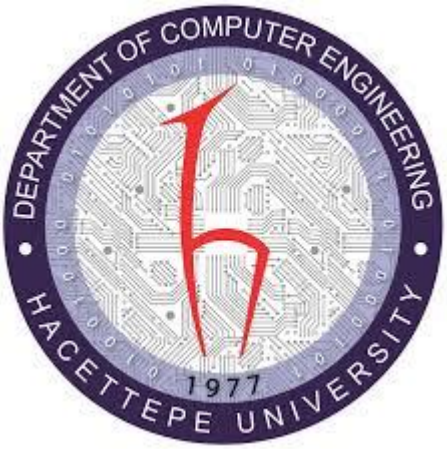


Hacettepe University
Department of Computer Engineering

BBM204 Software Laboratory II

Assignment I

İlayda Atmaca

21827101

Express the idea behind this assignment

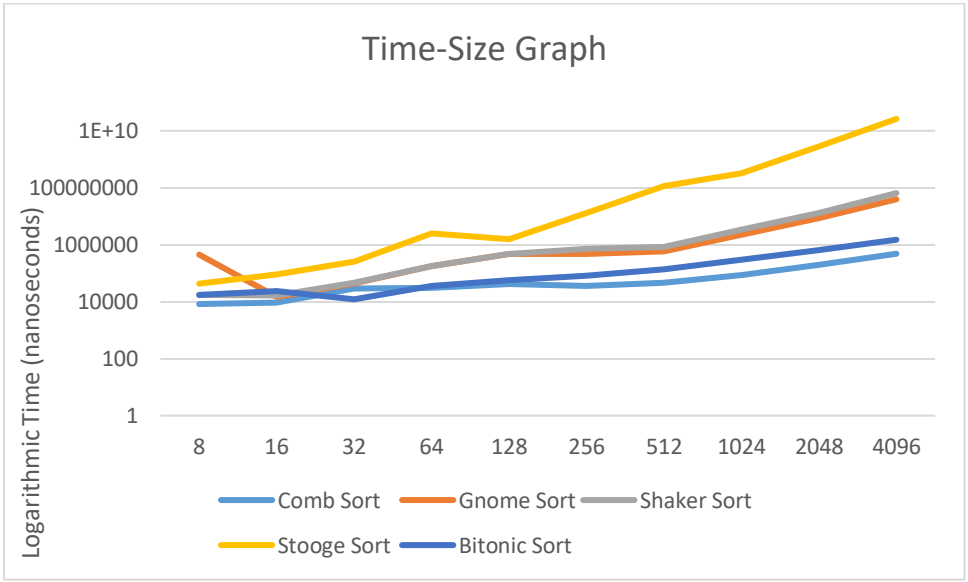
In this assignment, there were 5 different algorithms and the pseudocodes given to us.

The purpose in this assignment is that we wrote the possible cases for each algorithm to find worst cases and average cases complexities. We determined the worst case and average case complexities of each algorithm and thus we reached the execution time data in each complexities. With the size-execution time data that we reached, we had the opportunity to make various graphs. In addition, we were able to compare all algorithms.

And as a result, we determined the relationship between execution time and theoretical asymptotic complexities.

