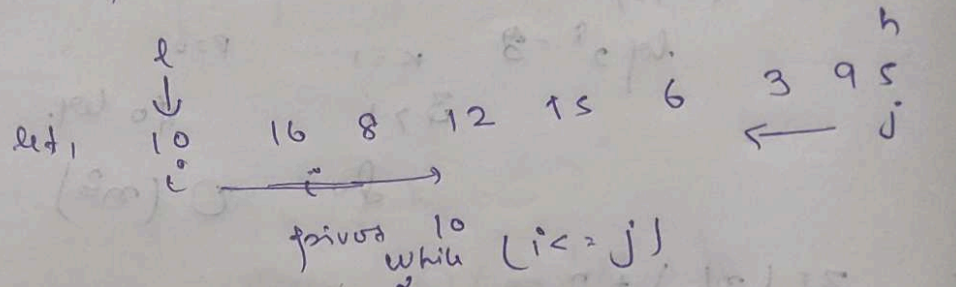
# Quick sort

Approach :-

Step 1 :- Select one element

Step 2 :- On that element all right side has been smaller than that

Step 3 :- left side be greater than that element




pivot 10

while (i <= j)

Increment i until you find element is > 10

16 > 10

so, 10 16 . . . .

i

Decrement j until you find element smaller than 10

. . . 5

j

then exchange then

10 5 8 12 15 6 3 9 16

i → ← j

Increment i — next greater elem is 12

decrement j — next smaller elem is 9

exchange.

10 5 8 9 15 6 3 12 16
i j

continue...

Now 15 > 10
i

& 3 < 10
j

Swap

10 5 8 9 3 6 15 12 16
i → ← j

10 5 8 9 3 6 15 12 16
j i

Stop

doesn't swap i & j bcz both i > j

we found pos. of pivot i.e. j

fixed

6 5 8 9 3 10 15 12 16

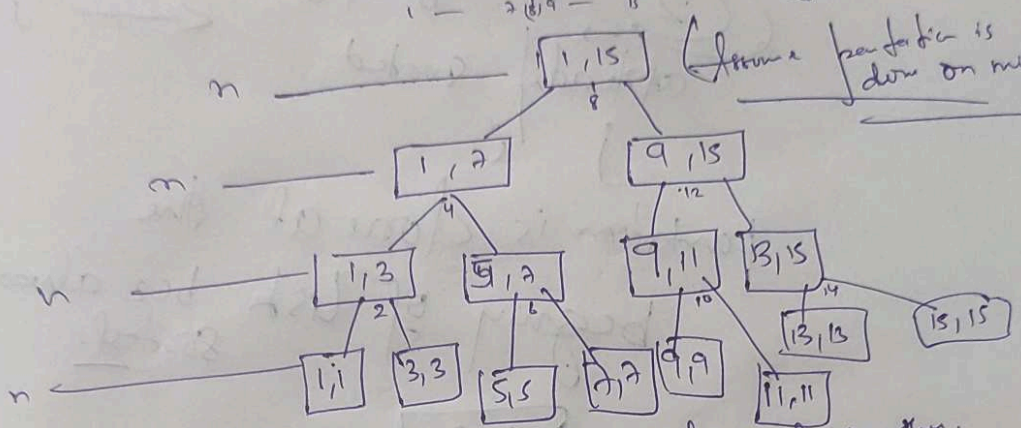
Now, perform Quick sort recursively.

```
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;
}
QuickSort (l, h)
{ if (l<h)  // at least two element
    j= partition (l, h);
    QuickSort (l, j);
    QuickSort (j+1, h);
}
```

## Analysis of Quick sort

O(nlogn) (Best Case)




Here compare the element and exchange them.

So, ——— n time

n ——— j = partition (l, h)

n × n × . . (height of tree)

they all are divide by 2 again by 2, . .

$n/2$ $n/2^2$ ....

$n/2^k = 1$

$k = \log_2 n$

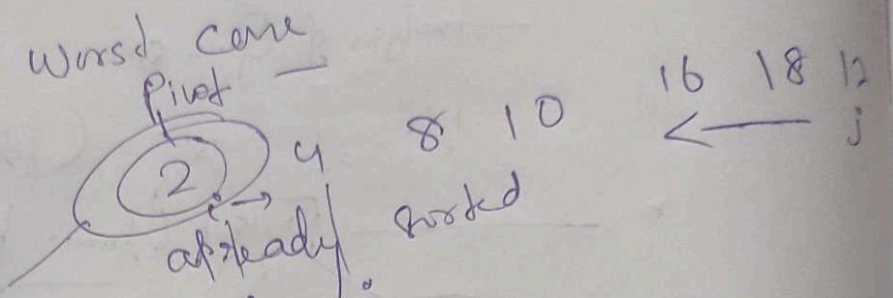
$O(n \times \log_2 n)$ ← 1

Partition is done always on middle

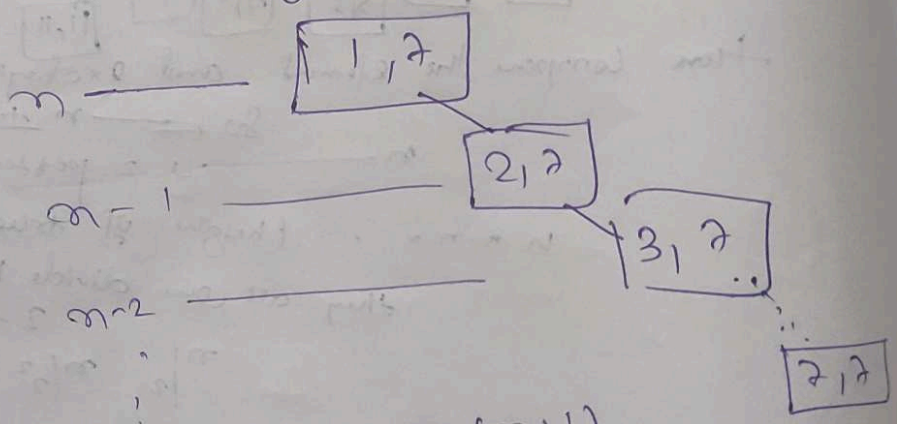
Only when we known median should be posted — 1 2 3 (4) 5 6 7

by It is not possible always

Worst case




partation is done at the beginig of list bcz already sorted




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

$= O(n^2)$ (worst case)

So, It is problem.

So, for solving this

don't select 1st element as pivot

① Select the middle element as pivot

② Random element is pivot