Substitution Method

To Find the time complexity of recurrence relation there are three methods ; the first one is : Substitution Method .

The Name Suggest It try to Substitute the values in recursive term

Ex : T(n) = T(n-1) + n

Here T(n-1) is a recursive term ;

n is a Non Recursive term ;

For Any T(n) there is Some Recursive & Non Recursive Term

Now we try To Change n-1 where n is present

Now our Recursive term look like :

T(n-1) = T(n-2) +n-1 (Now we change n 🡪 n-1) So,

T(n-1) + n 🡪 T(n-2) + n-1.

Here we substitute n-1 where n is present .

Now we Take Factorial Example after substituting:

1st Time 🡪 T(n-2) +n-1

2nd Time 🡪 T(n-2)+n-1+n

3rd Time 🡪 T(n-3)+n-2+n-1+n

If we Do it For 50

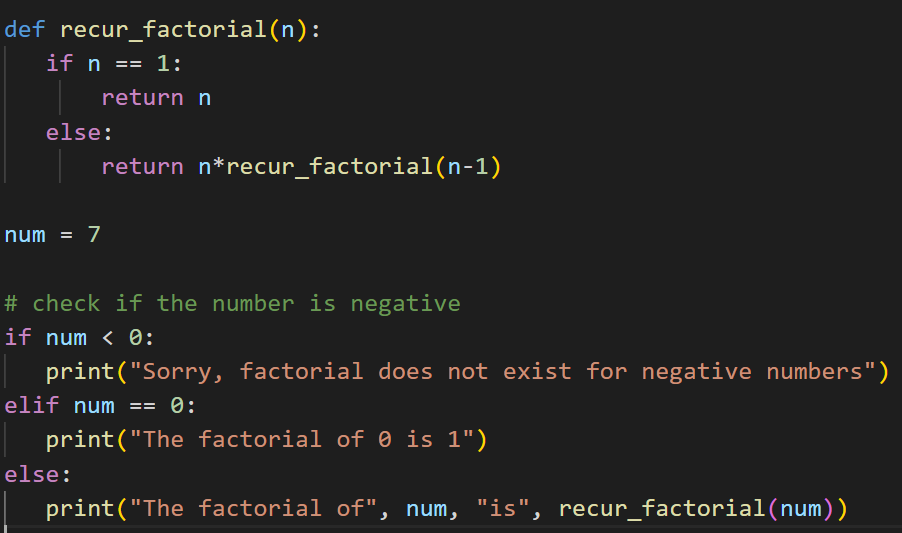
T(n-50) + n-49 + n-48 + n-47 …………………+n-1 + n

How much time it need To Taken

Until n becomes 1 (n=1).

T(n)=1 🡪this is the Base Condition

Code Example (factorial) :



Now we see the n > 1 it is goes else part and comes again and again in else part by -1 for ex : n = 40 first it 39 second 38 third 37 likewise it will come until n = 2 then If n = 1 it goes to base condition and finish the program

So To Get the T(n) = 1 ;

T(n-50) + n-49 + n-48 + n-47 …………………+n-1 + n

If comes to n it meets the base Condition

we substitute n has n-1 if we get this , the next one is T(n) = 1

so if our value is k

T(n-k) + n-K +1 + n-k+2 + n-k+3 + n-k+4 + n-1 + n

If it come n it meets the base condition and exit the loop

n-k = n-(n-1)

= n-n +1

= 1

n-k+1 = n-(n-1) + 1

= n-n +1+1

=2

So now our value is : 1 + 2 + 3 + 4 ……………+ n-1 + n

So it is n natural number ; To substitute this we use the formulas

N(n+1)/2 🡺 n2+n\2 🡪n2/2 🡪 n\2 🡺 O(n2)

So the Time Complexity is O(n2)

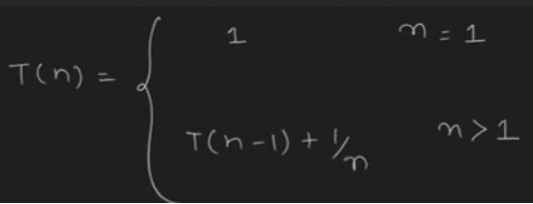
Example 2 :

T(n) = n(1) if value is 1

Otherwise n > 1 (n greater than 1)

T(n) = T(n-1) / 1/n

**Base Condition : T(n) 🡺 n(1)**



What is the time Complexity ?

