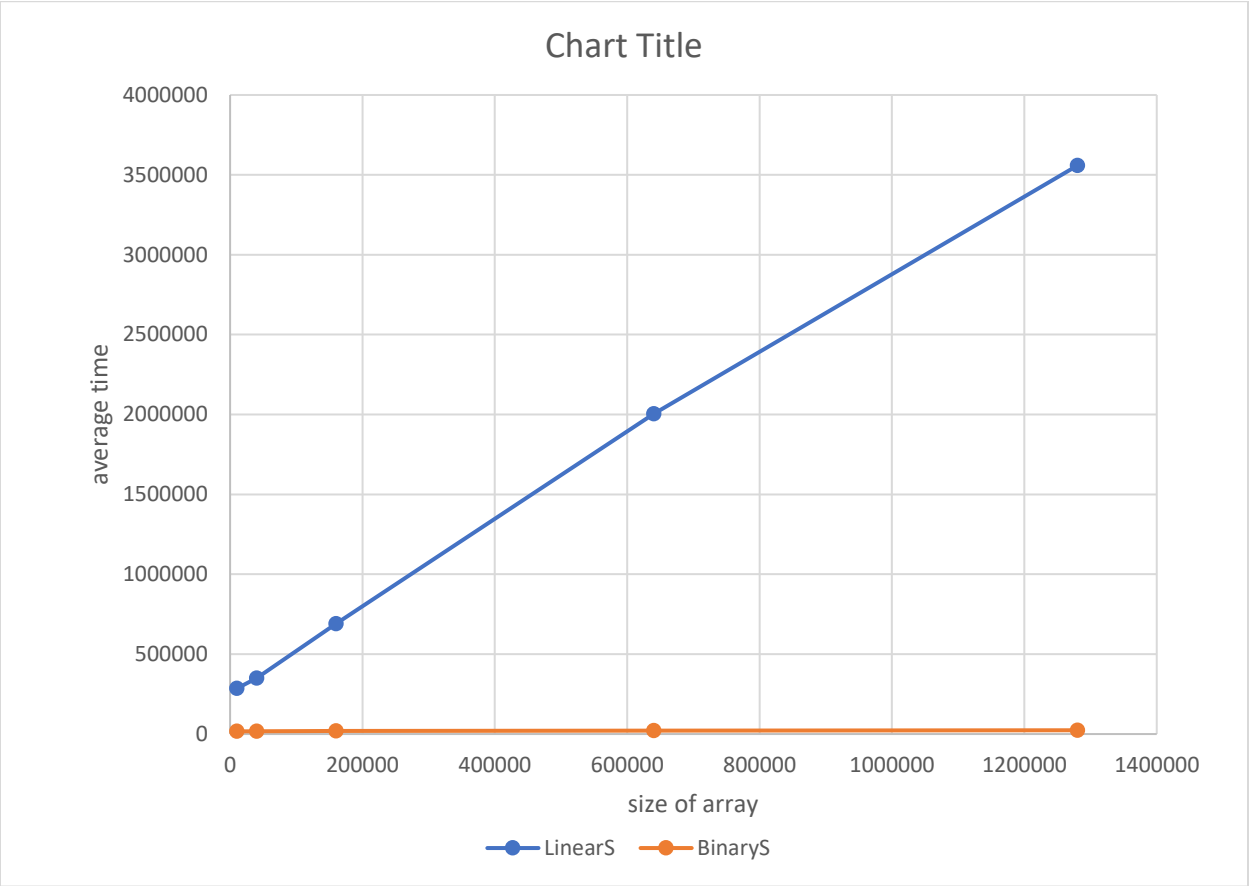
Ken Amamori
11/11/2018
Lab7
MW 3:00-4:20pm

It was the first time to consider and experiment the time elapsing of coding since most of the codes are very quick to compile and run. With this lab, I realized the need of compute the time of code because there is always more than one way to get the solution and it requires to know which is more efficient to pick the right one. This computer is Sony VAIO model-SVF14N13CXB x64 based pc, Intel Core i5-4200U CPU @ 1.60GHz 2.29, Windows 10 Home, Memory available 4,951MB/9,368MB, and java version 9.0.4. In order to gather the data for this experiment, I made two different methods (solutions) to find the index of a random value inside of an array of certain size that I generated in the program. Calculate the time for two methods for 50 times to be accurate and find the average to compare the two methods with 5 different size of arrays, 10000, 40000, 160000, 640000, and 1280000. The two methods were Binary Search and Linear Search which give the same solution in a different way. Linear Search looks the value from 0 to the last element int the array, which makes it O(n). In addition, the graph of linear search is a straight line, therefore, we know that O(n) is the time complexity. Binary Search separate the array into two and compare if the value is greater, smaller or same. If the value is not the same, it will look only for the other half and continue until the value is found. Every time the range of the array that will be searched is halved, which makes binary search an O(log_2(n)). From the graph, we also can say that binary search is following the graph of log_2, therefore, O(log_2(n)).

Ken Amamori
11/11/2018
Lab7
MW 3:00-4:20pm



	LinearS	BinaryS
10000	284839	16680
40000	350188	17849
160000	690684	20179
640000	2004348	21768
1280000	3557834	23588