Analysis of Quick Sort

l								h	
0	1	2	3	4	5	6	7	8	
10	16	8	12	15	6	3	9	5	∞

pivot = 10.

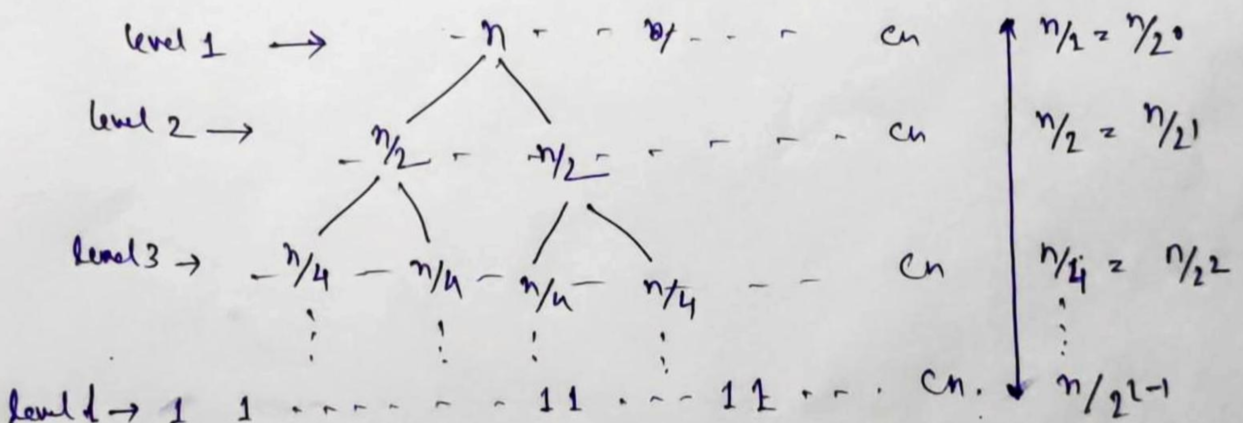
```
QuickSort (l,h)
{
  if (l<h)
  {
     j = partition (l,h);
     Quicksort (l, j) ;
     Quicksort (j+1,h);
  }
}

partition (l,h) .
{
  Pivot = A[l]
  i = l , j = h;
  while (i<j) {
   do
    { i++ ;
    } while ( A [i] <= pivot);
   do{
      j --;
    } while ( A [j] > pivot);
   if (i<j)
    {
     swap (A [i] , A [j])
    }.
  }
  swap (A[l] , A[j]);
  return j; // dividing position>.
}.
```

Best case running Time:

QuickSorts best case occurs when the partitions are as evenly balanced as possible: their size either equal or are within 1 of each other. The former case occurs when there are (in the subarray) an odd number of element & the pivot is right in the middle after partitioning, & each partition has $(n-1)/2$ elements. The later case appears if the subarray has an even number elements & one partition has $n/2$ elements & the other has $n/2 - 1$ elemts.

In either case each of them has at most $n/2$ elements.



$$n/2^{l-1} = 1$$

$$2^{l-1} = n \qquad \text{or, } l-1 = \log n$$

$$\text{or } l \simeq \log n.$$

$\therefore$ total complexity $= cn + cn + \dots$ till lth level

$= l\,cn$

$= cnl = cn\log n.$

Using big-theeta notation we get the result $\Theta(n\log n)$.

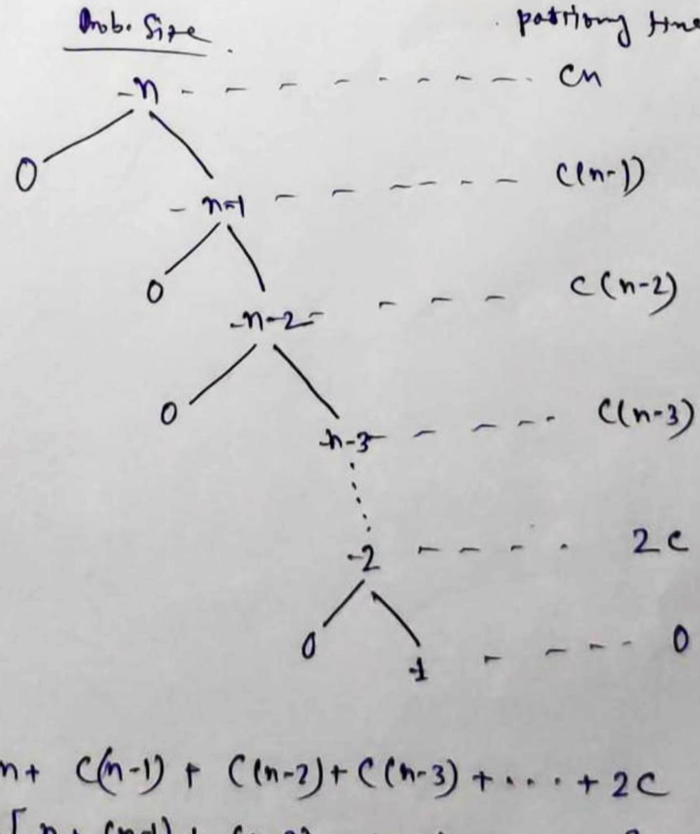
Recurrance Relation:

$$T(n) = \begin{cases} b \\ 2T(n/2) + cn. \end{cases}$$

⇒ Same as merge sort. use subtitution and/or ~~recursion~~ Masters theorem to solve the problem.

Wrost Case running Time for quick sort

when quick sort always has most unbalanced partitions possible then the original call takes cn time for some constant c. the recursive call on (n-1) element takes c(n-1) time, the recursive call on (n-2) elements takes c(n-2) time & so on.



$$\therefore \text{total time} = cn + c(n-1) + c(n-2) + c(n-3) + \dots + 2c$$

$$= c[n + (n-1) + (n-2) + (n-3) + \dots + 2]$$

$$= c\left(\frac{(n+1)n}{2} - 1\right). \quad [\text{we substracted 1 as the Series starts from 2}].$$

$\therefore$ complexity $= O(n^2)$.

Recurrence Relation.

$$T(n) = \begin{cases} T(n-1) + n \\ 0 \end{cases}$$

<u>Substitution</u>.

$$T(n) = n + T(n-1)$$

$$= n + (n-1) + T(n-2)$$

$$= n + (n-1) + (n-2) + T(n-3)$$

$$\vdots$$

$$= n + (n-1) + (n-2) + \cdots + 3 + 2.$$

$$= \frac{(n+1)\,n}{2} - 1$$

$$\simeq \Theta(n^2).$$

Think: Can we solve it with masters theorem?