Report 1

The aim of this assignment is to analyze the time complexity of 4 different selected algorithms and convert this analysis into graphical form.

For this purpose, I used linear search, bubble sort, binary search and a nested for loop algorithm. The time complexities of these algorithms are O(n), O(n^2), O(logn) and O(nlogn), respectively.

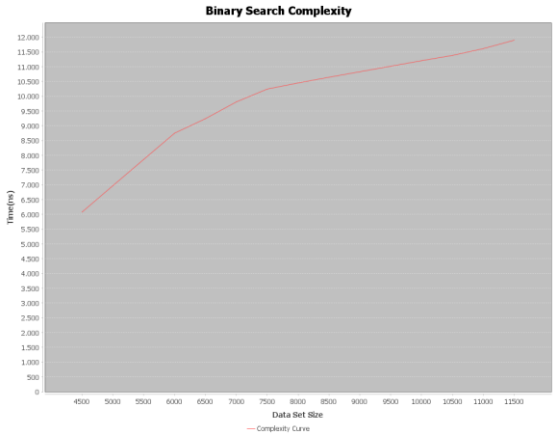
In my project, I used 3 classes other than the Main class. These are MeasureTimeAndOperationCount, Algorithms and Graph. The reason for using 3 different classes is that I want to write a cleaner code and to ensure that the desired result is achieved gradually.

To tell you what I did in these classes in order, in the Algorithms class I wrote the algorithms I chose in order. I also defined the types of operations used in each algorithm and increased their number.

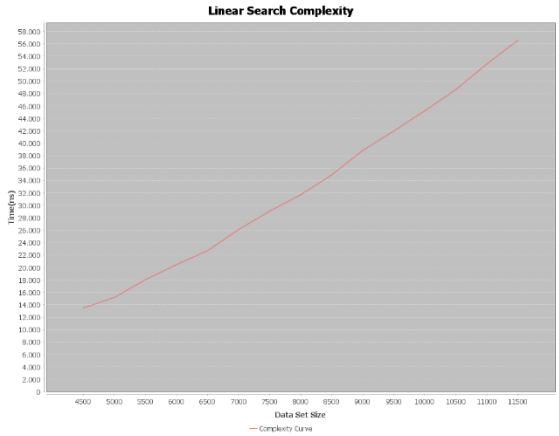
In the MeasureTimeAndOperationCount class, I used the algorithms in the Algorithms class to test the elapsed time and number of operations for each algorithm separately with arrays of 15 different elements. In this class, I used the inheritance property of object-oriented programming to use the algorithms in the Algorithms class. I used the Random class to generate arrays with random elements. To calculate the elapsed time, I used a method that keeps the elapsed time in nanoseconds. The reason for calculating in nanoseconds is that these operations take place in nanoseconds. In this class, I recorded the time passes and the number of operations I obtained. I processed the resulting values more consistently on average, and I tried to use sequences with more elements.

In the Graphs class, I created the graphic creation draft that I needed to visualize the data I obtained. This class also includes the method, which recorded data in the previous class as x and y axes. I used swing and jfreechart classes and methods to do this.

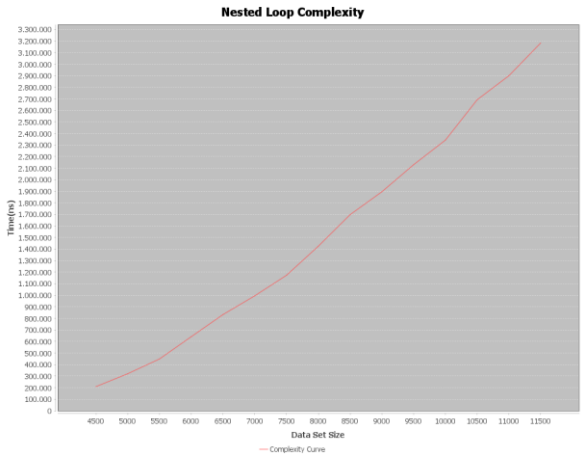
I created an object from the MeasureTimeAndOperationCount class in the Main method to see the outputs and graphs obtained, and I called the methods written for the 4 algorithms separately. So each algorithm has a separate graph. The graphs are as shown below.



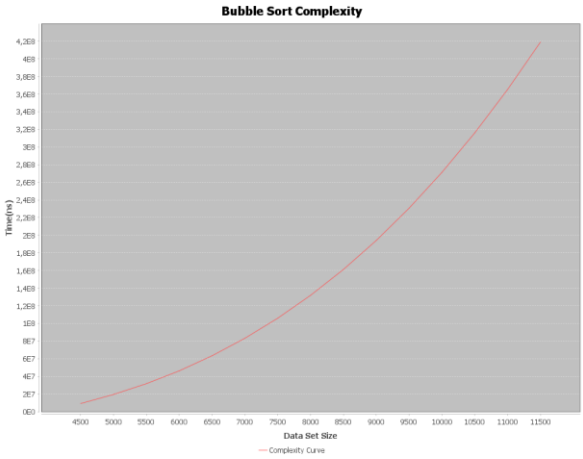
Binary Search Complexity Graph



Linear Search Complexity Graph



Nested Loop Complexity Graph



Bubble Sort Complexity Graph

Theoretically, and according to the results obtained, the algorithm with the lowest complexity should be sorted to the highest.

Theoretically, and according to the results obtained, the algorithm with the lowest complexity should be sorted to the highest; Binary Search - Linear Search – Nested Loop – Bubble Sort

Although the dimensions of the graphs are the same, the value range on the y-axis of each is different. Taking this into consideration, I analyzed the accuracy of the graphs.

The curves obtained in the graphs may vary as they work for random elements each time, but most often give the desired graph.