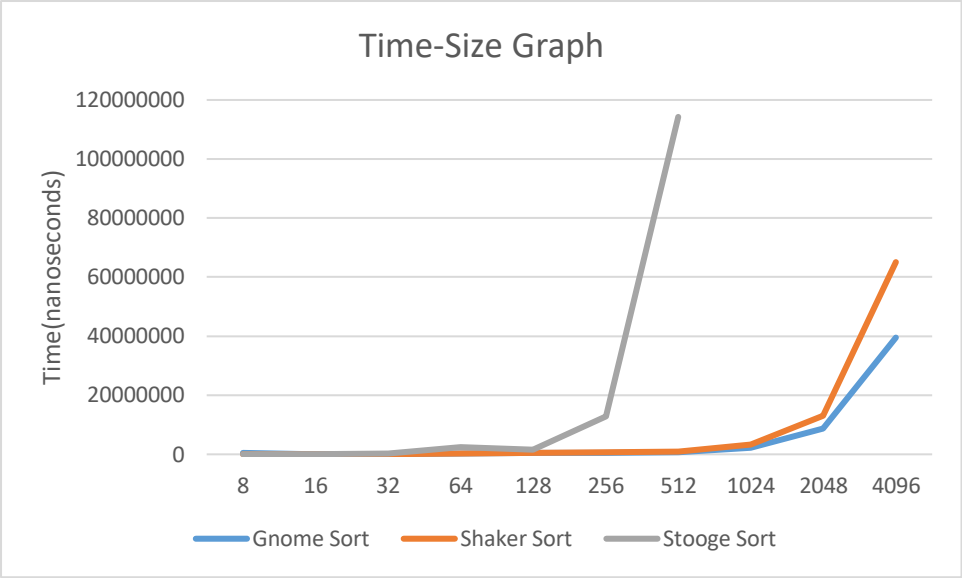
Plot a graph showing the relation between the input size and execution times.

Average Time Complexity



It is quite difficult to show the values on the graph clearly without making the Logarithmic Scale, so I used this expression.

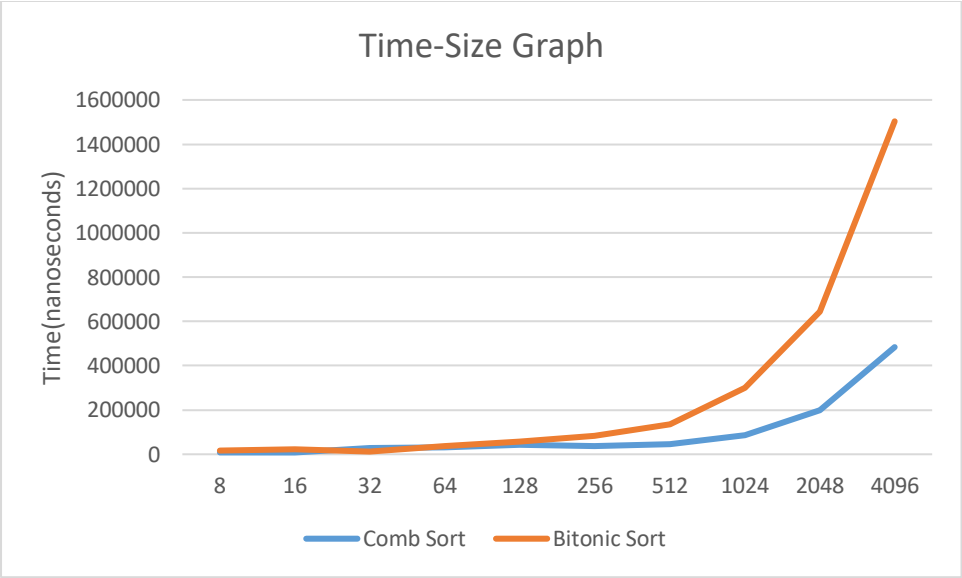
I got the Stooge> Gnome ~ Shaker> Bitonic> Comb comparison.



When I look at the Gnome Sort, Shaker Sort and Stooge Sort graph, I can see average time complexities.

