Baykam Say, 21802030
CS 201 – 1

CS 201 HW 2

PART 1:

The sequential search algorithm starts from the first item of the array and looks through the items one by one. Take the input size as N and the time to look a single item as 1t:

If the key is at the beginning it takes 1t to find the key.

If the key is in second place it takes 2t to find the key.

.

.

.

If the key is in the middle it takes (N/2)t to find the key.

.

.

.

If the key is in the end it takes Nt to find the key.

Therefore, the time complexity for the upper bound for the sequential search is O(N).

The binary search algorithm looks at the middle item in the array. If the item is not equal to the key it divides the array into two, if the key is smaller it searches the first part and if the key is larger it searches the second part. In each case it again looks to the middle and repeats until it finds the key. Take the input size as N and the time to look a single item as 1t:

If the key is in the middle it takes 1t to find the key.

If the key is at the end of the first quadrant or the third quadrant it takes 2t to find the key.

.

.

.

If the key is one of the last items looked, it takes log(N)t to find the key.

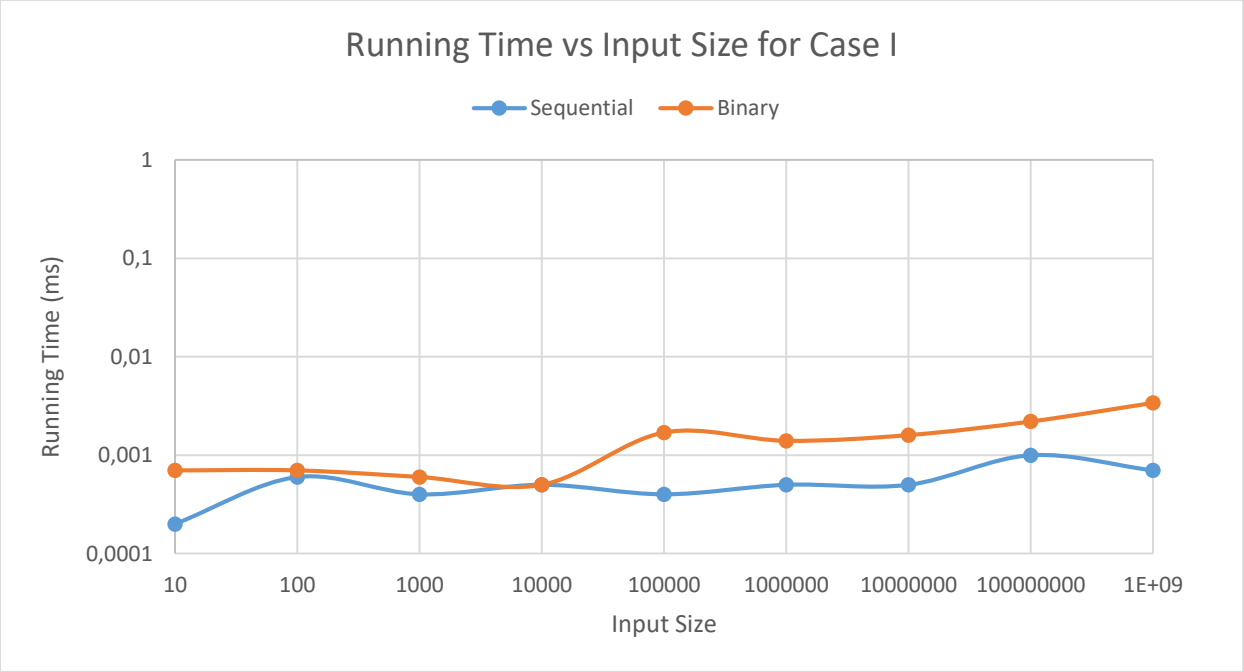
Therefore, the time complexity for the upper bound for the binary search is O(log(N)).

