12.08.2024

ESO207 - Data Structure & Algo

(1) $\theta(n^{1.6}) \overset{+}{\approx} \dfrac{n \cdot n^{\log_2 3/2}}{n^{\log_2 2} \cdot n^{\log_2 3/2}} \approx n^{\log_2 3}$

(2) Merge (A, n, i, B, m, j, C, k)

- -

- $C[k] \leftarrow \min\{A[i], B[j]\}$

$k++$

Merge (A, n, i, B, m, j, C, k)

$T(n) = 2 \cdot T(n/2) + f(n)$

$\downarrow$

$= \Theta(n \log n)$

What if $n \neq 2^k$

- $\lfloor n/2 \rfloor$, $\lceil n/2 \rceil$

- 0 0 0 1 1 1 1 1 (n)

Fact $\forall x \quad \exists$ perfect power of 2 $\in [x, 2x]$

$5 \qquad\qquad 8 \in [5, 10]$

Median Finding

Def^n [i-th Statistic] i-th smallest element in the array.

Def^n [Median] $\lceil n/2 \rceil$-th statistic

$\lceil \frac{n}{2} \rceil$

[Lower median]

[higher median]

Q: Given array A, return the i-th statistic.

A: Sort low to high & return i-th one.

$O(n \log n)$ time

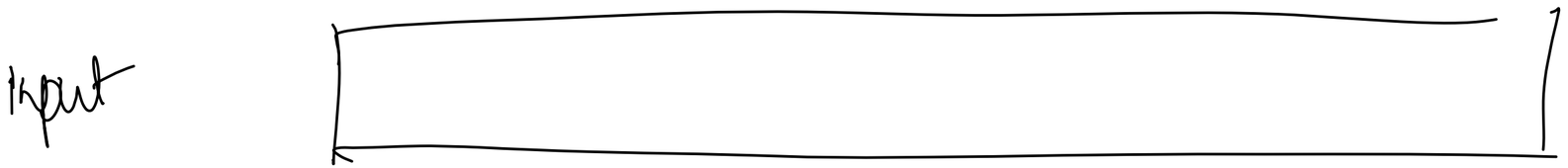
Q: Can we do better?

Ans $O(n)$ time!

Suppose $i = 1, 2, \ldots,$ constant.

$$i = \lfloor n/2 \rfloor$$

i-th order statistic



$$T(n) = a\,T(n/b) + cn$$

$$3 \quad 2 \longrightarrow n^{\log_2 3}$$

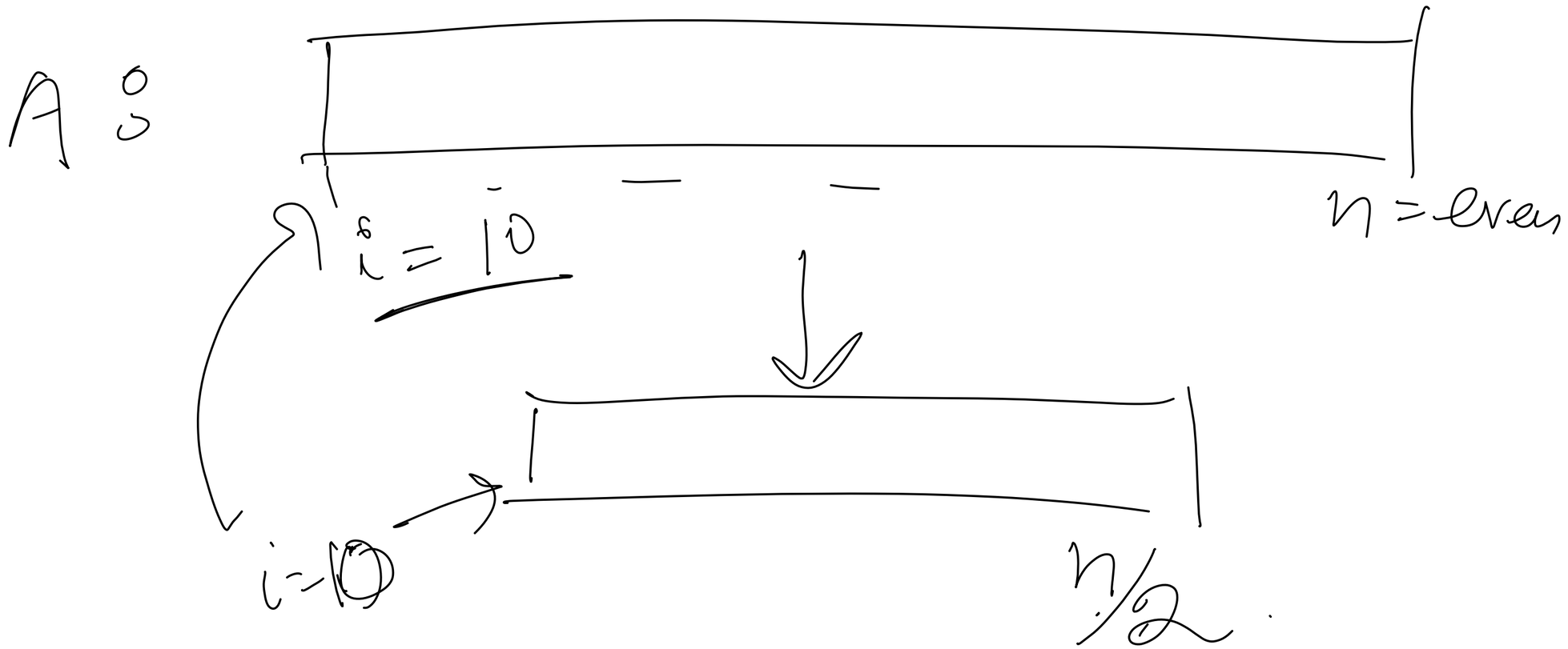
$$a = 2,\ b = 2 \longrightarrow n \log n$$