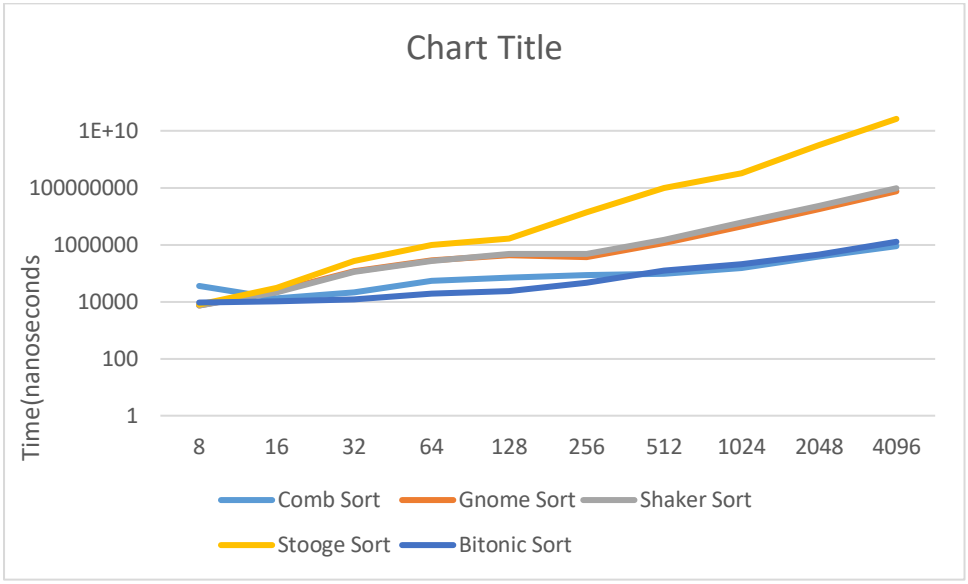
Gnome Sort is like O (n ^ 2) and Also Shaker Sort like O (n ^ 2).Stooge Sort looks very similar to the O (n ^ 3) graph.

As the size grows, the difference between shaker-gnome sort and stooge sort becomes quite obvious.



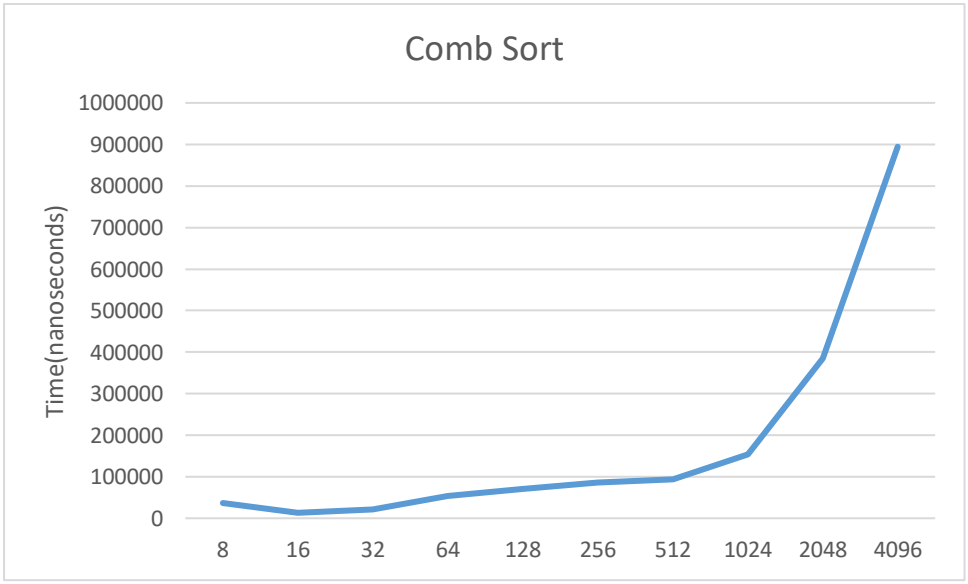
Bitonic is like O(log^2(n)) and with the result that I extracted from the bitonic sort,when I looked at the comb sort, I see quite low on the graph.So the comb average time complexity is much lower than O(log^2(n)). Comb is like O(n^2/2^p).As its size increases,the difference between comb sort and bitonic sort becomes quite obvious.