Baykam Say, 21802030
CS 201 – 1

PART 3:

Case I (Key is close to the beginning):

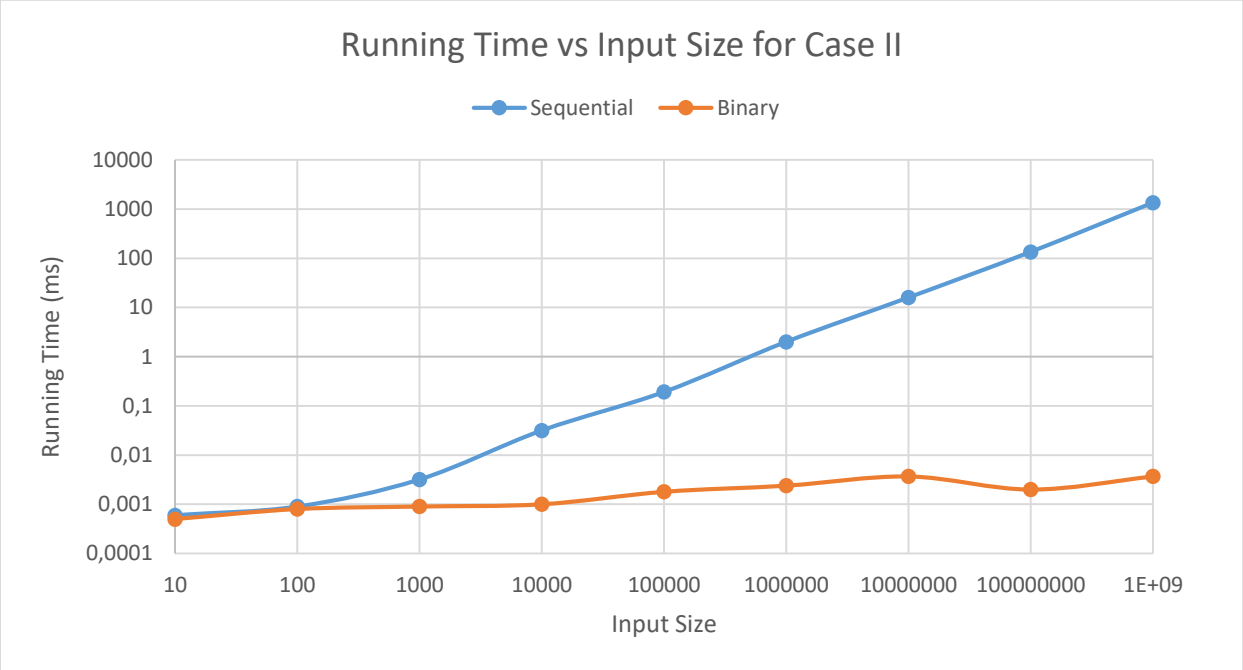
Input Size	Time (Sequential)	Time (Binary)
10	0,0002	0,0007
100	0,0006	0,0007
1000	0,0004	0,0006
10000	0,0005	0,0005
100000	0,0004	0,0017
1000000	0,0005	0,0014
10000000	0,0005	0,0016
100000000	0,001	0,0022
1000000000	0,0007	0,0034



Baykam Say, 21802030
CS 201 – 1

Case II (Key is close to the middle):

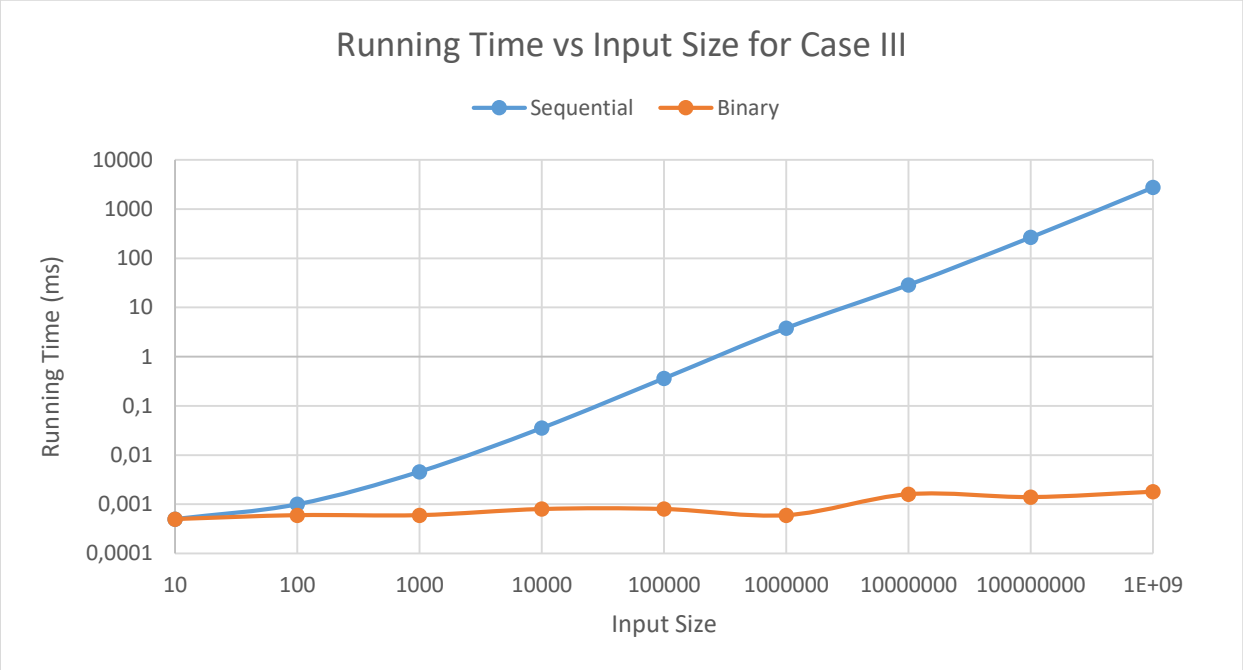
Input Size	Time (Sequential)	Time (Binary)
10	0,0006	0,0005
100	0,0009	0,0008
1000	0,0032	0,0009
10000	0,0317	0,001
100000	0,1921	0,0018
1000000	2,0083	0,0024
10000000	15,917	0,0037
100000000	134,759	0,002
1000000000	1347,6	0,0037