In this the recursive term is T(n-1) & Non Recursive term is 1/n

If we use First it is in :

1. T(n-1) + 1/n (then)
2. T(n-2) + 1/n-1 + 1\n (then)
3. T(n-3) + 1/n-2 + 1/n-1 + 1\n

IF we do it for 50 time our equation is :

T(n-50) + 1/n-49 + 1/n-48 + 1/n-47 ……… + 1/n-1 + 1/n

So how many time we Done this ?

It will Go to the Base Condition T(1)

So the time is k ,

So our Output is Something Like :

= 1 + ½ + 1/3 + ¼ + 1/5 +………+1/n+1 + 1/n

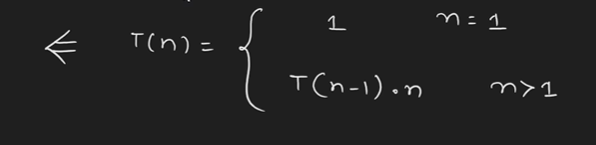
See our Output is like log series

Our Time Complexity = O(log n)

Example 3 :

T(n) = n(1) 🡪 if value is 1

T(n) = T(n-1) \* n 🡪 if value n > 1



Now we use substitute method to solve the problem and find the time complexity .

Same if the

first time T(n) = T(n-1) \* n

Second time T(n) = T(n-2) \* n-1 \* n

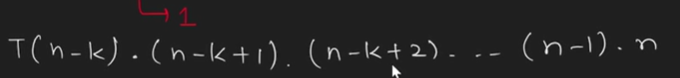
Third Time T(n) = T(n-3)\*n-2 \*n-1\*n

If it is 50th time

T(n) = T(n-50)\*n-49\*n-48\*n-47………..\*n-1\*n

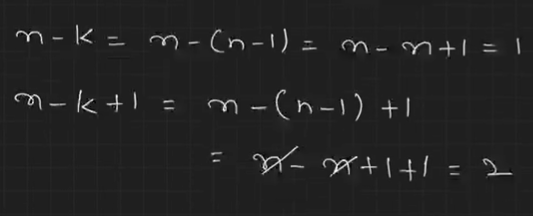
It will run until it meet the base condition t(1)

Now how our Model works :



It is increase 1 by 1

If we Substitute :

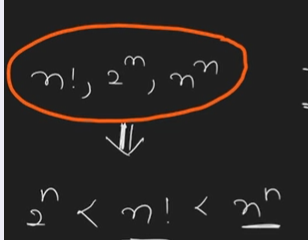


So our value we get is : T(1) \* 2 \*3 \*4 \*5…….(n-1)\*n

1, 2 , 3 , 4 ………(n-1) \*n

So it is similar like n Factorial O(n!)

As we dicuss in Big O :



The Time complexity is

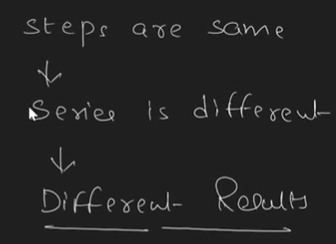
Big 0 🡺 O(nn) (n square) worst case

Omega Ω 🡺 Ω(2n) best case

The Biggest difference between this example and previous example is previous are addition between recursive & Non recursive term

But now it is multiplication

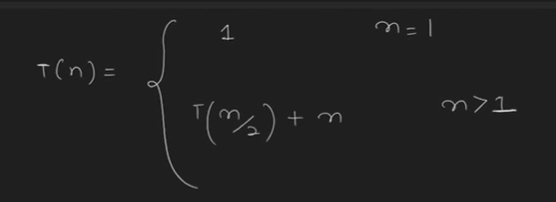
The Steps are Same the Result is Different



Example 4 :

