**Time Complexity with GFG**

Mostly calculated for

* Large input values and
* Worst case complexity

Ex: T(n) = 2n2+ 2n + 1

The highest power is 2n square so we will remove small one

T(n) = 2n2

Remove all the constant

T(n) = n2 = O(n2)

Path: O(1) < O(log n) < O(n)< O(n log n) < O(nc) < O(n!)

1. **For Loop:**

For(i=1;i<=n;i++){

x = y+z; //constant time

}

Sentence: the time complexity will take n times to execute constant (x = y+z) = C

= Cn //drop the constant

=O(n)

1. **Nested For Loop:**

For(i=1;i<=n;i++){// n times loop

For(->n){ x = y+z; //constant time C} // n times loop

}

T(n) = C\*n\*n

= Cn2  Drop the Constant

= O(n2)

1. **Sequential Statement:**
2. a = a+b; // Constant C1
3. For(i=1;i<=n;i++){x = y+z; //constant time Constant C2} n
4. For(i=1;i<=n;i++){c = a+b; //constant time Constant C3}n

= C1  + C2 n + C3 n //remove all constants

= 2n //again remove the constants

= n

TC = O(n)

1. **Conditional Statement:**

Suppose we have if and else statement

If {

O(n)

}

Else{

O(n2)

}

Here always note that we focus on worst case so here else would be considered because it has worst case time complexity.

**Q. Imagine a classroom of 100 students in which you gave your pen to one person. You have to find that pen without knowing to whom you gave it.**

Here are some ways to find the pen and what the O order is.

* **O(n2):** You go and ask the first person of the class, if he has the pen. Also, you ask this person about other 99 people in the classroom if they have that pen and so on,   
  This is what we call O(n2).
* **O(n):** Going and asking each student individually is O(N).
* **O(log n):** Now I divide the class into two groups, then ask: “Is it on the left side, or the right side of the classroom?” Then I take that group and divide it into two and ask again, and so on. Repeat the process till you are left with one student who has your pen. This is what you mean by O(log n).

I might need to do:

* The **O(n2)** search if **only one student knows on which student the pen is hidden**.
* The **O(n)** if **one student had the pen and only they knew it**.
* The **O(log n)** search if **all the students knew**, but would only tell me if I guessed the right side.

O(n log n) = Algorithms that repeatedly divide a set of data in half, and then process those halves independently with a sub algorithm that has a time complexity of O(N), will have an overall time complexity of O(N log N). Examples of O(N log N) algorithms: Merge sort, Heap sort, and Quick sort.

<https://www.geeksforgeeks.org/time-complexity-and-space-complexity/>

https://www.geeksforgeeks.org/time-complexities-of-different-data-structures/