

* Kth Largest Element in an Array

$O(n \log n)$ $O(1)$ → - sort and take k^{th} → noob

$O(n \log k)$ $O(k)$ → • Priority Queue & take k^{th} - only items are given in ~~order~~ ^{order} when ~~popping~~ ^{polling, peeking}

• Quick Select → relates to quick sort.

3 2 1 5 6 4 ^{pivot}
_l _r

i, j

if j less than pivot, $i++$ & then swap i, j
 else $j++$

worst case

$O(n^2)$

3 2 1 5 6 4 \Rightarrow 3 2 1 5 6 4 \Rightarrow 3 2 1 5 6 4
_i _j _i _j _i _j

3 2 1 5 6 4 \Rightarrow 3 2 1 5 6 4
_i _j _i _j _i _j

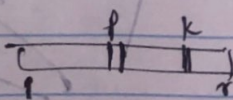
now swap pivot & $(i+1)$

3 2 1 4 5 6

← less than 4

→ greater than 4

check pivot place from right & if $== k$, return value.

if $place < k$ 
 $l = p+1$ place

else $place > k$
 $r = p-1$ place

We can optimize this using a random pivot position from (l, r) range.

select random value & swap with r .
now also pivot is at the end.

~~More~~

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

$$T(n/2) = T(n/4) + (n/2 - 1)$$

$$T(1) = 0$$

$$T(n) = (n-1) + (n/2 - 1) + (n/4 - 1) + \dots$$

tree
depth

$$\sum_{i=0}^{\log n - 1} (n/2^i - 1)$$

$$n \left(\sum_{i=0}^{\log n - 1} \frac{1}{2^i} \right) - \sum_{i=0}^{\log n - 1} 1$$

$$n \left(\sum_{i=0}^{\log n - 1} \left(\frac{1}{2} \right)^i \right) - \log(n)$$

$$n \left(\frac{1 - \left(\frac{1}{2} \right)^{\log n}}{1 - \frac{1}{2}} \right) - \log n$$

Atlas

No: _____

Date: ____/____/____

$$2^n \left[1 - \left(\frac{1}{2} \right)^{\log n} \right] - \log n$$

$$2^n \left[1 - \frac{1}{2^{\log n}} \right] - \log n$$

$$b^{\log_b K} = K$$

$$\lg n = n$$

$$\lg 2 = \lg n$$

$$\lg n \lg 2 = \lg n$$

$$2^n \left[1 - \frac{1}{n} \right] - \log n$$

$$\frac{2^n}{n} (n-1) - \log n$$

$$a^b = c$$

$$\log_a c = b$$

$$2^{(n-1)} - \log n$$

leading complexity

$$\therefore O(n)$$