**You have to write your name and intro here**

Table of Contents:

1. Introduction…………………………………………… 2
2. Empirical Analysis of Algorithms……………………. 2
   1. Report Purpose…………………………………… 2
   2. Efficiency metric…………………………………. 2
   3. Input Size…………………………………………. 3
   4. Algorithms…………………………………………3
   5. Tools……………………………………………….4
   6. Run the Algorithms………………………………..5
   7. Analysis……………………………………………6
3. Conclusion………………………………………………8
4. **Introduction:**

“The empirical approach can be used when you do not have enough theoretical background on algorithm analysis. It can also be useful when you do not have access to the source code, or your source code is minified. In this case, we can run the algorithm with varying input size. We record the time taken by the program to run on each input. We can use a mathematical tool like regression to obtain the mathematical model that can be used to predict the running time of unknown input size.

In the theoretical approach, we use various theoretical and mathematical knowledge to find the running time. We do not need a computer for this. On the other hand, an empirical approach, we run the program on the actual computer. We run the program on various data sets; the relative performance gives the approximate running time of the program. Let me say it again that the theoretical approach is much superior and I strongly recommend doing it wherever applicable.”

1. **Empirical Analysis of Algorithms:**

**2.1 Report Purpose:**

“I am writing this report to compare the efficiency of two different techniques and at the end after doing all experiments i will tell which one is best. I have to compare two approaches to partition in quick select: Sedge wick (slides) and Lumoto (textbook). We apply different type of technique and different size of input on algorithms and analyze which is best which is not.”

* 1. **Efficiency metric:**

There are two ways to measure the efficiency

* Count how many basic operation will do in this algorithm.
* Count Physical time how much time an algorithm take.

2.3. **Input Size:**

* First of all we have to decide the size of array so initially I take size of array 100,2000,30000 and then keep on increment upto massive amount so we can find difference between two of them.
* Second thing we have to select input value in array which I selected from 0-100 as written in assignment.

2.4. **Algorithms:**

In this project we have used the following algorithms:

* **Quick select: Sedge wick**

**Write the reference of this algorithm here**

* **Quick select: Lumoto**

**Write the reference of this algorithm here**

* 1. **Tools:**

The tools which I used are given below:

* Notepad++

I used notepad++ to edit and write code of algorithms

* Excel Sheet

I used Excel Sheet to record the data of each algorithm output and used to draw the graphs.

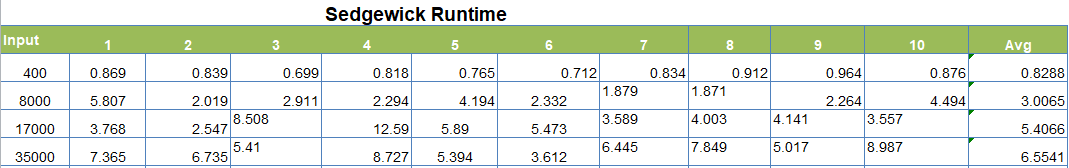
* Firefox Browser

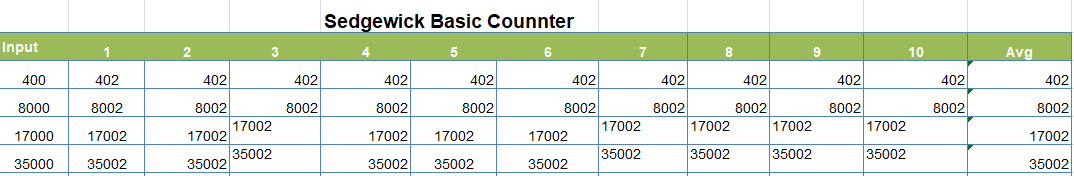
I used to run the code and show the output on the console

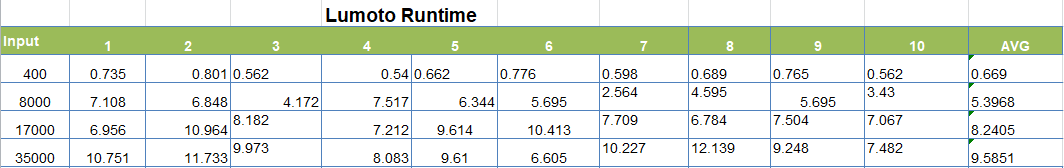
* 1. **Run the Algorithms**

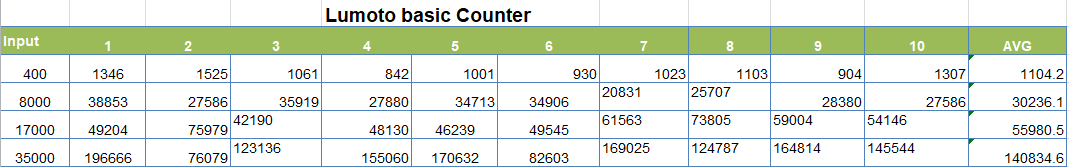
Now I am going to run these compiler I will each of them with different values at least ten times so my result will be more accurate and take average of them.

In the below table’s record of these data are given:

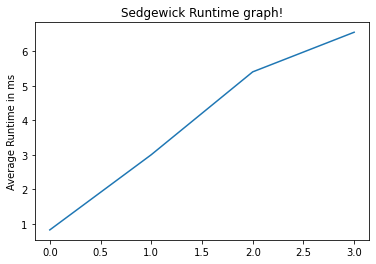
****

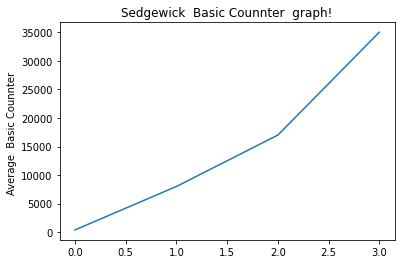


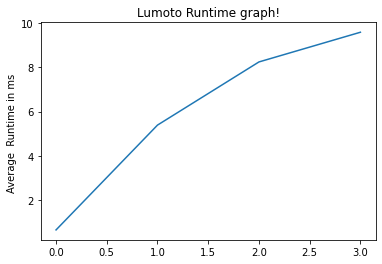
****

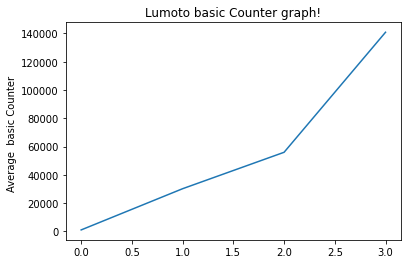
****

* 1. **Analysis:**









1. **Conclusion:**

The time complexity of Sedge wick is O (N). This algorithm is said to be almost 3 times faster than Lomuto partitioning. Lomuto partitioning is also a O(N) time complexity, O(1) space complexity algorithm and does the work in just one array traversal like Sedge wick partition but Lomuto partition requires more swaps and hence is relatively inefficient in this respect. However, Lomuto partition puts the pivot at the correct position in the array as well as returns the index whereas Sedge wick partition only returns the correct index of the pivot.

**Space Complexity**

The algorithm does not use any auxiliary space, hence space complexity is

O (1).