OH TODAY

- 1:30pm - 2:20pm
- ~~Zoom~~

Exam 1

- Feb 27
- during class time
- seating will be posted on class page.

Selection

Input: Array A containing n distinct elements.

Obj: To find the i^{th} smallest element in A.

$\boxed{13}$ 24 4 8 39

↓

3^{rd} smallest element.

Naive alg: 1. Sort A

2. return the element at the

i^{th} index of the array.

Running time: $\Theta(n \lg n)$

Target runtime: $\Theta(n^{\cancel{2}} \lg n)$, $\Theta(n)$.

$\boxed{\underline{\text{Alg.}}}$ $\rightarrow$ Select (A, i)

1. Divide A into $\lfloor n/5 \rfloor$ groups, each containing exactly 5 elements & at most one group containing $n \bmod 5$ elements. $O(n)$

2. Find the median of each group. Let M be the set of medians. $O(n)$.

3. Find the median of the medians. Call it x.

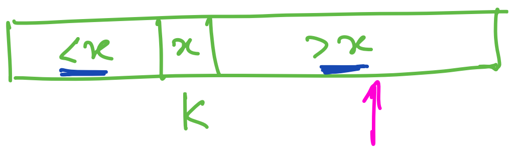
$\text{Select}\left(M, \frac{1}{2}\left\lceil \frac{n}{5} \right\rceil\right)$

$T\left(\left\lceil \frac{n}{5} \right\rceil\right)$

4. Partition the array A around x. from QS

Let $k = \text{rank}(x)$

pos^n of x in the sorted array. $O(n)$



5. if $k = i$ then return x $O(1)$

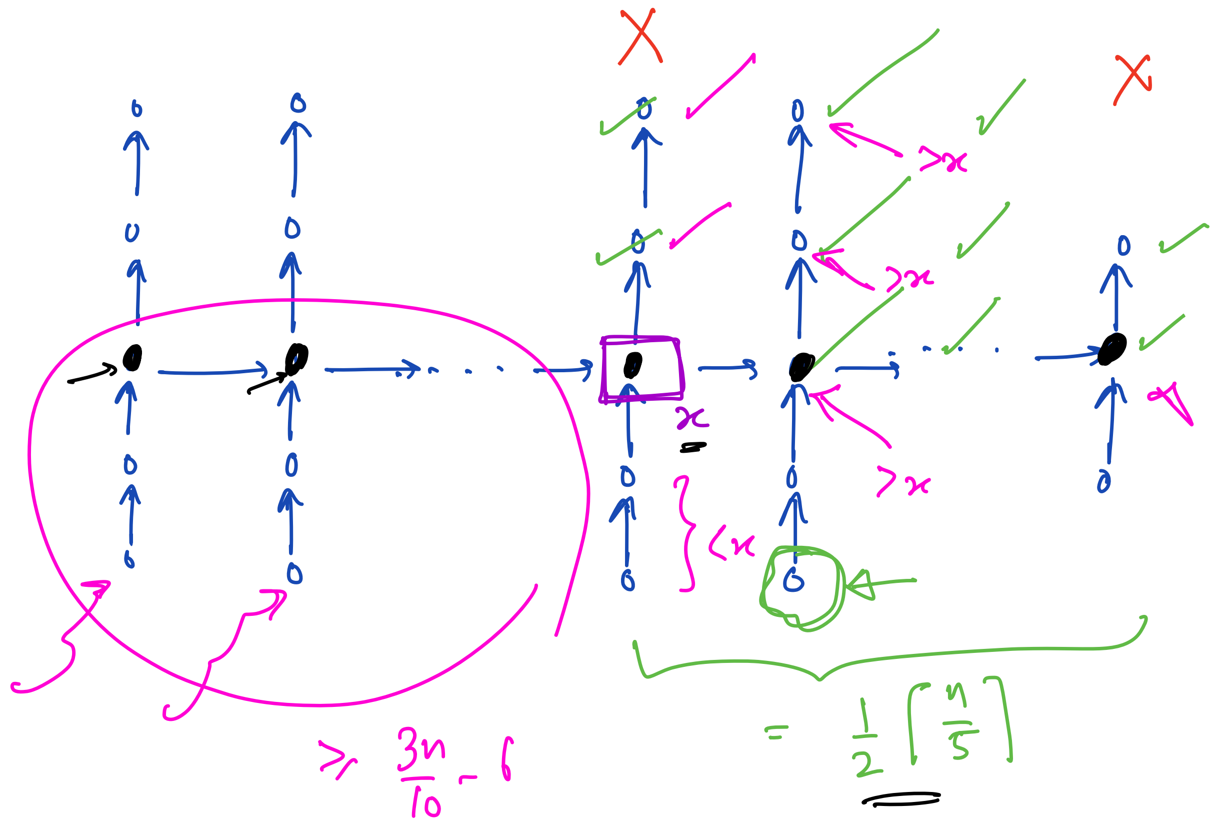
6. else if $k > i$ then

return $\text{Select}(A[1..k-1], i)$.

7. else return $\text{Select}(A[k+1..n], i-k)$.

