Assignment 1

Garvit Shah [U21CS089]

January 2023

Hardware Details

- Memory: 8 GB 1600 MHz DDR3
- Processor: 1.8 GHz Dual-Core Intel Core i5

Software Details

- Apple clang version 14.0.0 clang-1400.0.29.202
- xcode-select version 2395.

1 Linear Search

1.1 Algorithm

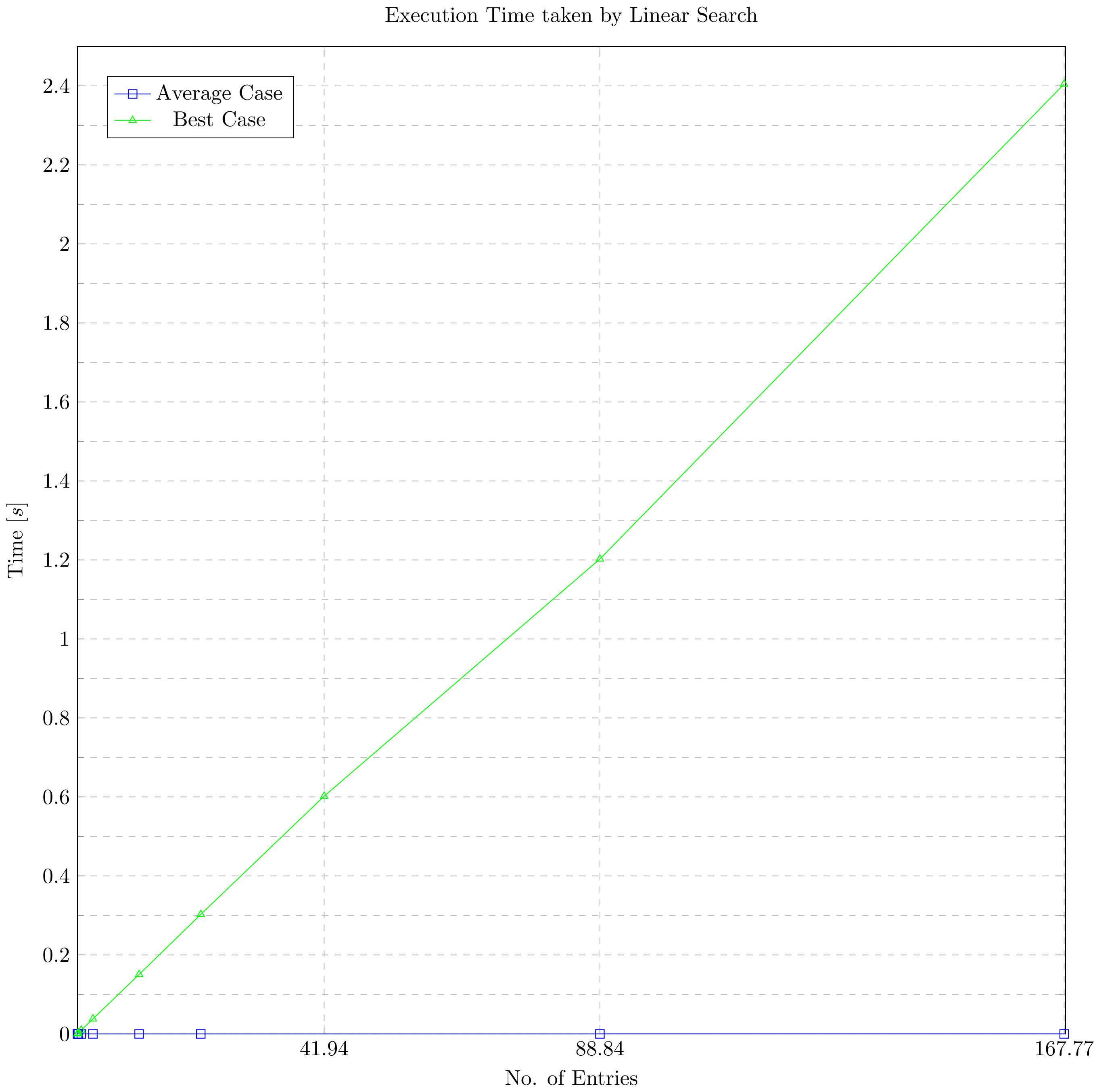
1. Create a file pointer and Open the File
2. Read the number from the file.
3. Compare the number with the value to find.
4. If found exit else continue reading the file.

