Hacettepe University
Computer Engineering

BBM 204 SOFTWARE LABORATORY I

-2021 Spring Semester-

ASSIGNMENT I REPORT

Ahmet Dursun AVCI

21827129

Contents

1. Explanation Of Assignment

1.1-> Explanation:İn this assignment, we are expected to analyze some sorting algorithms such as Gnome,Stooge etc. We should implement their source code after that, according to some inputs we should calculate execution times then we should compare each of them. In this assignment, we will understand which algorithm is better to use.

1.2->Gnome Sort:

Gnome Sort Works like Insertion sort. So, it is really easy to implement it. Firstly it compares sequantial elements of array than try to find correct position of element.

1.3->Bitonic Sort:

Bitonic Sort is parallel sort algorithm. This algorith made me suprised in terms of execution style. It Works like “Divide and Conquer Approach” method.

1.4->Comb Sort:

Comb sort is improves on bubble sort. It Works like it, Main differences is Bubble sort compare sequantial elements till list is ordered, However Comb constantly divide the Size of array with Shrink which is 1.3, than compares elements (a[i]-a[i+size(a)/1.3]) like that.

1.5->Stooge Sort:

Stooge is the worst algorithm among them. It is really slow and uses lots of memory. Because, it divides array two different overlapping parts. İt recall the function for 2/3 of array three times. So, it is really inefficient algorithm.

1.6->Shaker Sort:

Shaker sort goes from left to right then it goes from right to left. While wandering through the arrays, it compares sequantial elements till array is sorted.

2. Table And Graphics Of Algorithms For Average Case

2.1-> Explanation:In this experiment, I tested the algorithms with 10 different number of array size. Then, I recorded execution times on the table.

After that I sketched the graphs.