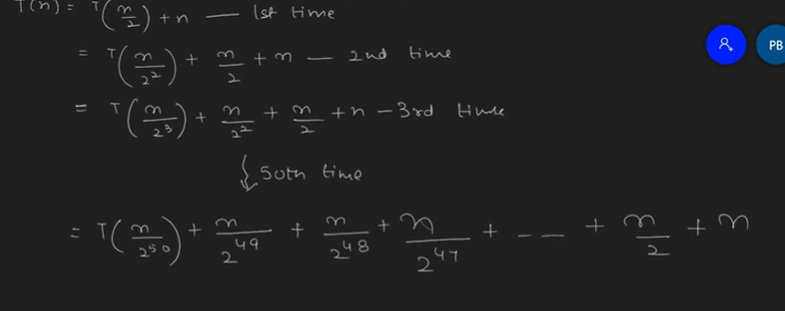
Now we see the AP & GP Series

The Problem is :

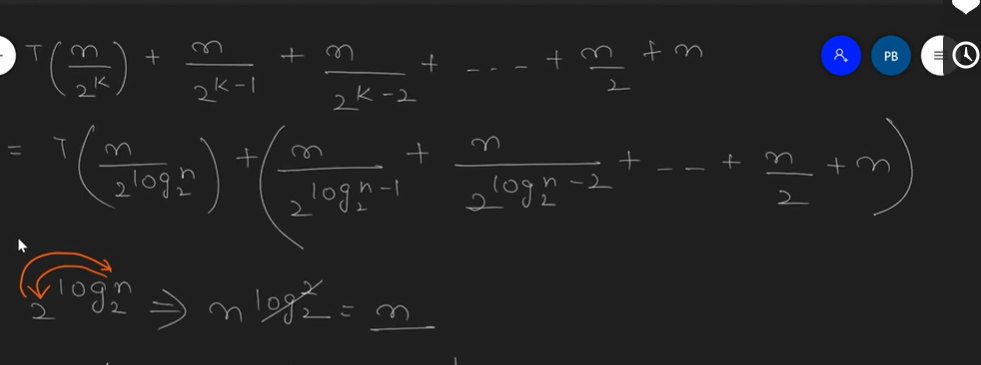


Here it is Division Term

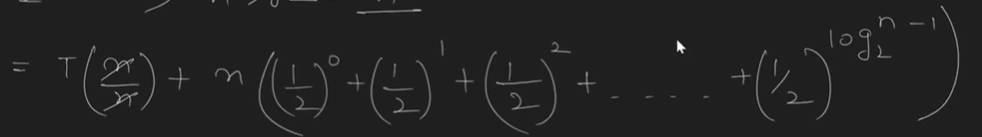
The Steps are :



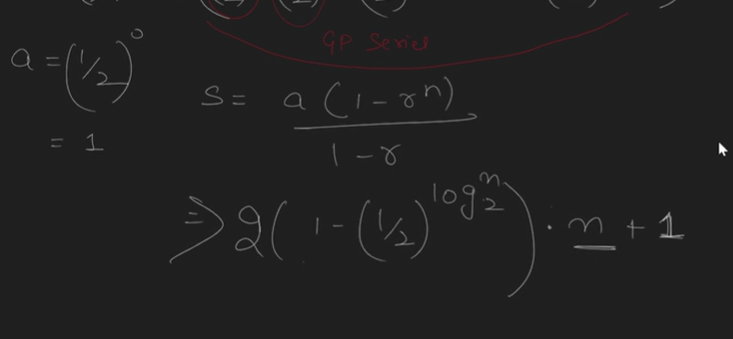
The base Condition T(n) = 1

The Steps are 

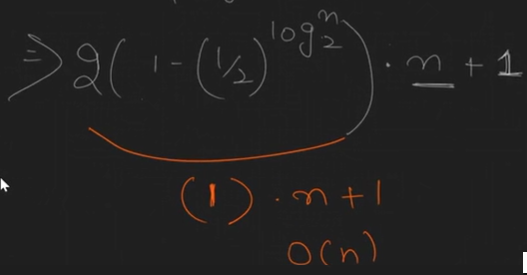
The Values are :



This is what GP Series



The Time Series is :



O(n)