Worst Time Complexity

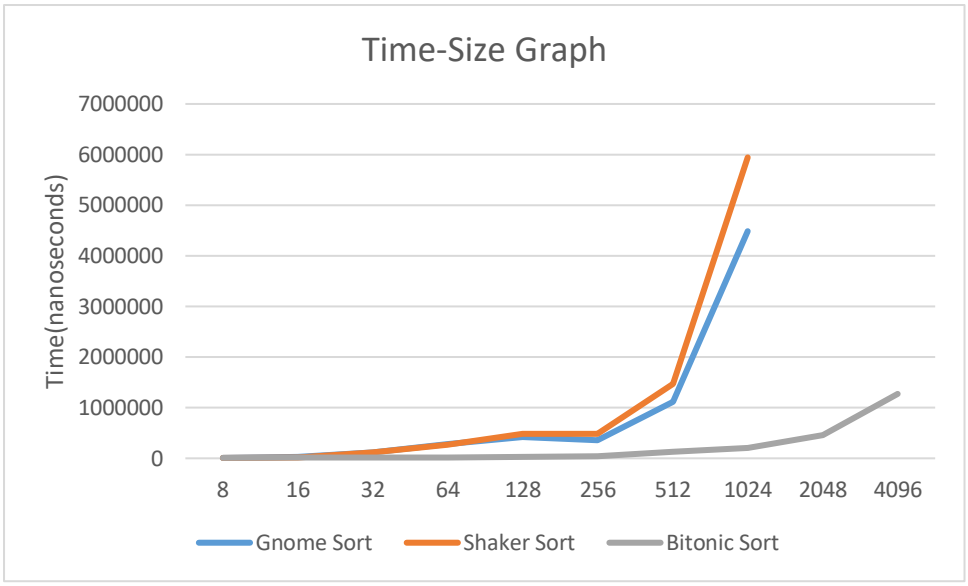


It is quite difficult to show the values on the graph without making the Logarithmic Scale, so I used this expression. When I compared the worst complexities of these 5 algorithms.

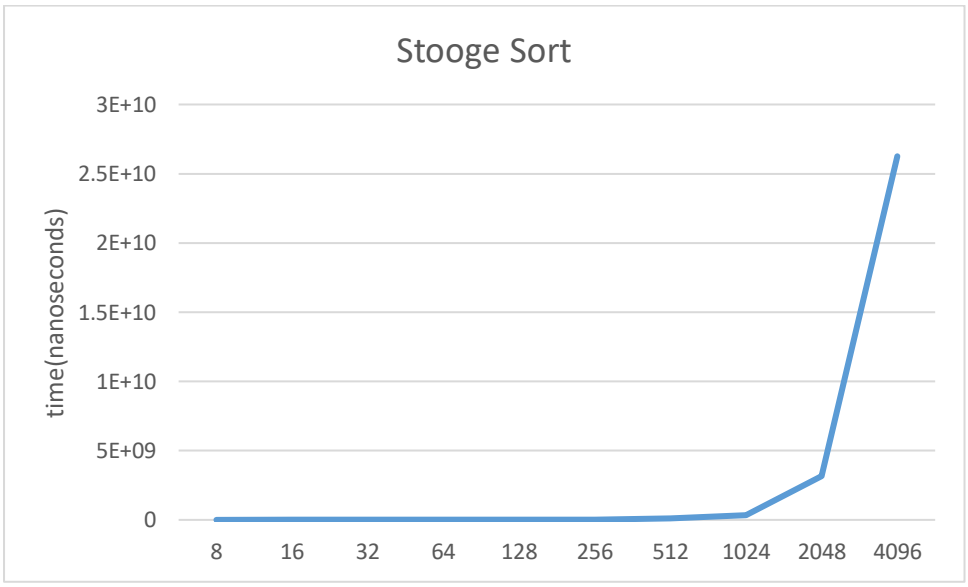
I got the result Stooge> Shaker ~ Gnome ~ Comb> Bitonic



After the looking the graph of comb sort,it is like O(n^2).