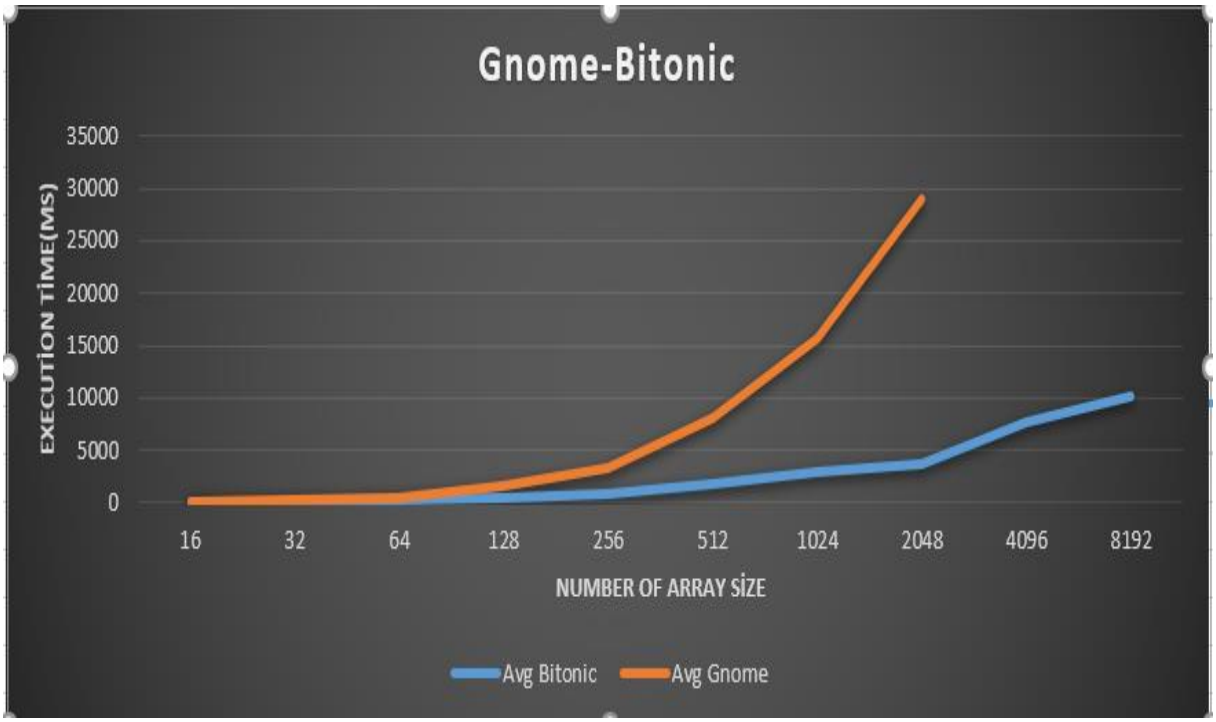
2.2->Table

Algorithms/n	16	32	64	128	256	512	1024	2048	4096	8192
Gnome	0,053	0,177	0,534	1,521	3,340	8,134	15,635	28,995	50,759	183,905
Bitonic	0,036	0,160	0,340	0,434	0,871	1,763	2,862	3,646	7,740	10,073
Comb	0,017	0,063	0,149	0,220	0,473	0,930	2,337	4,460	14,798	11,176
Stooge	0,274	0,610	1,958	3,807	21,055	127,428	219,028	1563,257	13562,733	138015,790
Shaker	0,035	0,141	0,275	1,588	3,465	6,341	9,556	26,021	129,940	306,051

2.3->Graphs

Firstly, I have to say that I compares all graph of algorithm with graph of Gnome. Because, Calculate the avarage of Gnome is really easy. Since it is variation of insertion sort, we can know that its avarage time execution equals O(n^2).

2.3.1->Gnome-Bitonic

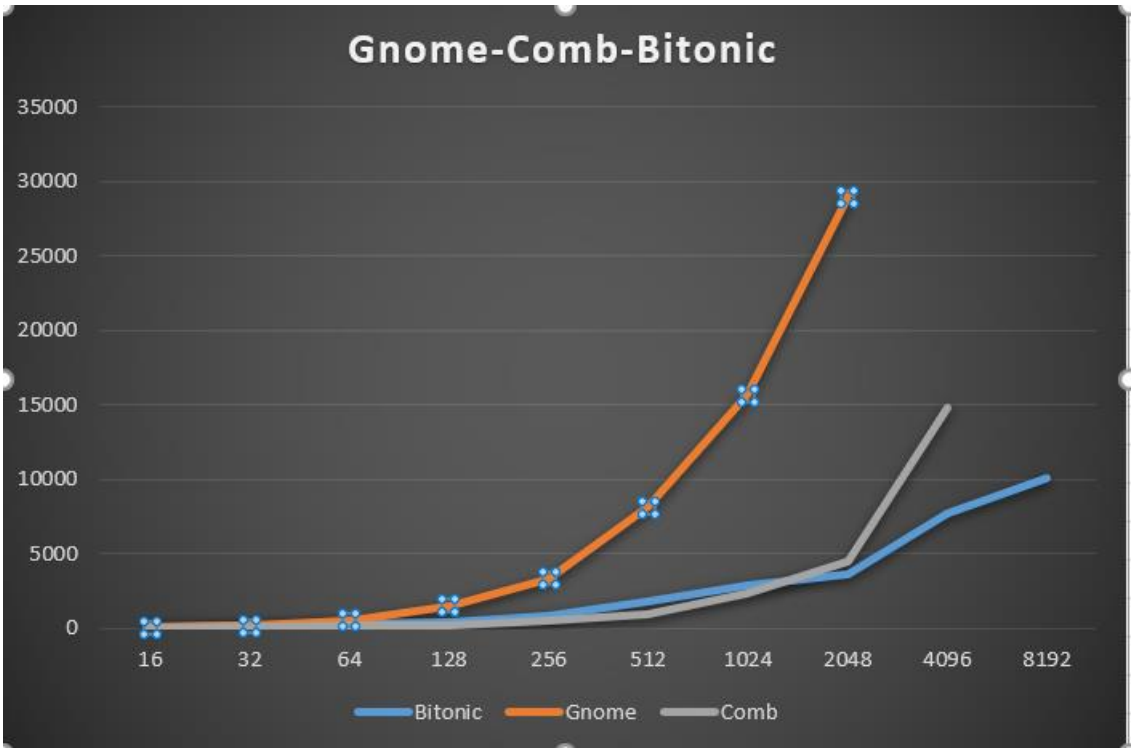


If we compares their algorithms we can realize that Bitonic sort's avarage case must smaller than O(n^2).
In addition, we can understand it is not linear.
Firstly I can say it's big O notaion must be O(nlogn), However I compared it with the merge sort which has O(nlogn) notation. I understood it has bigger complexity than merge sort. So I can say it must have $O(n\log^2 n)$.

