 milliseconds.
N = 10^2	2e-05 milliseconds.	0.00027 milliseconds.	0.0005 milliseconds.	0.00051 milliseconds.
N = 10^3	1e-05 milliseconds.	0.00257 milliseconds.	0.00471 milliseconds.	0.00468 milliseconds.
N = 10^4	2e-05 milliseconds.	0.02361 milliseconds	0.04716 milliseconds.	0.04729 milliseconds.
N = 10^5	2e-05 milliseconds.	0.23594 milliseconds.	0.47163 milliseconds.	0.47106 milliseconds.
N = 10^6	3e-05 milliseconds.	2.37766 milliseconds.	4.76218 milliseconds.	4.76097 milliseconds.
N = 10^7	3e-05 milliseconds.	23.899 milliseconds.	47.9624 milliseconds.	48.8717 milliseconds.
N = 10^8	3e-05 milliseconds.	258.312 milliseconds.	485.355 milliseconds.	497.9 milliseconds.

Result table for binary search algorithm

Size / Case & Complexity	(i)	(ii)	(iii)	(iv)
N = 1	2e-05 milliseconds.	1e-05 milliseconds.	1e-05 milliseconds.	2e-05 milliseconds.
N = 10	3e-05 milliseconds.	3e-05 milliseconds.	3e-05 milliseconds.	3e-05 milliseconds.
N = 10^2	6e-05 milliseconds.	7e-05 milliseconds.	7e-05 milliseconds.	6e-05 milliseconds.
N = 10^3	9e-05 milliseconds.	0.0001 milliseconds.	0.00471 milliseconds.	9e-05 milliseconds.
N = 10^4	0.00013 milliseconds.	0.00018 milliseconds.	0.00017 milliseconds.	0.00013 milliseconds.
N = 10^5	0.00016 milliseconds.	0.00019 milliseconds.	0.00018 milliseconds.	0.00016 milliseconds.
N = 10^6	0.00019 milliseconds	0.00024 milliseconds	0.00024 milliseconds	0.0002 milliseconds
N = 10^7	0.00025	0.00027	0.00028	0.00024

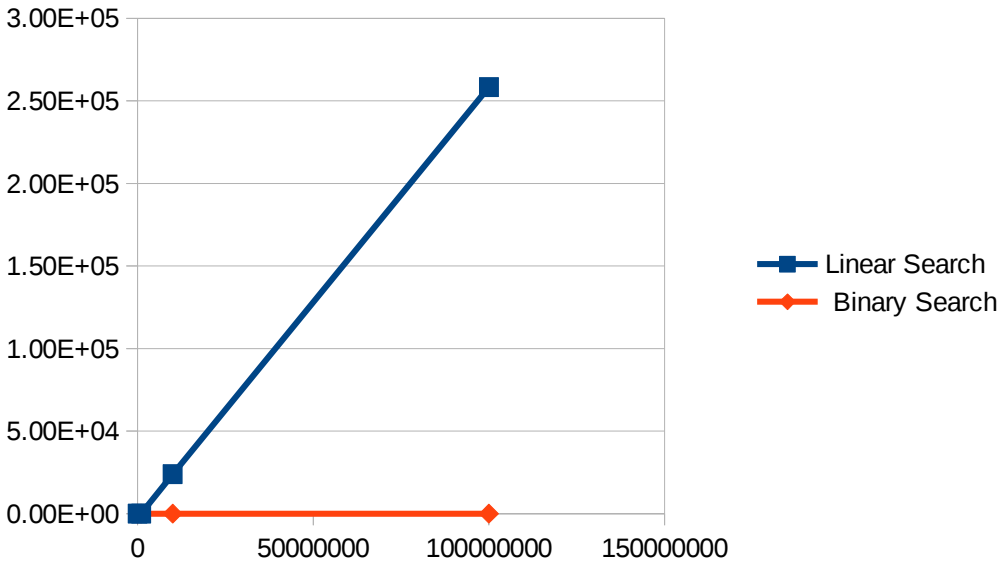
	milliseconds	milliseconds.	milliseconds.	milliseconds.
N = 10^8	0.00028 milliseconds.	0.00032 milliseconds.	0.00032 milliseconds.	0.00028 milliseconds.