When I look at the graph with these three sorts, I see that the Gnome Sort and shaker sort are very close and their worst time complexity is similar to O (n ^ 2). Bitonic Sort looks like O (log ^ 2 (n)).



When I look at the time values in the stooge sort graph, I see that it is quite large when I compare it with other sorts and it looks like O (n ^ 3).

Your report has to include table as shown in Table 3, figures as shown in Figure 1 and your analysis for the experiment.

Worst Time Complexity

	Comb Sort	Gnome Sort	Shaker Sort	Stooge Sort	Bitonic Sort
8	36059	7369	7360	7940	9360
16	13170	23319	21169	30280	10139
32	21869	120660	113169	271060	12049
64	54129	282879	274630	984260	19449
128	69909	425440	484350	1614840	23979
256	86399	359030	486959	13690420	46280
512	93260	1121559	1464059	97775580	126590
1024	153629	4486470	5943539	3,22E+08	203520
2048	385460	17700610	23070599	3,14E+09	453249
4096	894420	75193120	96142849	2,63E+10	1272660

Average Time Complexity

	Comb Sort	Gnome Sort	Shaker Sort	Stooge Sort	Bitonic Sort
8	8290	442759	17170	43120	17579
16	9420	14769	16619	92280	23620
32	28379	44710	47480	259299	12160
64	31319	177350	180609	2535680	35730
128	42519	474140	475560	1569839	57970
256	36669	472209	731929	12945980	83159
512	46329	598189	851600	1,14E+08	135619
1024	85430	2303110	3381559	3,3E+08	298830
2048	199869	8695269	13159149	2,86E+09	643769
4096	483870	39528480	65083729	2,62E+10	1503320

How you designed your experiments?

First, I wrote the code for each algorithm using the pseudocode given to me and created a class for each algorithm.

```
int[] size={8,16,32,64,128,256,512,1024,2048,4096};
```

By creating int [] size array (), I saved various size values in this array to calculate an accurate execution time.

While I was creating an object for each size, I created 2 attributes. I created these attributes in my Person Class and defined the names as String name and Integer personID. Also, I used set and get methods for these attributes in my Person class.

I assigned these String name and Integer personID randomly. I did the task of assigning the string randomly in the randomString () method.

I saved my values in the ArrayList named persons to be able to use it in every algorithm.Then I called this random arraylist for each algorithm in the same order.

But since I don't know in which situations all my algorithms are best, worst or average complexity, I evaluated these 3 cases in my code. And that's why I created various methods to satisfy the Ascending Order, Descending Order and Random requirement.

I used timeCalculationandRunningforSortings () in my Calculation Class to calculate the execution time of each algorithm.

I created the descendingOrder () method for the Descending case and used this method to sort by my arraylist.For the ascending sort, I also created the ascendingOrder () method and created this method to sort my arraylist as descending sort.

Thus, I was able to evaluate these 3 cases for each algorithm and this enabled me to find Best, Worst, Average Time complexity of each algorithm. Also I runned a lot of time my code to finding the accurate time for each algorithm.

Finally I printed the execution time of each algorithm in nanoseconds for all three cases.When I recorded these all situations on the graph, I saw that what is the worst and average complexities for which algorithm.

You are expected to determine related inputs to observe stability of sorting algorithm and interpret them as which sorting algorithms are stable

First of all, I created the stability () method in my Calculation Class to measure stability within these 5 algorithms. And I use the String name attiribute that I created earlier to measure stability.

Later, I created 2 arrays named name and id in this method to understand stability exactly, and I recorded them in my persons arraylist.

```
String[]name={"Andrew","Battle","Chen","Fox","Fruia","Gazsi","Kanaga","Rohde"};
int[] id={3,4,3,3,1,4,3,2};
```

I have sorted my ArrayList persons by string in the stringCompare () method. After this method, I obtained an array sorted by string.I runned this arraylist for each algorithm. So I got the following result from each algorithm.