Analysis :

Q: How many elements are we guaranteed to eliminate when we recurse?



$$\#\text{ elements} > x \geq 3\left(\underbrace{\frac{1}{2}\left\lceil\frac{n}{5}\right\rceil - 2}_{\#\text{ cols}}\right)$$

$$\geq \frac{3n}{10} - 6$$

$$\#\text{ elements} < x \leq \frac{7n}{10} + 6$$

Runtime recurrence

$$T(n) \leq \begin{cases} O(1), & n \leq 140 \\ T\left(\left\lceil\frac{n}{5}\right\rceil\right) + T\left(\frac{7n}{10}+6\right) + O(n), & \text{o.w} \end{cases}$$

an, for some const a.

$$T(n) = \Theta(n)$$

We will prove using induction that

$\rightarrow$

$\to T(n) \leq cn, \quad \forall n \geq n_0,$ where c & n_0 are positive consts.

<u>BC</u> : $n \leq 140$

$T(n) \leq c \cdot n$, for sufficiently large c.

<u>IH</u>: Let $k > n_0$ be an arb but particular integer. Assume that $T(j) \leq cj$, $\forall j$ s.t. $n_0 \leq j \leq k-1$.

<u>IS</u> : We want to show that

$$T(k) \leq ck, \quad \forall k \geq n_0.$$

$$T(k) \leq T\left(\left\lceil \frac{k}{5} \right\rceil\right) + T\left(\frac{7k}{10} + 6\right) + ak$$

$$\leq c\left\lceil \frac{k}{5} \right\rceil + c\left(\frac{7k}{10} + 6\right) + ak, \quad (\text{By IH})$$

$$\leq c\left(\frac{k}{5} + 1\right) + c\left(\frac{7k}{10} + 6\right) + ak$$

$$= \frac{9ck}{10} + 7c + ak$$

$$= ck + \left(- \frac{ck}{10} + 7c + ak\right)$$

The above expression will be $\leq ck$,

when

$$7c + ak - \frac{ck}{10} < 0$$

i.e.,
$$\frac{ck}{10} > 7c + ak$$

$$c\left(\frac{k}{10} - 7\right) > ak$$

$$c > \frac{10ak}{k-70}$$

Set $n_0 = 140$, $c > 20a$

Stacks.

ADT (Abstract Data Type)

- element could be arb objs
- operations — Push
  — Pop

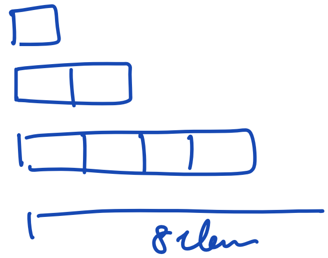
Array implementation :

Push (obj)

// s : stack size - # elems in the stack

// a : array size

// c : initial size of the array & increment.



$A[s] \leftarrow obj$

$s{+}{+}$

if $s == a$ then

$a \leftarrow a + c$

copy all elems into the new array.

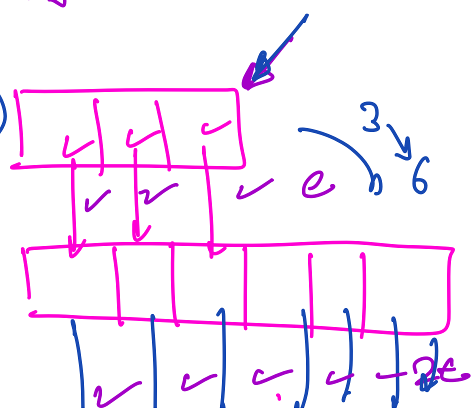
$a \leftarrow a * 2$

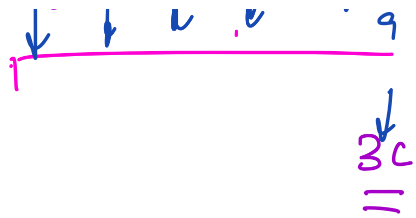
$T(n)$: total time to push n elems.

$$T(n) = n + \underbrace{c + 2c + 3c + \cdots + n}_{\text{copy}}$$

$$= n + c\left(1 + 2 + 3 + \cdots + \frac{n}{c}\right)$$

$$= n + c \cdot \frac{\left(\frac{n}{c}\right)\left(\frac{n}{c} + 1\right)}{2}$$





$= \Theta(n^2)$

Alt implementation $(a \leftarrow a * 2)$

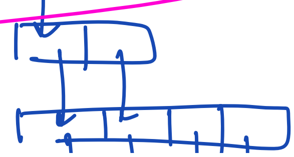
$T(n)$: worst case running time of n pushes.
(we start with an array of size 1)

$\le \lg n$ resizing of the array

$\le n$ elements to copy during each resizing

$\therefore \Theta(n \lg n)$. $O(n \lg n)$

$T(n) = n + 1 + 2 + 2^2 + \cdots + 2^{\lg n}$



$$= n + 2^{\lg n + 1} - 1$$

$$= n + 2 \cdot 2^{\lg n} - 1$$

$$= n + 2n - 1$$

$$= \Theta(n)$$

$n \lg n$

$\downarrow$

n

n^2

$\downarrow$

$n \lg n$

$$T(n) = 2T\left(\frac{n}{2}\right) + n \longrightarrow \Theta(n \lg n)$$