$$= a = 1;\ b \geq 1$$

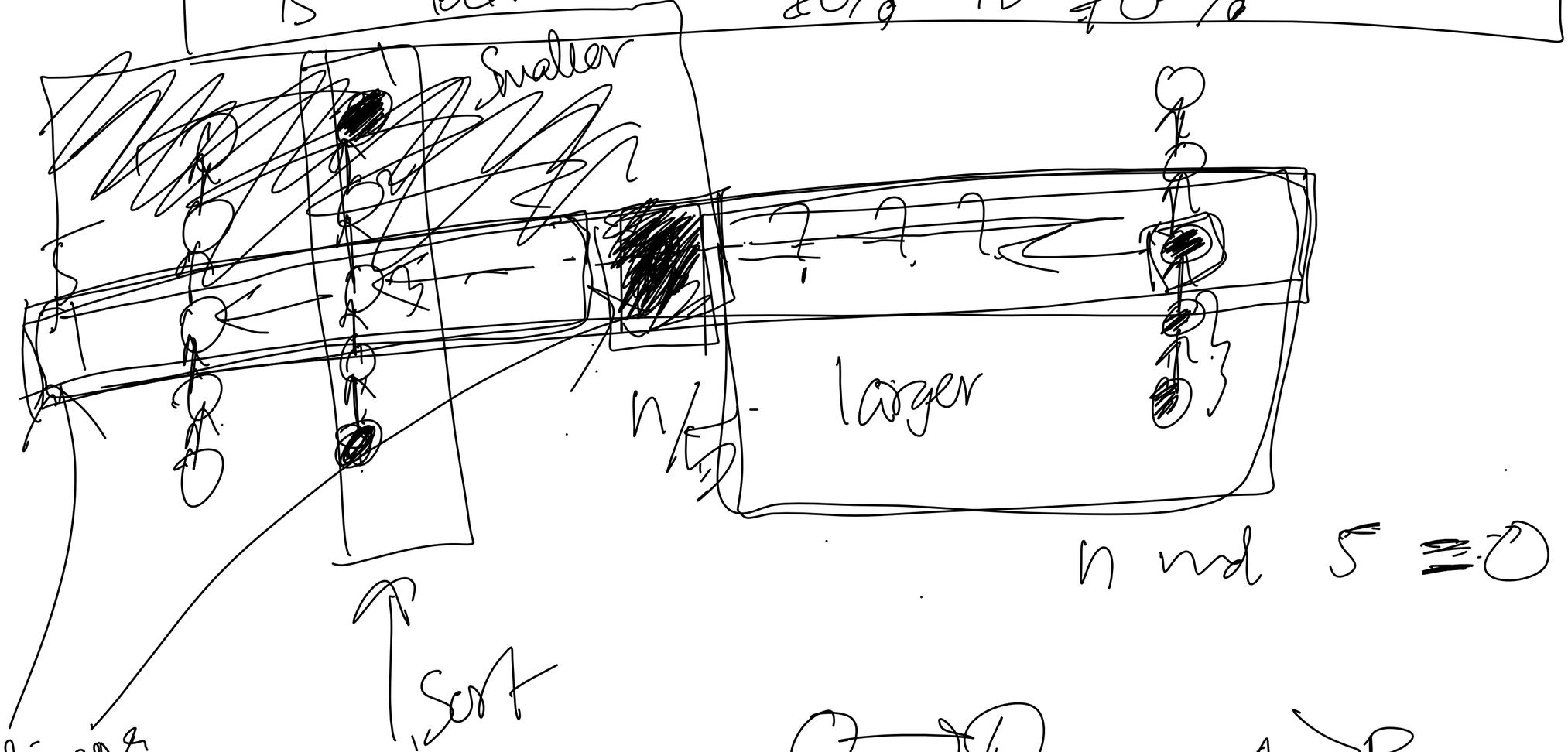
$T(n) = T(0.9n) + cn$

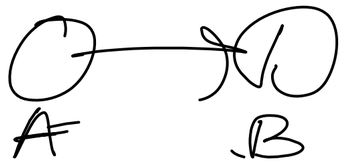
$\Rightarrow T(n) = \Theta(n)$

A :



need? Some element of the array