Plot for the case (i)



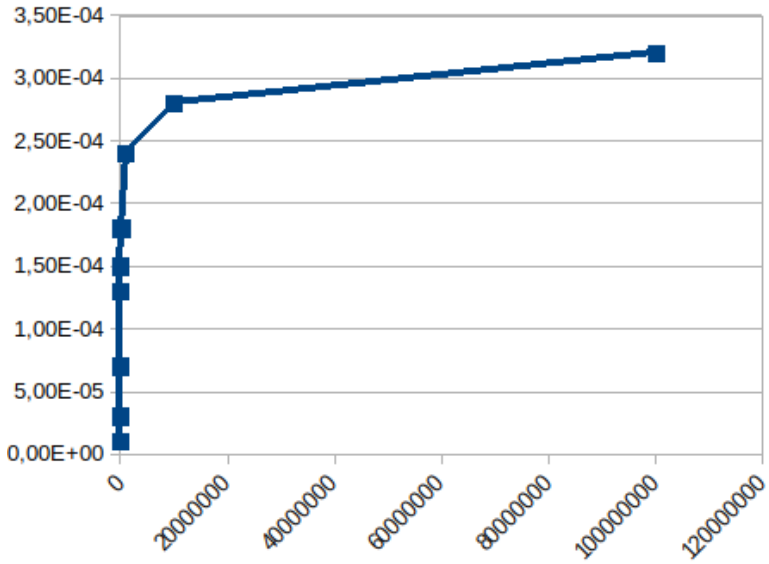
Here, we see the growth rate of time is logarithmic for binary search. Also, growth rate for binary search is nearly 0. Thus complexity of linearSearch is O(1) and complexity of binarySearch is O(logN) in this case.

Plot for the case (ii)

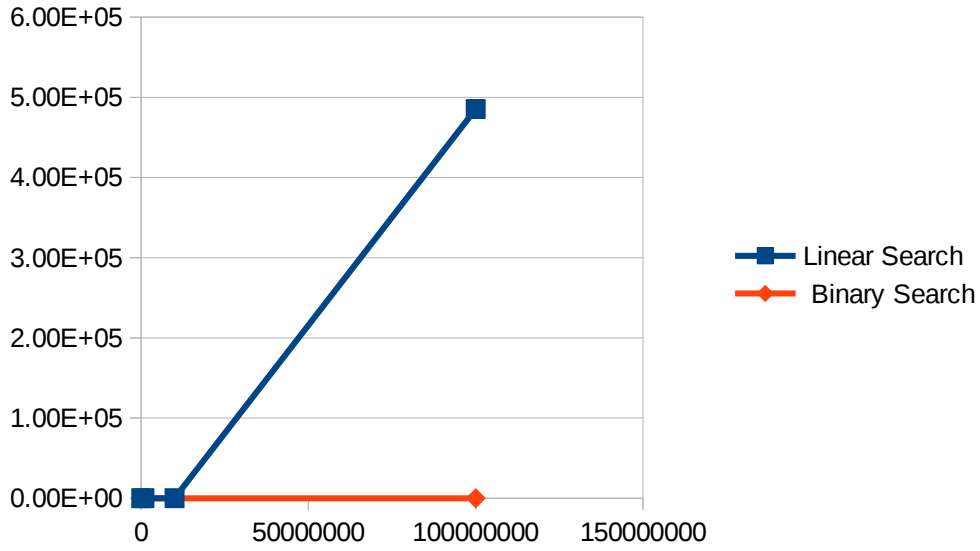


The growth rate of time for linear search is O(n). Here growth rate of time for binary search is again logarithmic so its O(logN), but since the plot is zoomed out it seems like its linear (although it is not).

To see this more clearly lets look at only the graph of binarySearch:



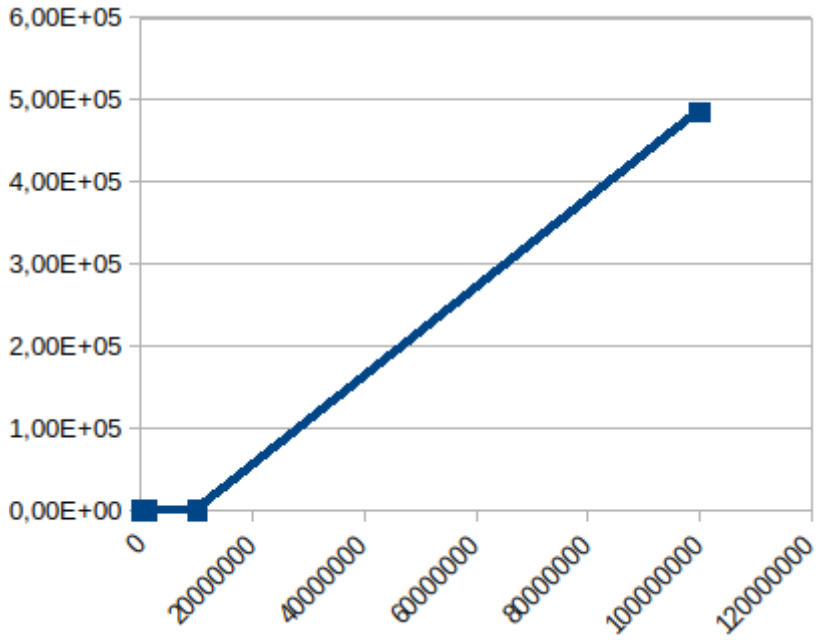
Plot for the case (iii)