Bitonic Has $O(n\log^2 n)$ time complexity.

Gnome Has $O(n^2)$ time complexity.

2.3.2->Gnome-Comb-Bitonic



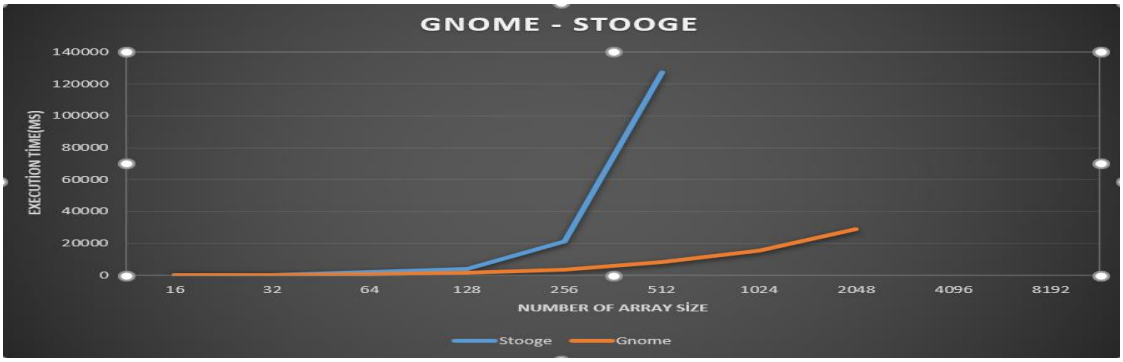
If we compares their algorithms we can realize that Comb sort's avarage case must smaller than O(n^2).
In addition, we can understand it is not linear. Also if we compares with bitonic's graph, it must larger than $O(n\log^2 n)$.

Comb has $O(n^2/2^p)$ time complexity. P is number of increments.

Bitonic Has $O(n\log^2 n)$ time complexity.

Gnome Has $O(n^2)$ time complexity.