1.2 Observations

Time Complexity for Linear Search			
File Name	No. of Entries	Best Case	Worst Case
File-1	1,024	0.000003	0.000156
File 2	4,096	0.000003	0.000644
File 3	16,384	0.000003	0.002465
File 4	65,536	0.000003	0.009768
File 5	2,62,144	0.000003	0.037983
File 6	10,48,576	0.000003	0.150615
File 7	20,97,152	0.000003	0.302565
File 8	41,94,304	0.000003	0.601808
File 9	88,83,608	0.000003	1.202313
File 10	167,77,216	0.000003	0.040706

Table 1: Time taken for linear search

1.3 Graph



1.4 Conclusion

The graph for the worst case is a straight line with some slope. A straight line implies that the change in time taken is linearly dependent on the change in the no. of elements in the file, as the line is given by $y = mx + c$. Therefore, it can be concluded that the time complexity for linear search is $O(n)$. Theoretically also time complexity comes out to be $O(n)$. Thus the conclusion matches to the theoretical value of the time complexity.

Time Complexity = $\theta(n)$

2 Bubble Sort

2.1 Algorithm -

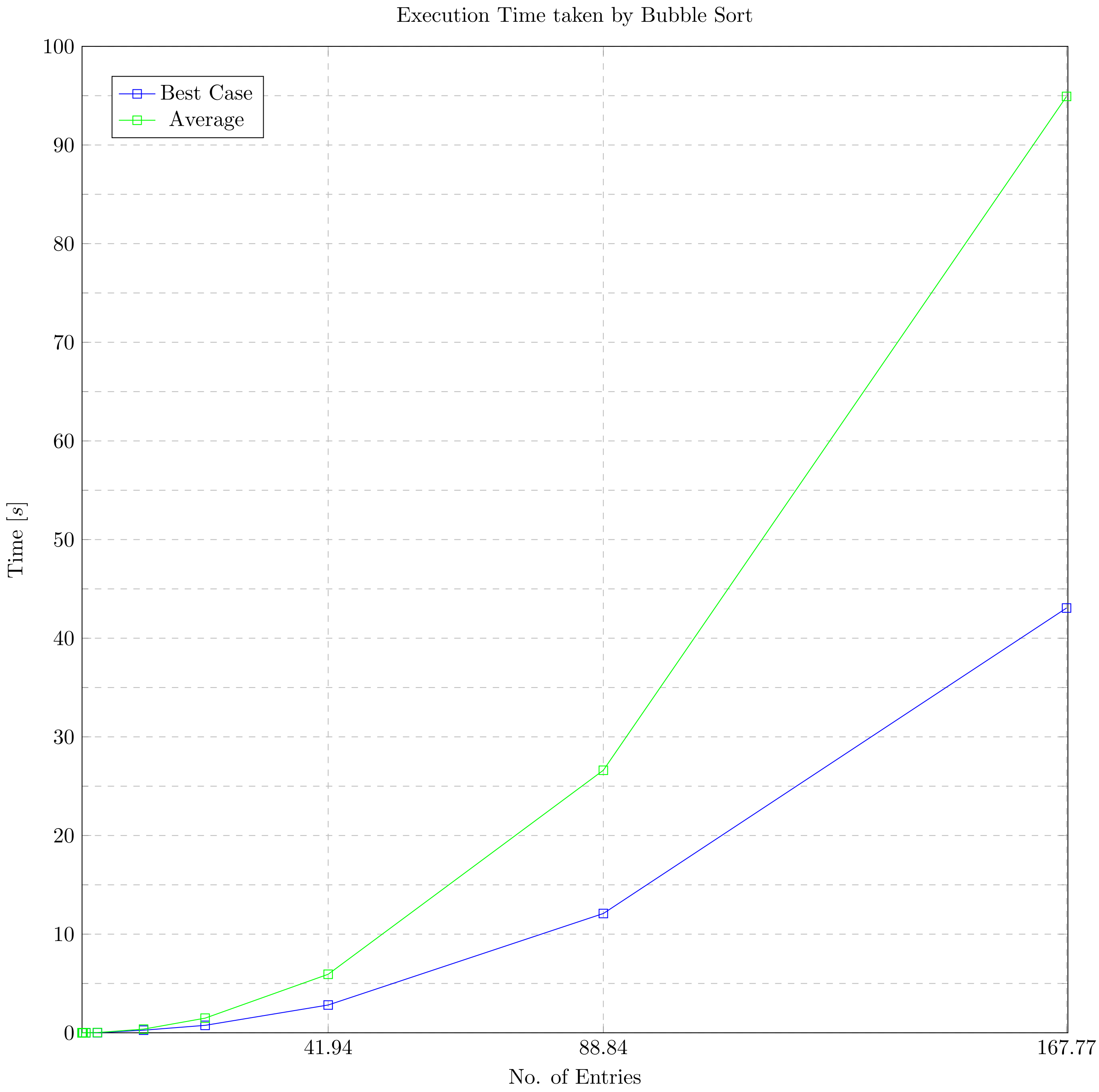
1. Create a file pointer and Open the File
2. Insert the values into an array.
3. Start two loops. Outer loop as number of elements in the array, inner as no. of elements - iterator
4. Swap the elements if the current is bigger than the next.