whose rank is between 25% to 75%





A ≥ B

High-level pseudocode

(1) putting all items in a $(5 * g)$-grid

$g = n/5$.

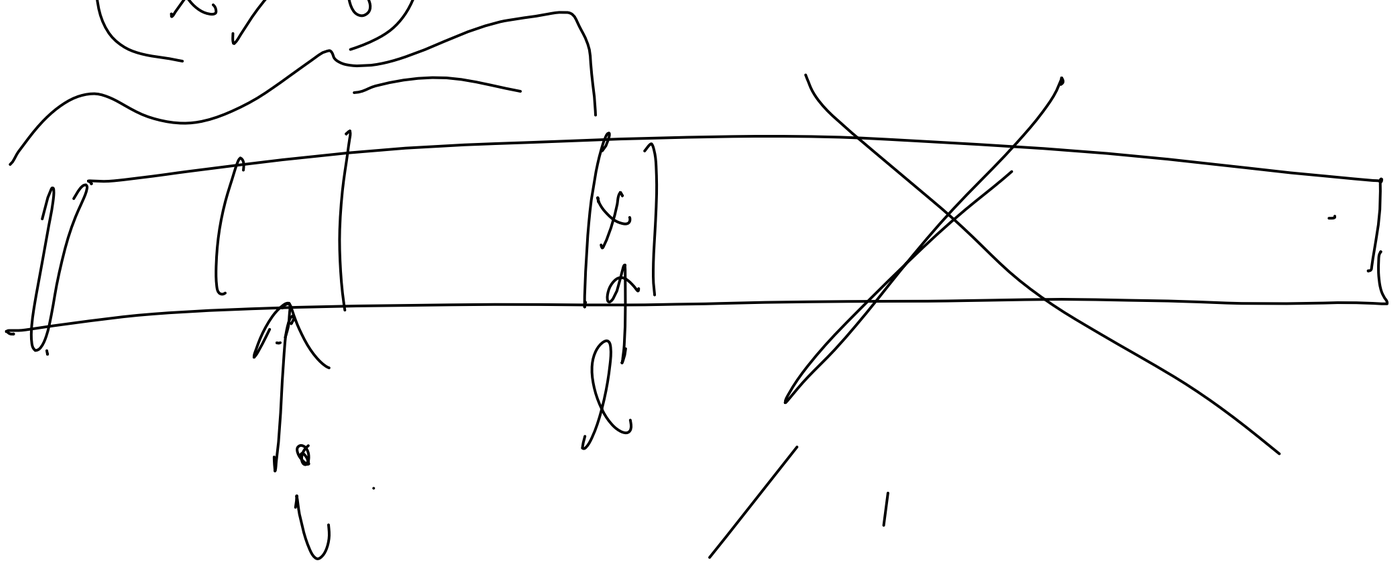
(2) Sort each of the g columns.