For Comb sort, I came to the conclusion that it is not stable.

For Gnome Sort, I measured the information that Gnome Sort is stable.

For Shaker Sort, I measured the information that Shaker Sort is stable.

For Stooge Sort,I came to the conclusion that it is not stable.

For Bitonic Sort, I came to the conclusion that it is not stable.

Briefly here, Gnome and Shaker is stable. Stooge, Comb, and Bitonic are not stable.

Advantages and Disadvantages of Stability

Advantages

Stable sorting algorithms preserve the relative order of equal elements while unstable

sorting algorithms don't.In other words, stable sorting maintains the position of two equals elements relative to one another.

Stable sort will always return same solution (permutation) on same input.

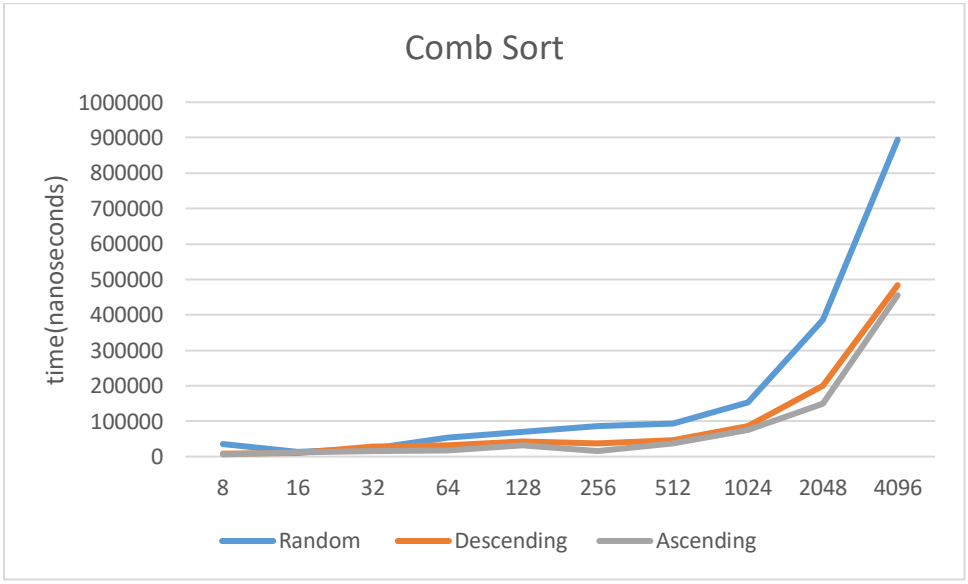
Unstable sort may yield different output for the same input from run to run. Such behavior is unsuitable for some applications, for example for client-server applications where the server uses pagination for output and performs a new search-and-sort for every new page requested by the client.So stable sort is necessary to some of applications.