2.3.3->Gnome-Stooge

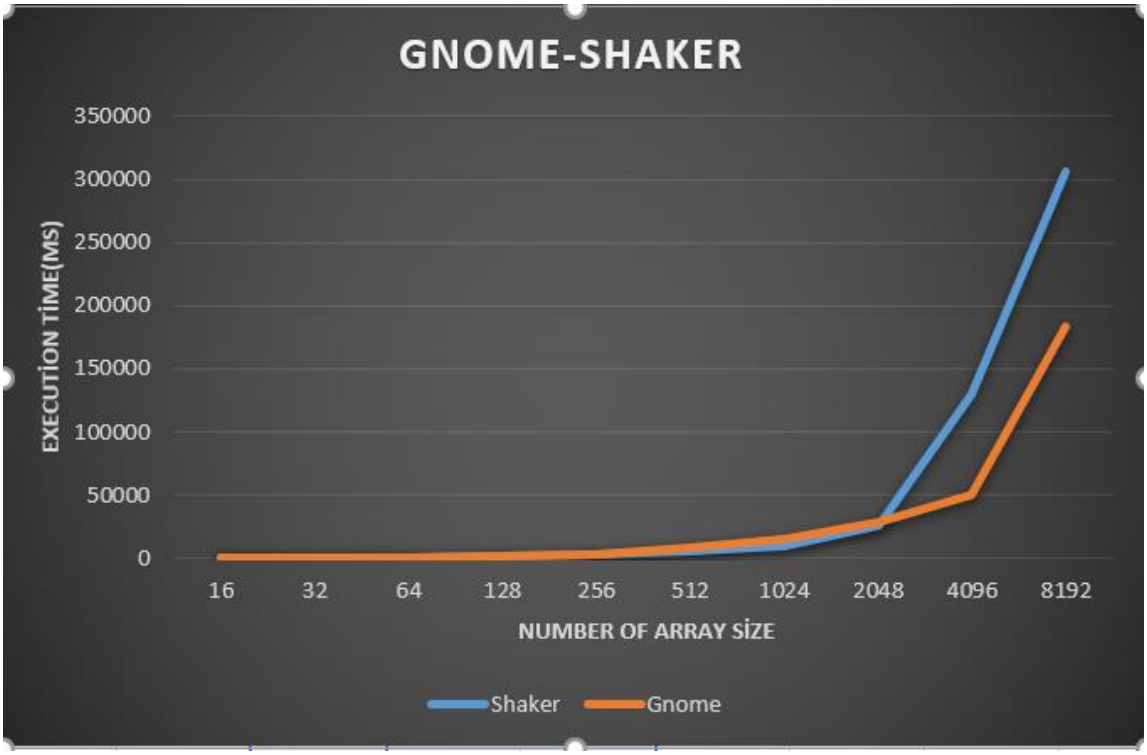


When we look at the graphs it must has bigger than $O(n^2)$ complexity.

If we look at its source code, we see that it constantly recalls 2/3 of the strings. So its complexity is O (n ^ (log3 / log1.5) approximately O (n ^ 2.71).

Stooge has O(n^2.71) complexity.

2.3.4>Gnome-Shaker



the graphics are very close to each other. So shaker must has $O(n^2)$ complexity too. The reason of this small differences is coefficients.
İf we look their tilde notation we can see the differences.

Shaker has $O(n^2)$ time complexity.

3.Table And Graphics Of Algorithms For Worst

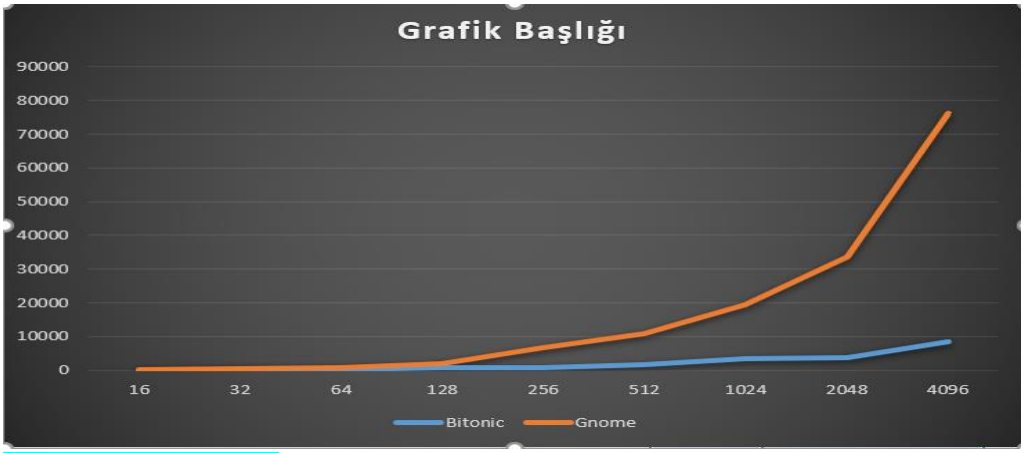
3.1->Explanation: If we look at the Gnome's source code, we can see that if it is used for descending order it makes n*(n-1)/2 times comparision it means it's worst case is descending order and it's complexity is $O(n^2)$. I made my comprasions by using Gnome's graph.