(3) find median of middle row →

(4) find out rank of $x \leftarrow l$

(5) if $(k == i)$ return X.

b) if $(k > i)$

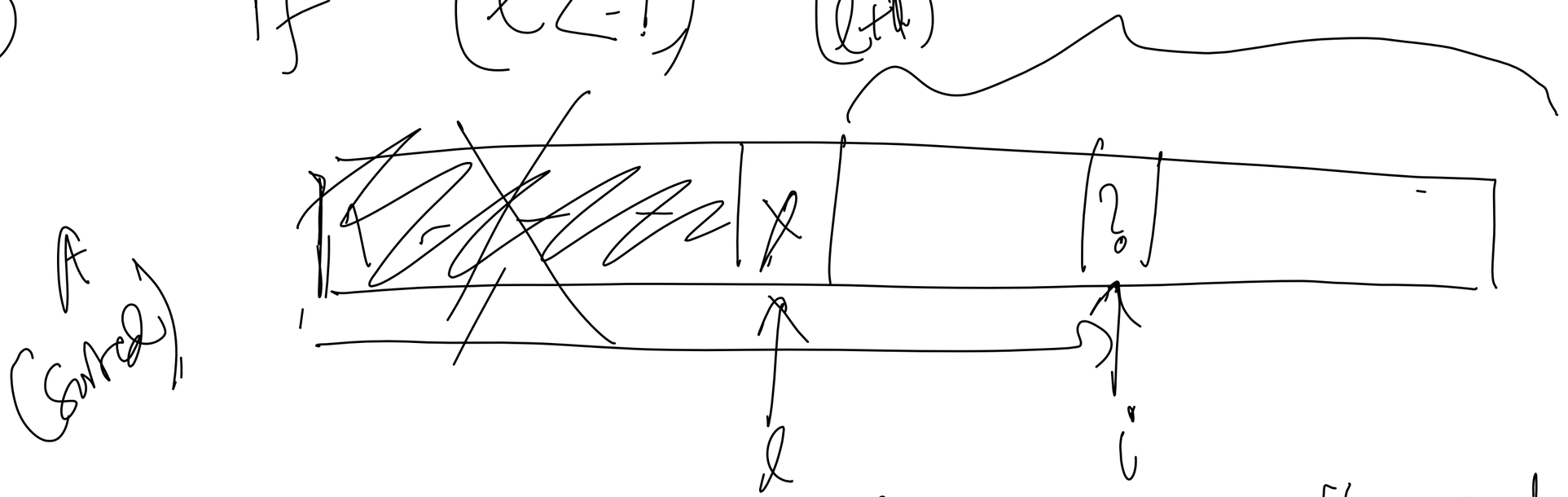


~~find~~

Create array B of all items with rank < k

Call ith statistic on B

(7) if $(\ell < i)$ $(\ell+1)$



Create an array B with all items having rank $> \ell$

Call $(i-\ell)$-th order statistic on B.

two recursive calls

- median : $n/5$ items
- other : $\leq 3n/4$ items.

$$T(n) \leq T(n/5) + T(3n/4) + cn.$$

~~or~~ $0.2n$ $\quad$ $0.75n$

$$T(0.95n) \quad \Theta(n)$$

$< n$

```
Procedure OrderStat(A, n, i)
    nmod5 ← n mod 5
    for p = 1 to nmod5
        // Assume 1 index.
        for j = p+1 to n
            if (A[p] > A[j])
                // Swap A[p] & A[j]
                =
                =
            elif ←
        end for
    end for
```

// Create a new array B with all but
// first n mod 5 items of A

$m \leftarrow sizeof(B)$

// Make $g * 5$ and C.

$g \leftarrow m/5$

$C \leftarrow$ empty $g * 5$ array.

$k \leftarrow 1$

$i \leftarrow i - (n \bmod 5)$

for p = 1 to g.

for j = 1 to s

C[p][j] ⟵ B[k]

k++

end for

end for.

// find median of C[3]

//

// create a new array D.

