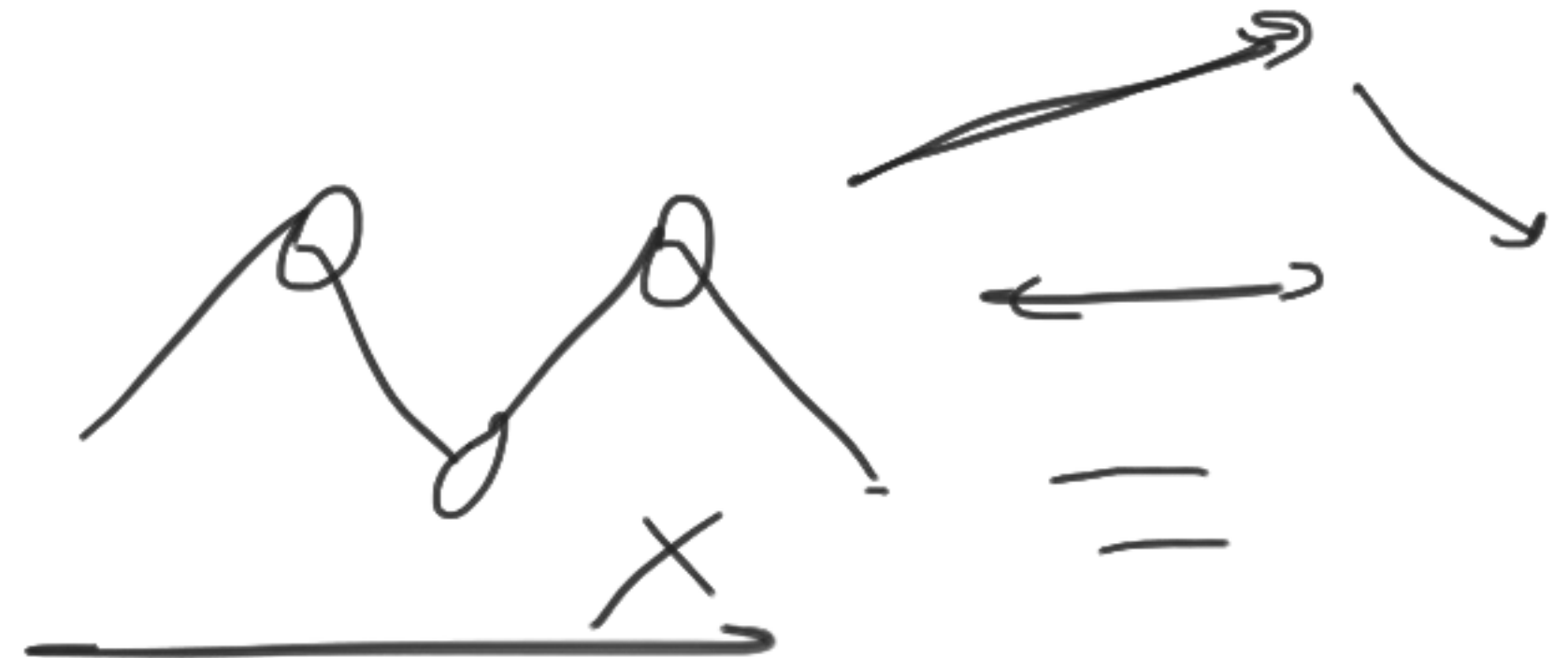
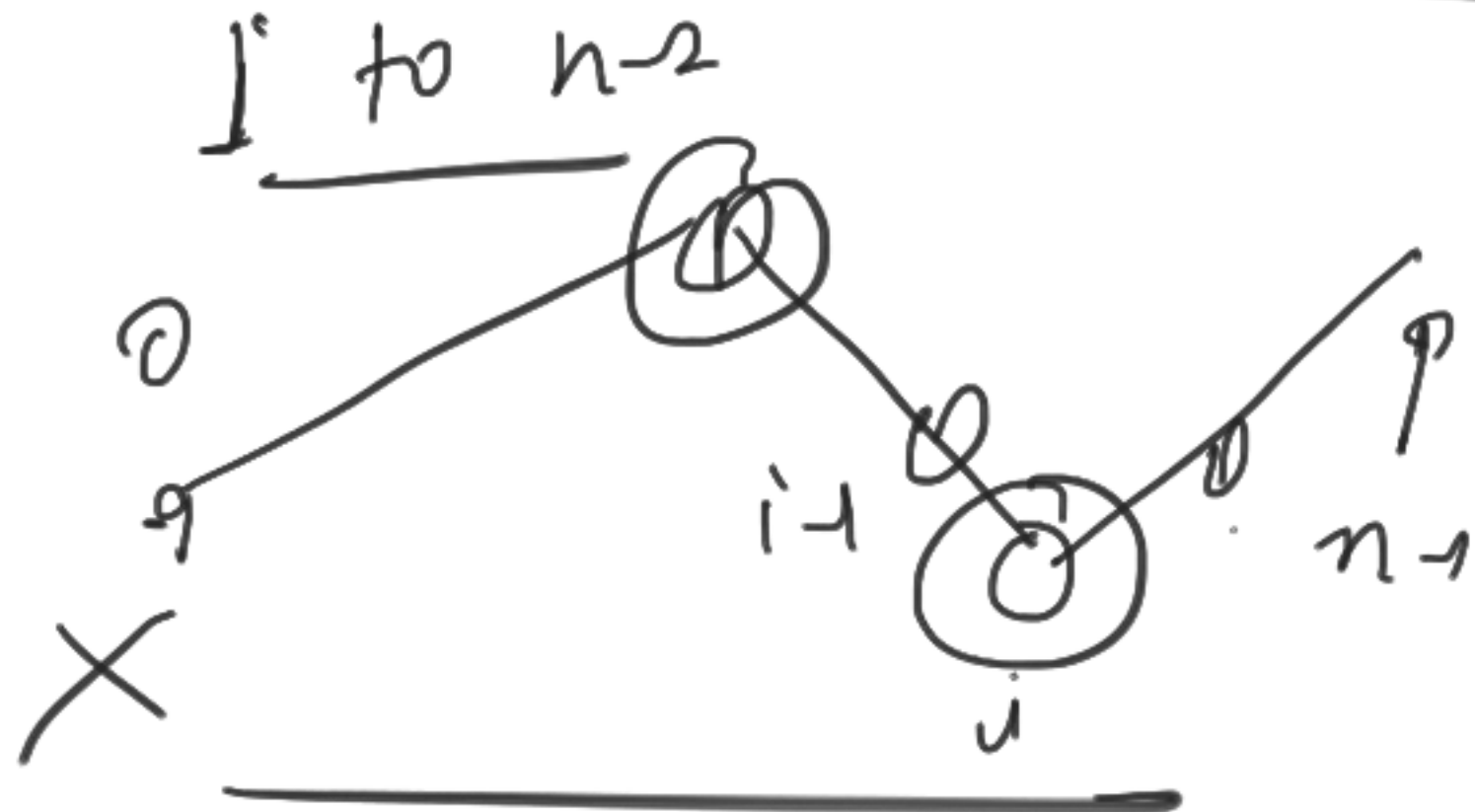