3.2->Table

Algorithms/n	16	32	64	128	256	512	1024	2048	4096
Gnome	0,076	0,375	0,738	1,982	6,845	10,847	19,428	33,492	76,212
Bitonic	0,054	0,128	0,292	0,815	0,920	1,548	3,529	3,844	8,621
Comb	0,027	0,073	0,122	0,254	0,338	1,248	4,736	5,254	12,644
Stooge	0,252	1,151	1,985	3,757	20,750	143,829	217,752	1688,211	14731,658
Shaker	0,086	0,225	0,552	2,705	8,658	12,224	18,910	53,473	110,730

3.3->Graphs

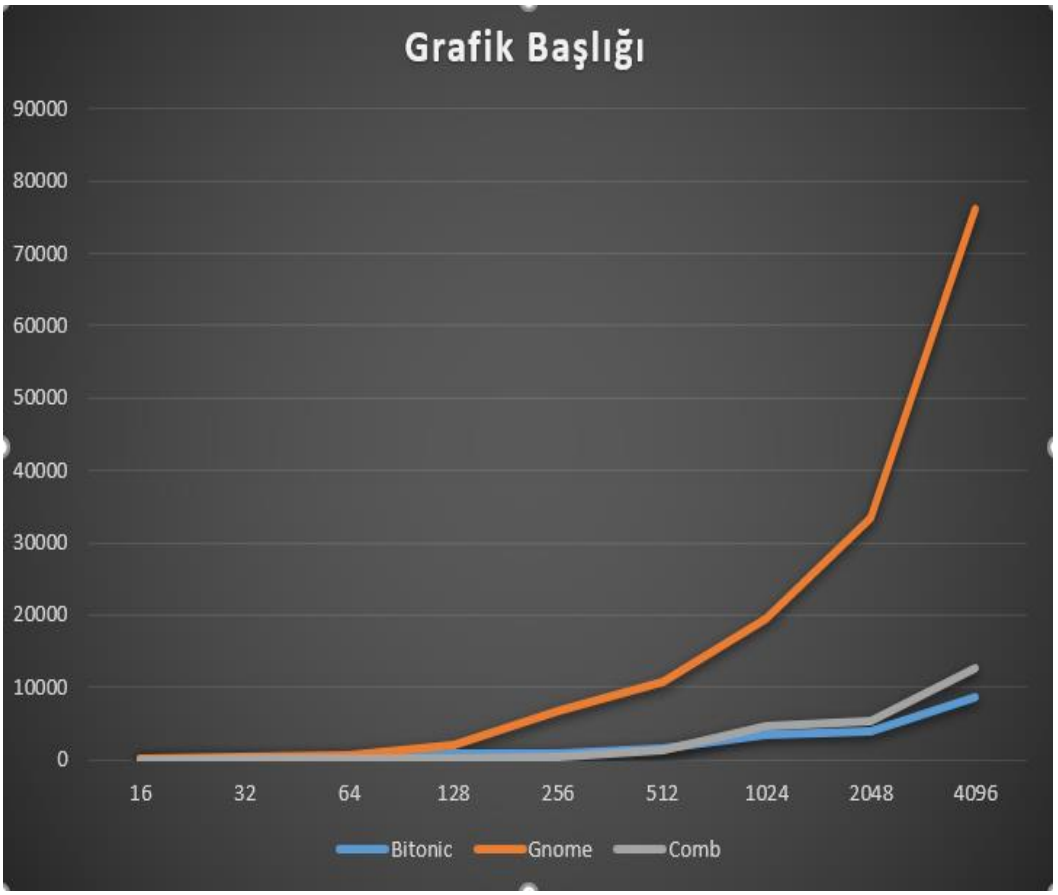
3.3.1 Gnome-Bitonic



Worst case of Bitonic : I tested lots of situations for bitonic, but I couldn't find anything which makes it's execution time is longer. For instance, descending order, half descending order, in-line order etc. Then I realized that it's worst-avarage-best cases complexity same like merge sort. Than I produced randomly inputs and calculate it's execution times. $O(n\log^2 n)$