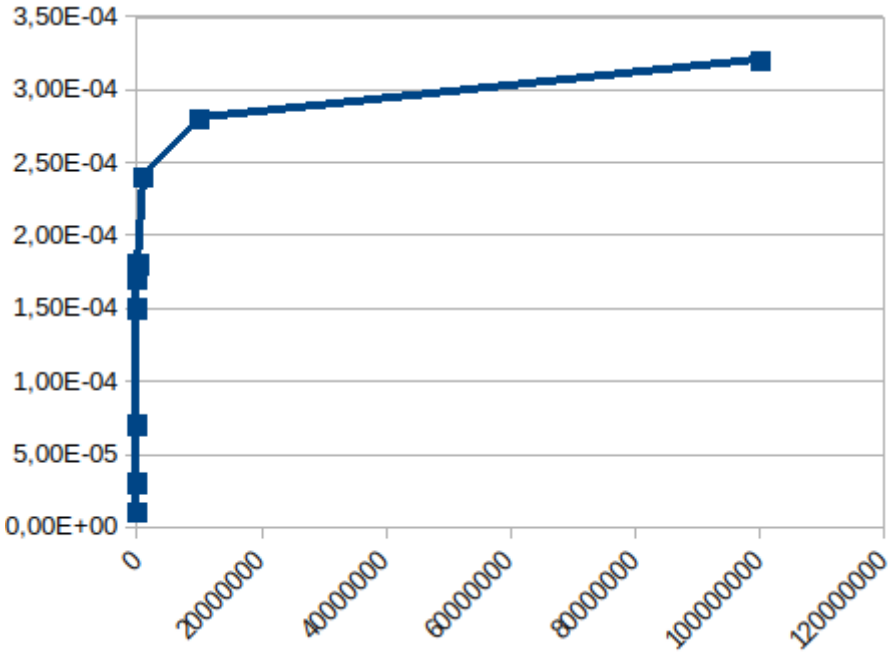
Situation is similar to (ii). Complexity of LinearSearch is O(n), complexity of binarySearch is O(logN).

Lets look at linearSearch and binarySearch one by one.

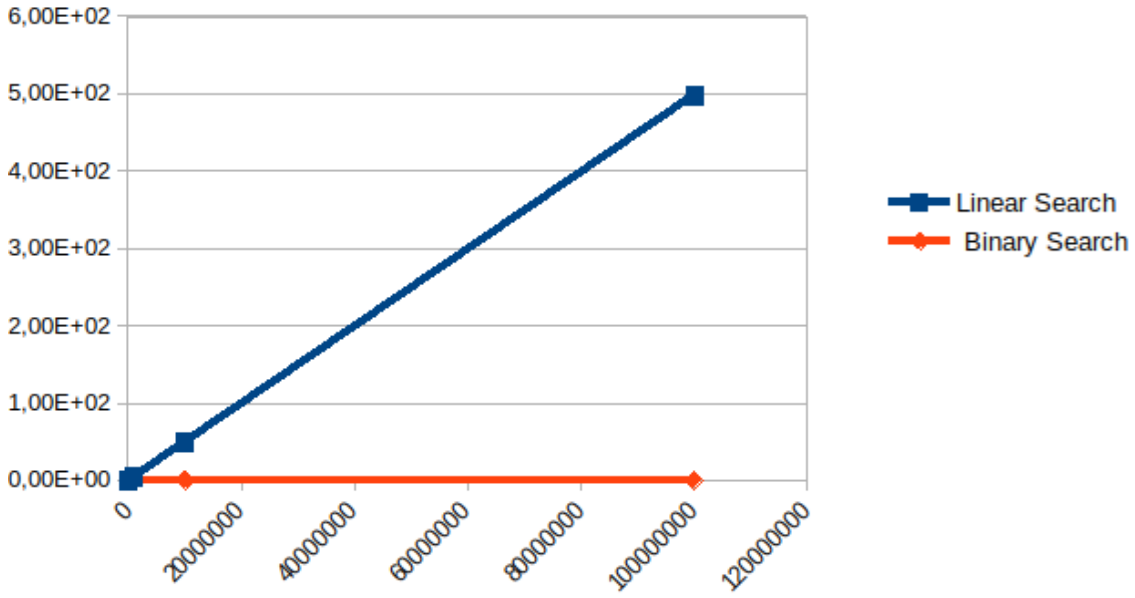
LinearSearch:



binarySearch



Plot for the case (iv): same as (iii), except the needed time is a little bit higher generally.



To sum it up (discussion)

From these observations we see that

- The best case scenario for linearSearch is case (i) and in best case it has O(1) complexity.
- The worse case scenario for linearSearch is case iii and iv and it hase O(n) comlexity.
- Binary search is in all cases has the O(logN) complexity.

Computer Specifications

Processor: Intel(R) Celeron(R) CPU 1000M @ 1.80GHz
Ram: 4Gb
OS: Linux Mint