Baykam Say, 21802030
CS 201 – 1

Case III (Key is close to the end):

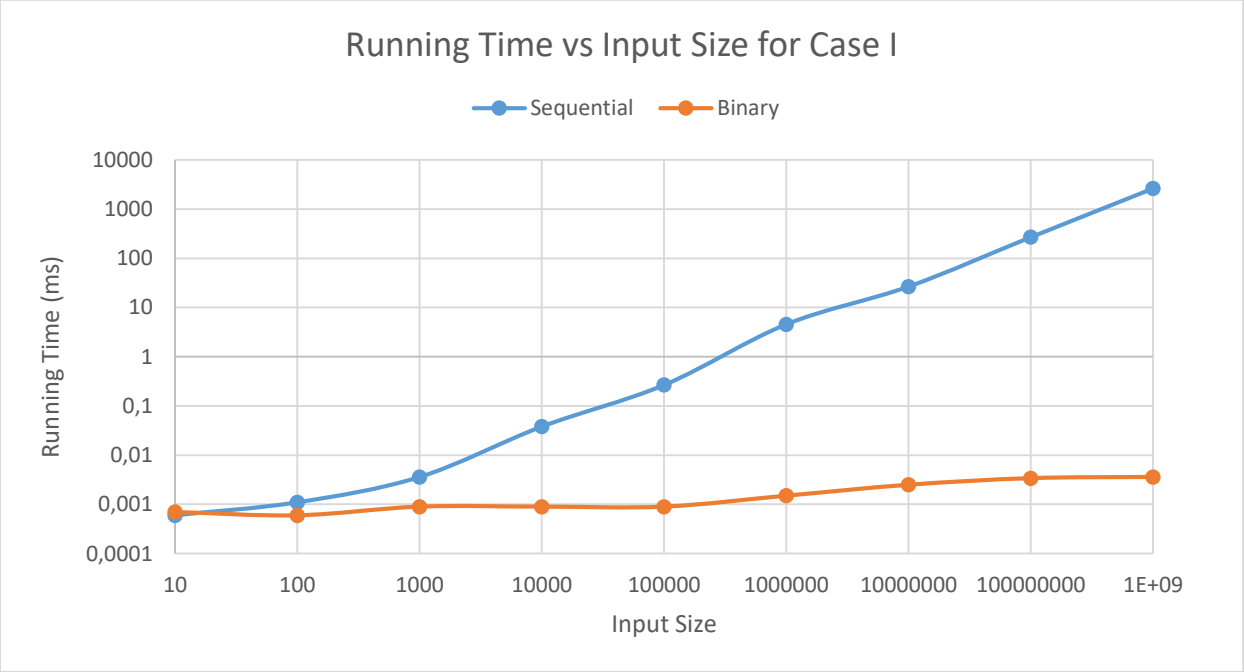
Input Size	Time (Sequential)	Time (Binary)
10	0,0005	0,0005
100	0,001	0,0006
1000	0,0046	0,0006
10000	0,0355	0,0008
100000	0,3624	0,0008
1000000	3,8088	0,0006
10000000	28,7592	0,0016
100000000	265,852	0,0014
1000000000	2737,86	0,0018