// D[1...11] have median of

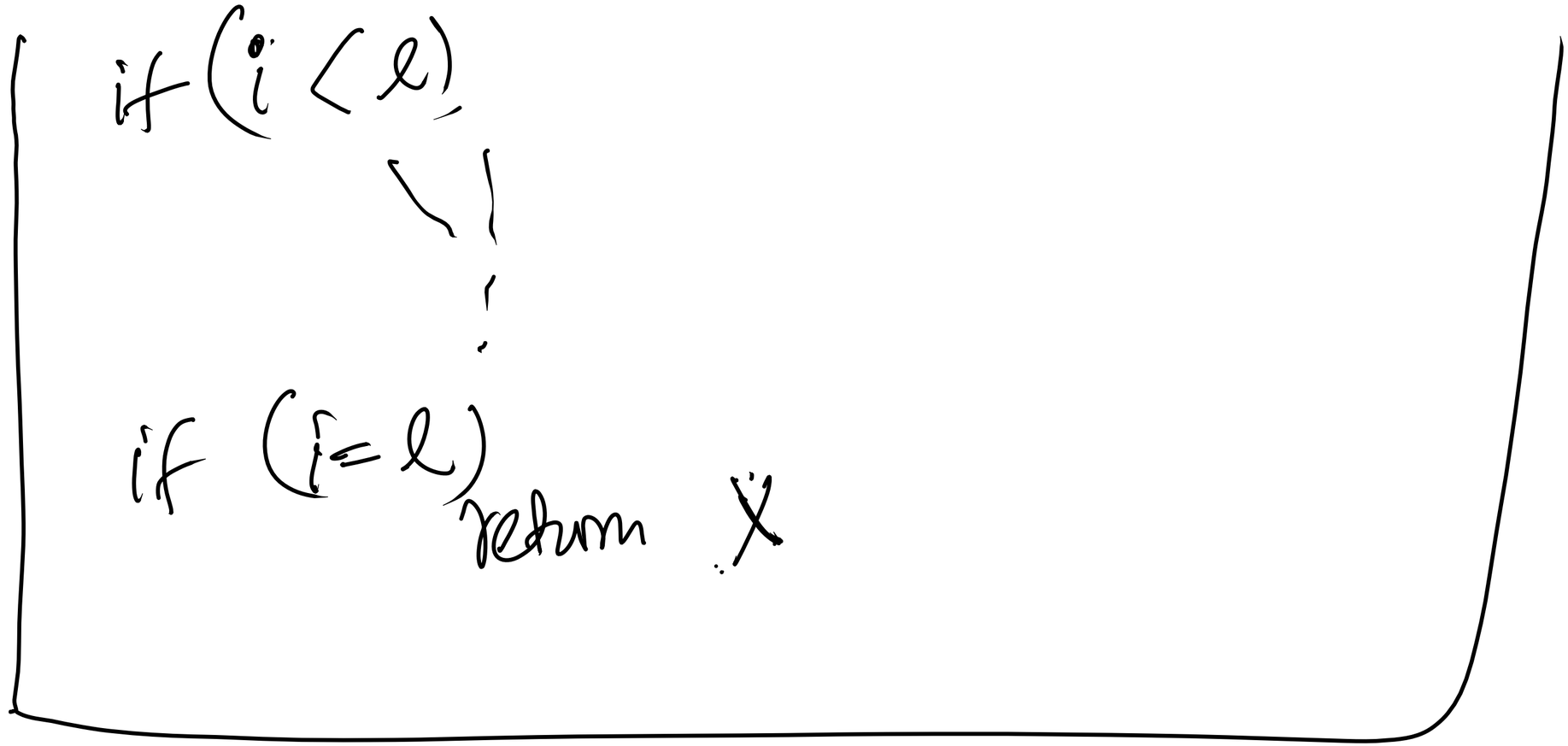
// C[1][j], C[2][j] C[3][j], C[4][j], C[5][j]

D ←

X ← OrderStat (D, 9, ⌊9/2⌋)

```
count ← 0
for p ← 1 to m
    if (B[p] < x)
        count ← count + 1
    end if
end.
l ← count + 1   // rank of x
```

```
if (i > ℓ)
    F ← empty array of size
                (m − ℓ) items.
    K ← 1
    for (j = 1 to (m − ℓ))
        if (B[j] > x)
            F[K] ← B[j]
            K++
return OrderStat(F, m − ℓ, i − ℓ).
```


if (i < l)
if (i=l)
return X