Date: May 20, 24 Day.

Submitted By : Junaid Asif
Roll No : BSAI-144
Class : BSAI - (IV-Sem)
Subject : Analysis & Algorithm

Submitted To : Dr. Hina Ali

Assignment # 02

Quick Sort

Array = [13, 22, 23, 48, 70]

Sol:

	p(low)				q(high)
	13	22	23	48	70
index:	0	1	2	3	4

Pivot element P = 13

∵ P = pivot element
∵ q = high index
∵ r = low index

r = 13 at 0 index
q = Partition (A, 0, 4)

P = 13 at 0 index
P = A[r] → P = A[0]
i = q+1
i = 4+1 => 5
j = 4 to 1
j = 4 , i = 5
if 13 >= 22
i = 4

Date: Day:

Similarly,

j = 3 , i = 3
j = 2 , i = 2
j = 1 , i = 1

Swap A[i-1] with A[r]
A[0] with A[0] → no change

Now Quick Sort (A, 0, -1)
QS (A, 1, 4)
P = 22 → r is at index 1
i = 5
j = 4 to 2
j = 4 , i = 5
if 23 >= 22
i = 4

Similarly,

j = 3 , i = 3
j = 2 , i = 2
Swap A[2-1] with A[r]
A[1] with A[1] → no change

Now,

Quick Sort (A, 1, 0)
QS (A, 2, 4)
i = 5 , P = 23
j = 4 , i = 5
if 48 >= 23
i = 4

Similarly,

j = 3 , i = 3
Swap A[3-1] with A[r]
A[2] with A[2] → no change

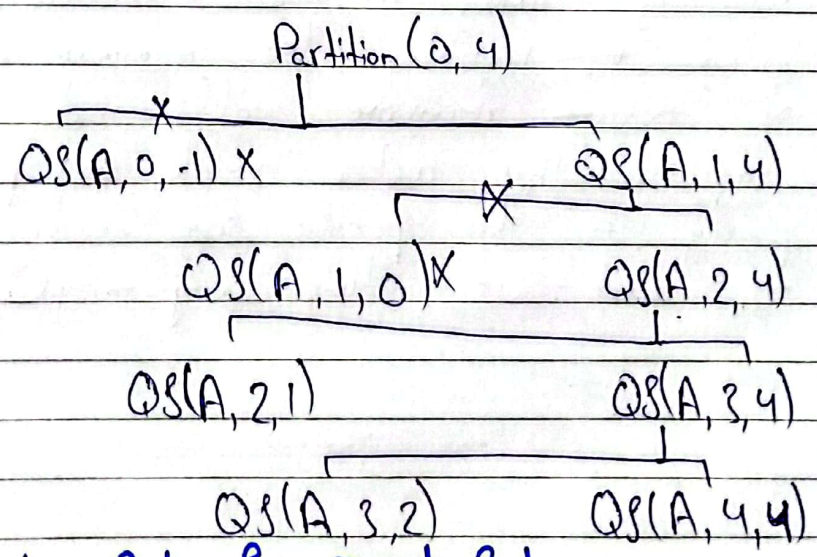
Date: Day:

Quick Sort (A, 2, 1)
QS (A, 3, 4)
P = 48
i = 5
j = 4 to 4
j = 4
if 70 >= 48
i = 4
swap A[i-1] with A[p]
A[3] with A[3] → no change

Now
QS (A, 3, 2)
QS (A 4, 4)

So, final sorted Array = 13 22 23 48 70

Tree:



Pseudo Code for Quick Sort (Left to Right)

```
QS(A, r, q)
if (r<q)
Partition (A, r, q)
QS(A, r, P-1)
QS (A, P+1, q)
```

Date: Day:

```
Partition (A, r, q)
Pivot (p) = A[r]
  i = q+1
  for j = q downto r+1
     if A[j] >= Pivot
        i = i-1
        Swap A[i] with A[j]
     Swap A[i-1] with A[r]
     return i-1
```

Sorted Array Worst Case:

The worst case scenario occurs because each partitioning step only removes one element and the remaining array is still essentially sorted requiring the maximum number of recursive calls and comparisons. This makes quick sort perform poorly in this specific case leading to its worst-case time complexity $O(n^2)$.