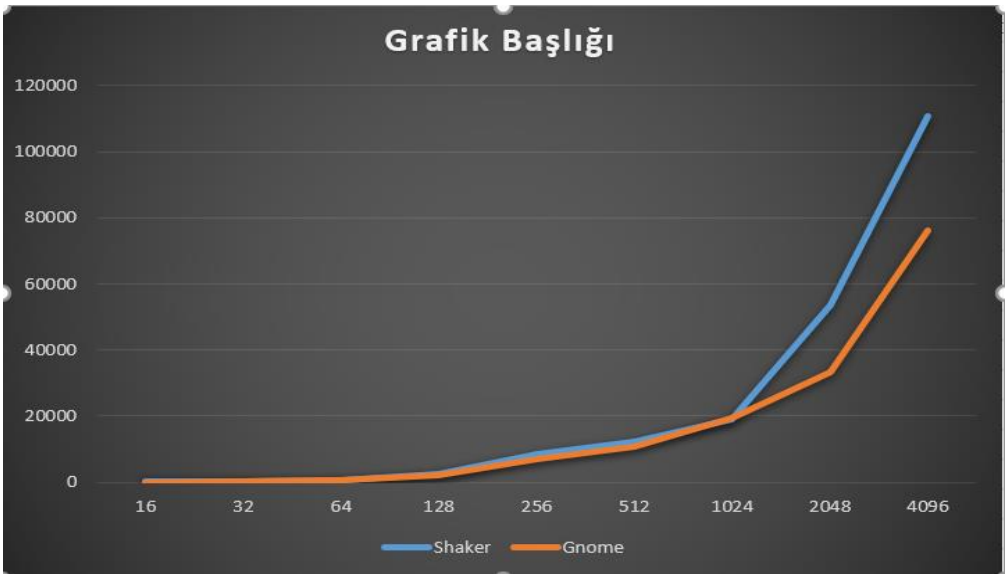
Worst case of Gnome: when we looked at source codes of gnome, we can see it's worst case is descending order like insertion sort. We have already know that it is variation of insertion sort. So I calculated it's execution times and find that its worst case complexity is same with it's avarage case complexity. $O(n^2)$

3.3.2 Gnome-Bitonic-Comb



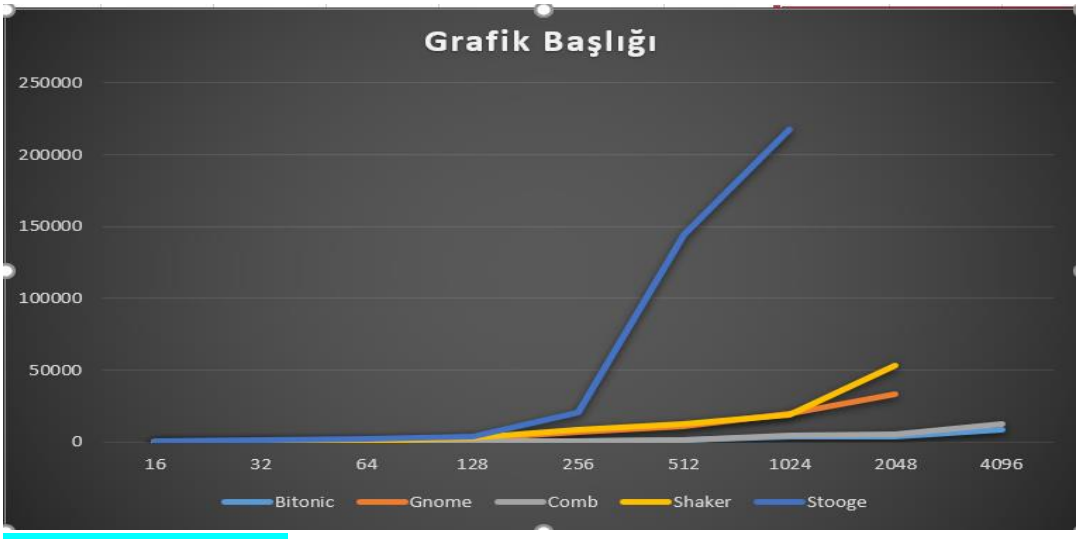
Worst case of Comb : Actually, I couldn't find anything which makes it's execution times is better or worse. But I finded it's avarage case, we know that neither it's avarage case is lower bound nor upper bound. Since it's avarage case complexity is $O(n^2/2^p)$ we can say it's upper bound is n^2. So it's big oh notation is $O(n^2)$.