Disadvantages

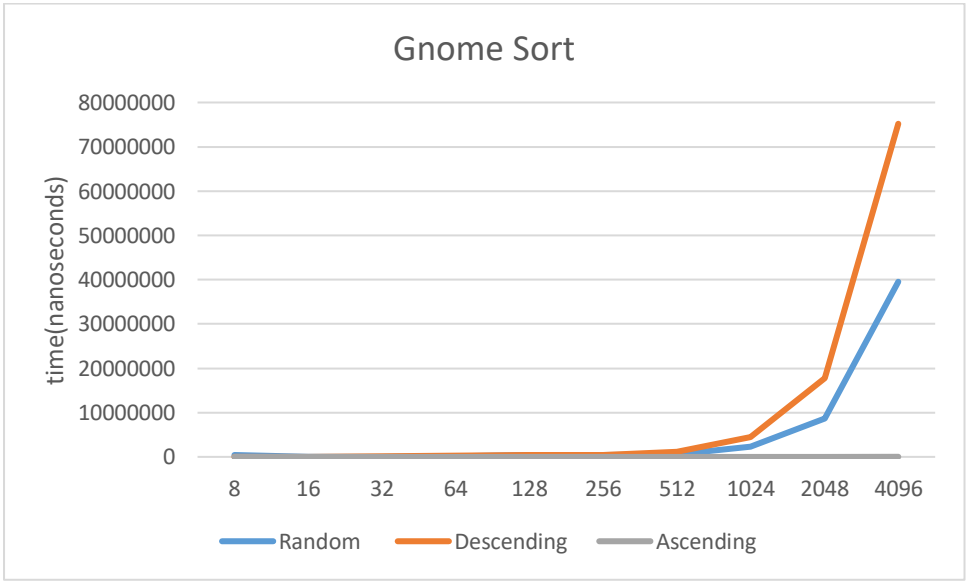
We can modify unstable sorting algorithms to be stable. However, we may use extra space to maintain stability.

An unstable sort will normally be slightly faster than the stable version of the same algorithm.

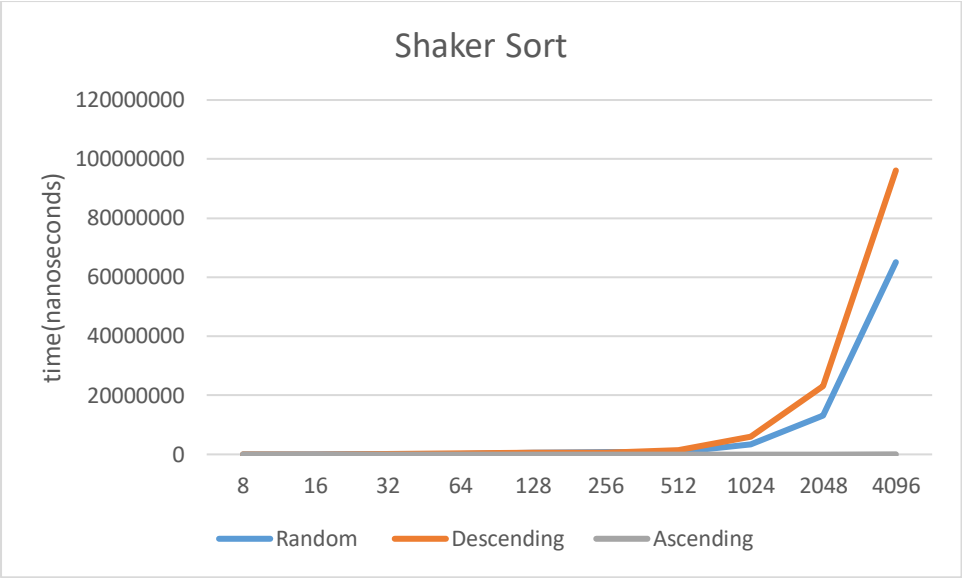
Show your analysis in detail for each algorithm.



