9.2 Selection in expected linear time.

Randomized-Select (A, p, r, i)

if $p = r$ return $A[p]$

else q = Randomized-Partition(A, p, r)

$k = q - p + 1$

if $i = k$ return $A[q]$

else if $i < k$ return Randomized-Select$(A, p, q-1, i)$

else if $i < k$ return Randomized-Select$(A, p, q-1, i)$
else return Randomized-Select$(A, q+1, r, i-k)$

1 2 3 4 5 6 7 8

A 17, 21, 3, 19, 51, 11, 26, 42

42, 21, 3, 19, 51, 11, 26, 17

i j

11, 21, 3, 19, 51, 42, 26, 17

i j ~~j~~

11, 3, 21, 19, 51, 42, 26, 17

j < i

11, 3, 17, 19, 51, 42, 26, 21

17 is now correctly positioned at index 3.

return 3.

e.g. A =

17	21	3	19	51	11	26	42

$i = 5$

call $R\text{-}S(A, 1, 8, 5)$ $\quad p = 1 \quad r = 8 \quad i = 5$

q = Randomized-Partition $(A, 1, 8)$

$q = 3$

$k = 3 - 1 + 1 = 3$

if $i = 3$ (No) - $i = 5$

elseif $i < 3$ (No). $i = 5$

else $R\text{-}S(A, 4, 8, 2)$

$A =$

1	2	3	4	5	6	7	8
11	3	17	19	51	42	26	21

pivot

$i = 2$ now.

$q = \text{RandomizePartition}(A, 4, 8)$

	1	2	3	4	5	6	7	8
$A =$	11	3	17	19	21	26	51	42

pivot

$q = 6$

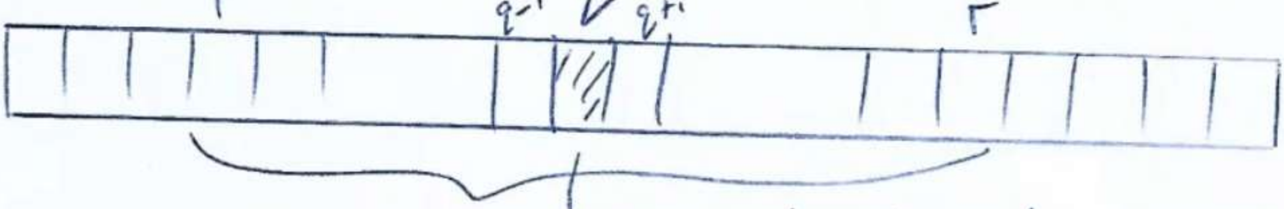
$k = 6 - 4 + 1 = 3$

if $i = 3$ (No, $i = 2$)

if $i < 3$ (Yes, $i = 2$)

looking for i^{th} statistic ..

q = Randomized-Partition (A, p, r, i)



correctly placed

k = position of q in $A[p, r]$.

k = position of q in $A[p, r]$.

if $i = k$ then return $A[q]$.

if $i < k$ recurse to left $A[p, q-1]$

R-S$(A, p, q-1, i)$

if $i > k$ recurse to right $A[q+1, r]$

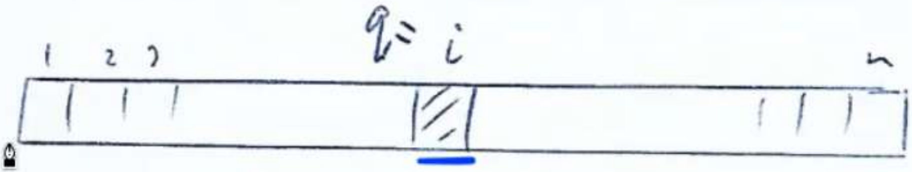
$$R\text{-}S(A, q+1, r, i-k)$$

the pivot

Note: If q is roughly in the middle of p and r then the recursive call is to an subarray of about half the size

Running-Time of Randomized-Select :

<u>Best</u> case A



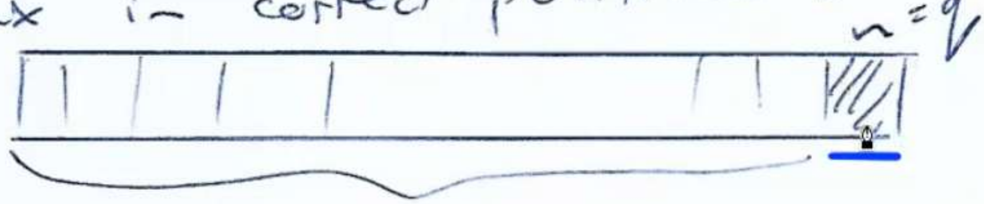
i

Say Randomized-Partition randomly places the correct value in position i

− $\Theta(n)$ for the partition −

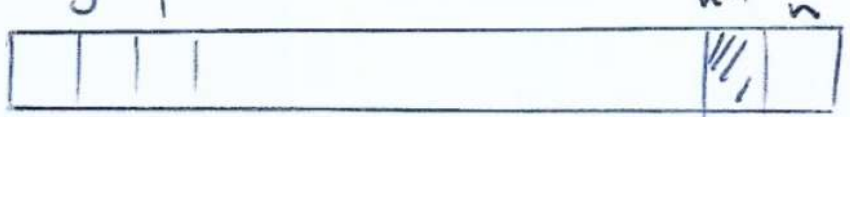
Worst case

Randomized-Partition randomly places the max in correct position -



Recursive call to array of size n-1

Say Randomized-Partition again randomly places max in correct position



Say Randomized-Partition again randomly places max in correct position

Recursive call to array of size n-2

Running-time is

$$\Theta(n) + \Theta(n-1) + \Theta(n-2) + \dots + \Theta(2) + \Theta(1$$

$$= \sum_{i=1}^{n} \Theta(i)$$

$$= \Theta\left(\sum_{i=1}^{n} i\right)$$

$$= \Theta\left(\frac{n(n+1)}{2}\right) = \underline{\underline{\Theta(n^2)}}.$$

We will show that the expected or average run-time is $\Theta(n)$.