3.3.3 Gnome-Shaker



Worst case of Shaker: Shaker Sort is working like buble sort, so we can simply say that the worst case is descending order. So i generated inputs which are in descending order. Then I calculated the execution times. Since we have already know the Gnome sort complexity, we can easily say that their complexities are same.
$O(n^2)$

3.3.4 Stooge-All



Worst case of Stooge: Stooge sort is really one of the most inefficient algorithms. Due to the fact that Stooge doesn't checked list is in-line or not, It always Works like worst. So, it's best-avarage-worst case complexties same. In this experiment I generated random numbers, then i calculated the exxecution times. O (n ^ (log3 / log1.5) approximately O (n ^ 2.71).

4. Stability

4.1 What is stability? What is it's advantages and disadvantages?

If i explain what stability is, i would like to give some examples. sometimes we want to sort some object by consider it's attribute, then we can want to sort it with other attributes, However we would like to make some object which equals each other in terms of this attribute in order with old attribute. (Hocam bunu İngilizce açıklaması zor oldu biraz. Şunu demek istedim: Mesela siz bize huffman encoding assignmentı vermiştiniz. bizim harfleri freqlerine göre sıralamamız istenmişti. Ancak freqleri aynı olanların da kendi aralarında alfabetik sırada olması

istenmişti. Eğer biz stabil algoritmalarla ilk harfe göre sonra freqlere göre sıralarsak, freqleri aynı olanlar stabiliteden dolayı aralarında alfabetik sırada kalır ancak stabil bir algoritma kullanmazsak bunlar karışır tekrardan.)

4.2 Stability Of Algorithms

4.2.1 Stability of Gnome Sort

I checked the gnome sort, than i realized that objects which are equals each other were staying same order after the sorting. Also i realized that if the algorithms compares the sequantial items so thet do not break their turn. Gnome sort is comparing the sequantial items. İt means Gnome Sort is Stable.

4.2.2 Stability of Bitonic Sort

I checked the inputs after the sorting, Bitonic sort is breaking their turns so it is unstable algorithm. Also we can understand it by looking it's source code.

4.2.3 Stability of Comb Sort

Comb Sort Compares spaced elements and swap them so it is breaking the stability. Also I checked it and I verified it is unstable.

4.2.4 Stability of Shaker Sort

Shaker sort is working like gnome, it is not breaking stability, because of it is comaparing sequantial items. İt is stable.

4.2.5 Stability of Stooge Sort

Stooge sort firstly compares first element and last element of the array. We can quickly say that it compares spaced elements so it can break the stability. Also I checked it. It is unstable.

5. Space Complexity

5.1 Space Complexity Of Gnome

İn gnome sort, we are using limited variable. An these variables don't increase or decrease with input. So, we can say it's Space Complexity is O(1).

5.2 Space Complexity Of Comb

We have some variables and iterations. But we don't have any recursions or arrays. So it does not depend on input. Easily say that it's space complexity is O(1).

5.3 Space Complexity Of Shaker

like comb and gnome, we are not using any recursion or arrays, we are only using some variables to wander in array. Since these variable do not change with the inputs, it has O(1) complexity.

5.4 Space Complexity Of Bitonic

Bitonic a little bit different. Bitonic recalls 2 times merge function 1 times recall itself with different parameter. Merge function calls itself 2 times. That means we have frames for calling that means we use some memory with that. İt is Constantly diveded by 2 so we have logn, but we call merge function 2 times it must be log^2n, then it recall it self one times it means n. İf we multiply them we obtaine $O(nlog^2n)$.

5.4 Space Complexity Of Stooge

In stooge sort function recall it self with 2/3 of array 3 times. We have limited variables in the function which are i,j. We generate some frame for recursions 3 times. But at the end of the recursion we need some memory depends on input size. İt must be larger than n but it must be smaller than n^2 so it's complexity is O(n).

6-> References

https://en.wikipedia.org/wiki/Time_complexity

https://en.wikipedia.org/wiki/Comb_sort

https://en.wikipedia.org/wiki/Gnome_sort

https://en.wikipedia.org/wiki/Cocktail_shaker_sort

https://en.wikipedia.org/wiki/Stooge_sort

https://en.wikipedia.org/wiki/Bitonic_sorter