Baykam Say, 21802030
CS 201 – 1

Case IV (Key does not exist in the collection):

Input Size	Sequential	Binary
10	0,0006	0,0007
100	0,0011	0,0006
1000	0,0036	0,0009
10000	0,0381	0,0009
100000	0,2668	0,0009
1000000	4,5702	0,0015
10000000	26,7213	0,0025
100000000	269,437	0,0034
1000000000	2631,05	0,0036



PART 4:

For the sequential search the best case is when the key is close to the beginning and the time complexity is O(1), the worst case is when the key does not exist or is at the end and the time complexity is O(N). Therefore, the average case is $\frac{O(1)+O(N)}{2} = O(N)$. For the binary search the best case is when the key is in the middle and the time complexity is O(1), the worst case is when the key does not exist and the time complexity is O(log(n)). Therefore, the average case is $\frac{O(1)+O(\log(N))}{2} = O(\log(N))$.

Baykam Say, 21802030
CS 201 – 1

PART 5:

Device: Lenovo Flex 14 API

Processor: AMD Ryzen 5 3500U with Radeon Vega 8 Graphics @2.10 GHz

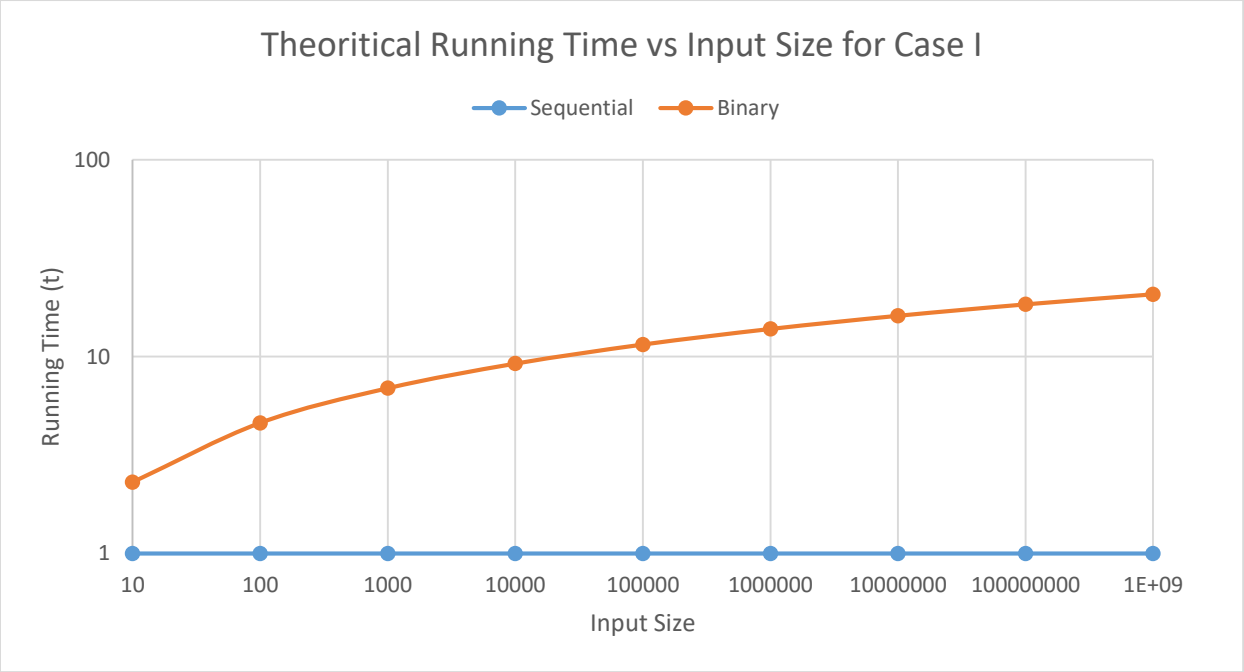
RAM: 20 GB (17.9 GB usable) @2400 MHz

Operating System: Windows 10 Home 1903

System Type: 64-bit OS, x64-based processor

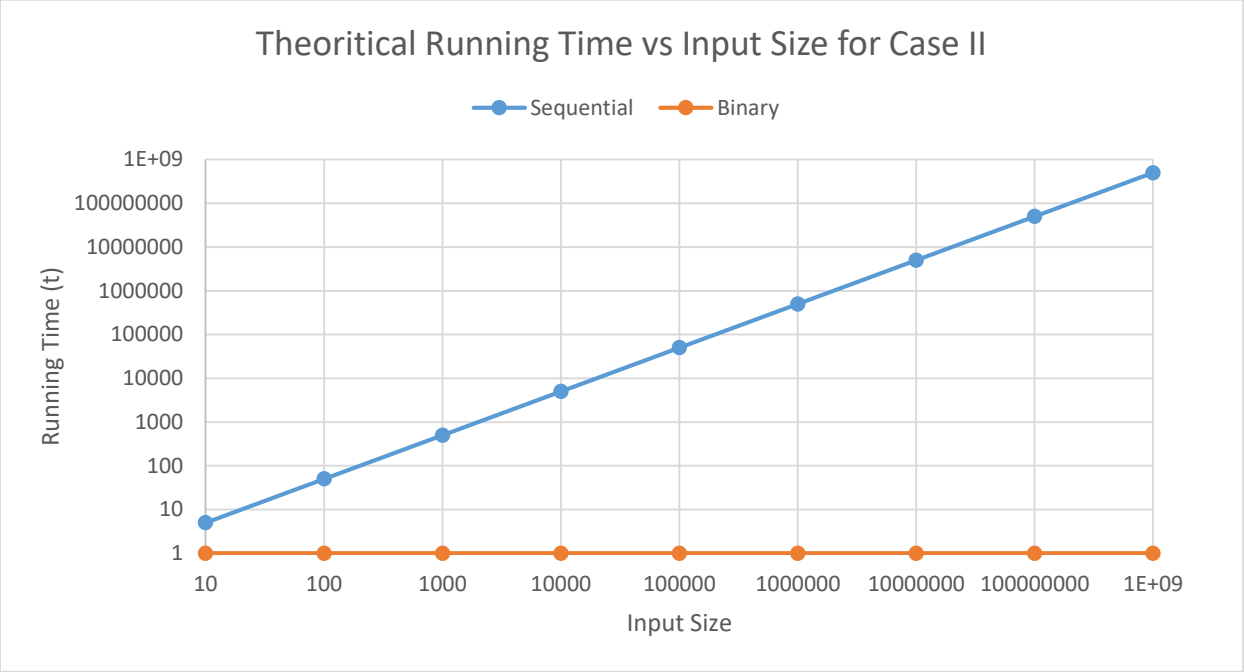
PART 6:

Case I:

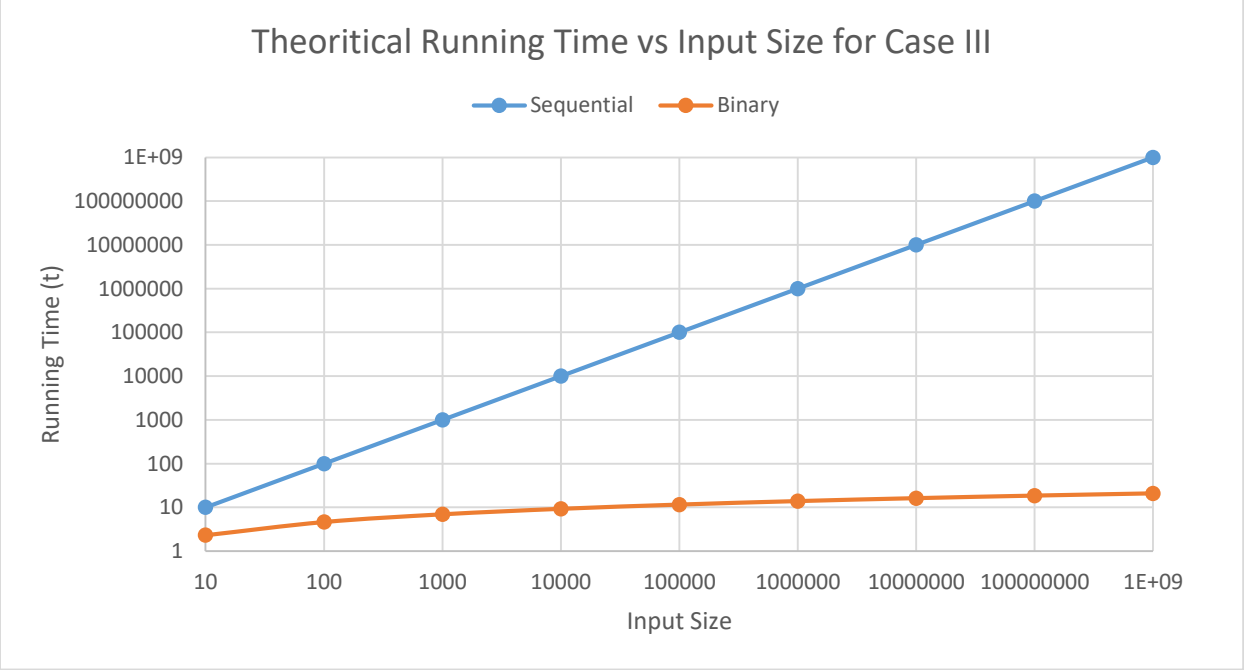


Baykam Say, 21802030
CS 201 – 1

Case II:

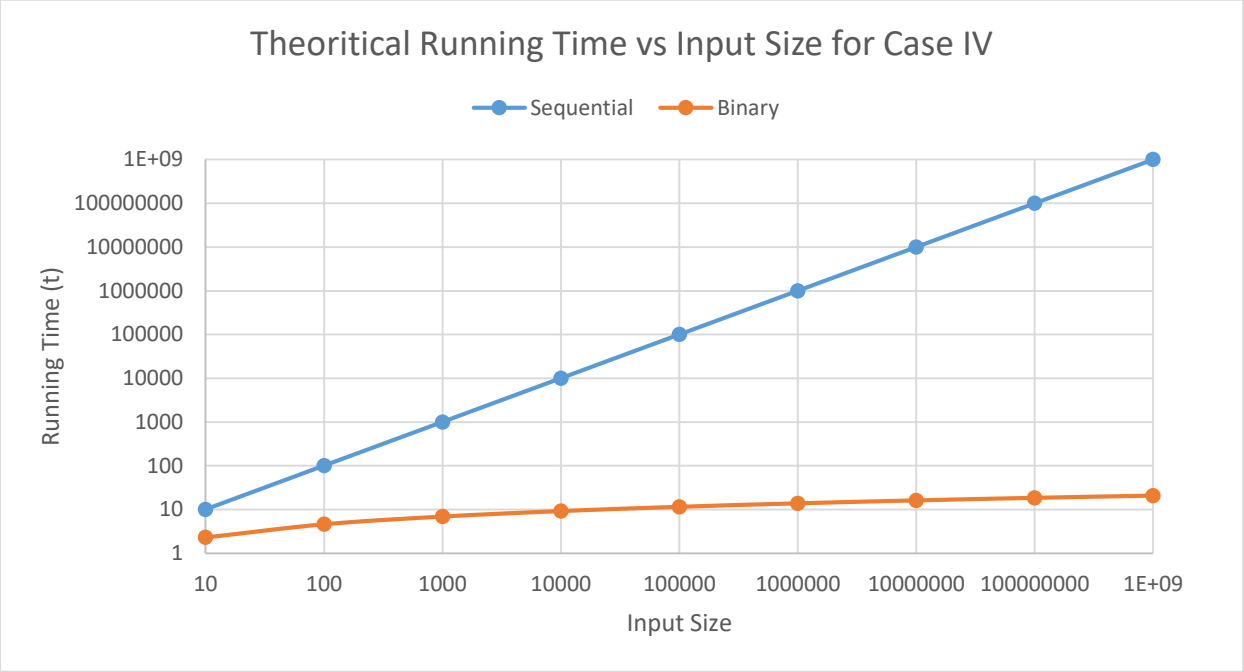


Case III:



Baykam Say, 21802030
CS 201 – 1

Case IV:



PART 7:

The expected growth rates compared to the experiments are similar. For case I, the time for sequential search is almost constant regardless of the input size and the time for binary search increases almost logarithmically. However, because the running time is so small for sequential search, the error margin is quite high compared to other cases. The expected growth rate for sequential search should have been closer to 0. On all three other cases sequential search increases linearly both in the experiment and in theoretical results. For the binary searches, on case II the search time is not constant on the experiment but it should have been constant according to the expected growth rates, on all other cases the increase is close to logarithmically which is similar to the expected growth rates. The errors in the experimental results in all cases are probably caused by the background tasks running on the system which cause fluctuations in the computing power of the system and the errors are higher when the running time is smaller because fluctuations affect the running time more when it is smaller. Therefore, the measured worst case of sequential search is O(N) and the measured worst case of binary search is O(log(N)). They are same as expected worst cases. To have more accurate measurements a virtual processor with a much slower clock speed or a system with less background tasks can be used.