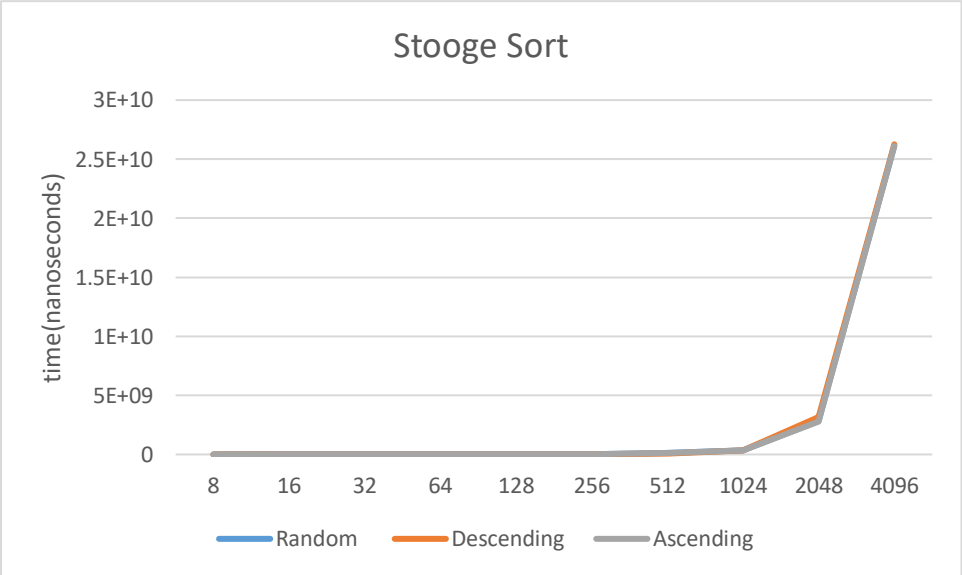
After doing this graph,I reached the information that average time complexity is doing with descending.Worst time complexity is doing with random.



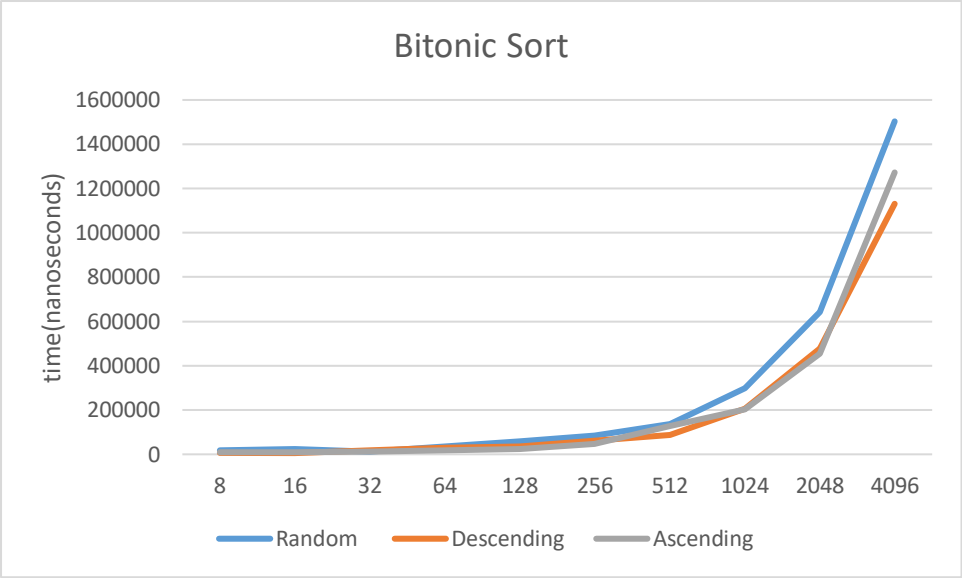
When I graphed to find time complexity for gnome sort, I saw that Descending is worst time complexity and random is average time complexity.Average and worst has same time complexity.Ascending is the best time complexity.



For shaker sort, I understanded that Descending is worst time complexity and random is average time complexity.Average and worst has same time complexity.Ascending is the best time complexity.



In stooge sort, best ,worst and average time complexity have the same value and this value is O (n^2).



In bitonic sort, best ,worst and average time complexity have the same value and this value is O (log ^ 2(n)).

Memory Requirements

The amount of extra memory required by a sorting algorithm is also an important consideration.Extra memory is very important in our algorithm, and we know that in place sorting algorithms do not require additional memory.

When we look at our 5 algorithm to understand whether it is inplace or not, we see that all of them are in place sorting algorithms.

It means that they don't need to create auxiliary locations for data to be temporarily stored. They do not require sufficent memory for another copy of the input array. Also We updated input sequence only through replacement or swapping of elements on each algorithm.