2.2 Observations

Time Complexity for Bubble Sort			
File Name	No. of Entries	Best Case	Average Case
File-1	1024	0.00348914	0.003746
File-2	4096	0.015065	0.056242
File-3	16384	0.238351	0.902742
File-4	65536	3.822212	14.473051
File-5	262144	80.9	231.703174
File-6	1048576	2635.25	3707.805908
File-7	2097152	7541	14831.596457
File-8	4194304	28163.8	59327.132655
File-9	8883608	120922	266143.099830
File-10	16777216	430619	949243.092676

Table 2: Time taken for bubble sort

2.3 Graph



2.4 Conclusion

The graph for the worst case is a quadratic. A quadratic curve implies that the change in time taken is quadratically dependent on the change in the no. of elements in the file, as the line is given by $y = 4ax^2$. Therefore, it can be concluded that the time complexity for Bubble Sort is $O(n^2)$. Theoretically also time complexity comes out to be $O(n^2)$. Thus the conclusion matches to the theoretical value of the time complexity.

Time Complexity = $\theta(n^2)$

3 Selection Sort

3.1 Algorithm -

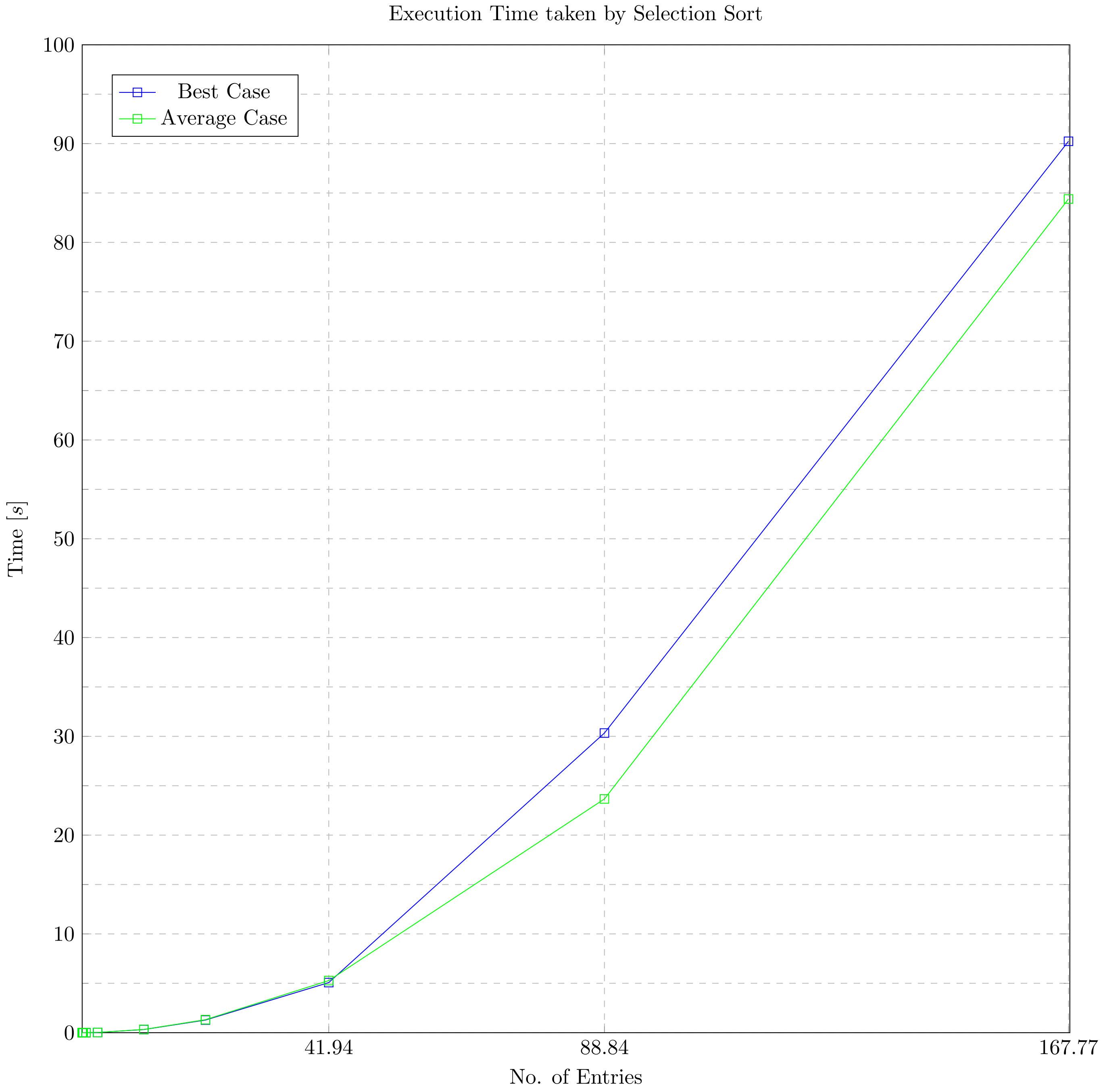
1. Create a file pointer and Open the File
2. Insert the values into an array.
3. Start two loops. Outer loop as number of elements in the array, inner as no. of elements - iterator
4. Find the minimum and place it at the index equal to the iterator.

3.2 Observations

Time Complexity for Selection Sort			
File Name	No. of Entries	Best Case	Average Case
File-1	1024	0.004686	0.004686
File-2	4096	0.027157	0.047667
File-3	16384	0.379485	0.785520
File-4	65536	11.653376	12.791755
File-5	262144	204.846	205.693883
File-6	1048576	3200.145	3295.337373
File-7	2097152	12800.532	13184.193533
File-8	4194304	50732.0834	52742.471015
File-9	8883608	303353.43523	236616.059310
File-10	16777216	902341.14534	843947.960471

Table 3: Time taken for selection sort

3.3 Graph



3.4 Conclusion

The graph for the worst case is a quadratic. A quadratic curve implies that the change in time taken is quadratically dependent on the change in the no. of elements in the file, as the line is given by $y = 4ax^2$. Therefore, it can be concluded that the time complexity for Selection Sort is $O(n^2)$. Theoretically also time complexity comes out to be $O(n^2)$. Thus the conclusion matches to the theoretical value of the time complexity.

Time Complexity = $\theta(n^2)$