why



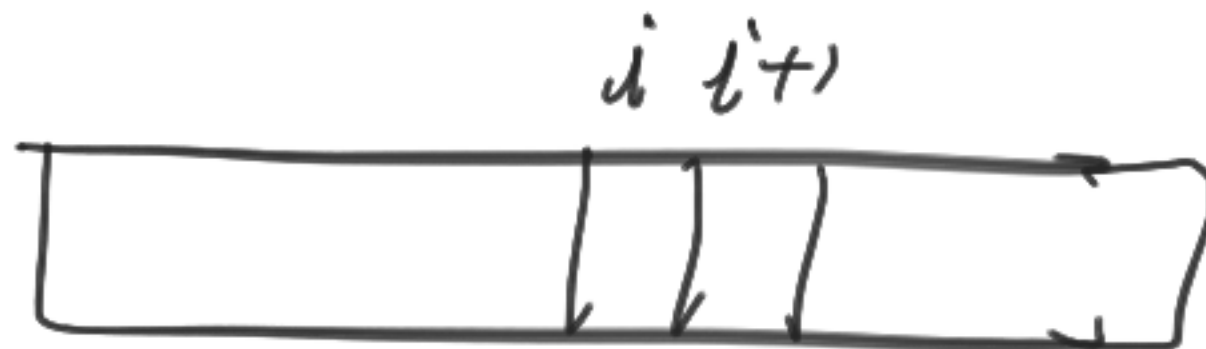
\Rightarrow

max and min

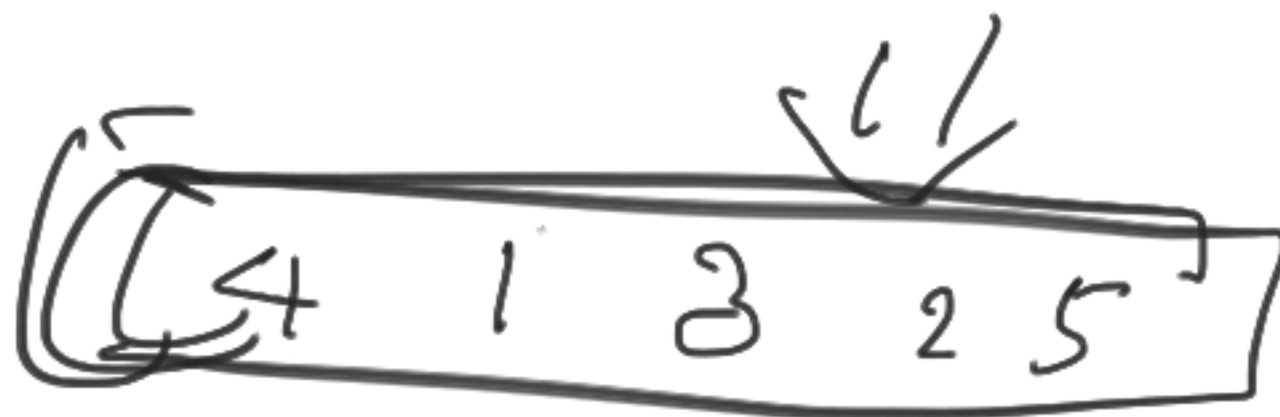
$$l = i + 2$$

\Rightarrow

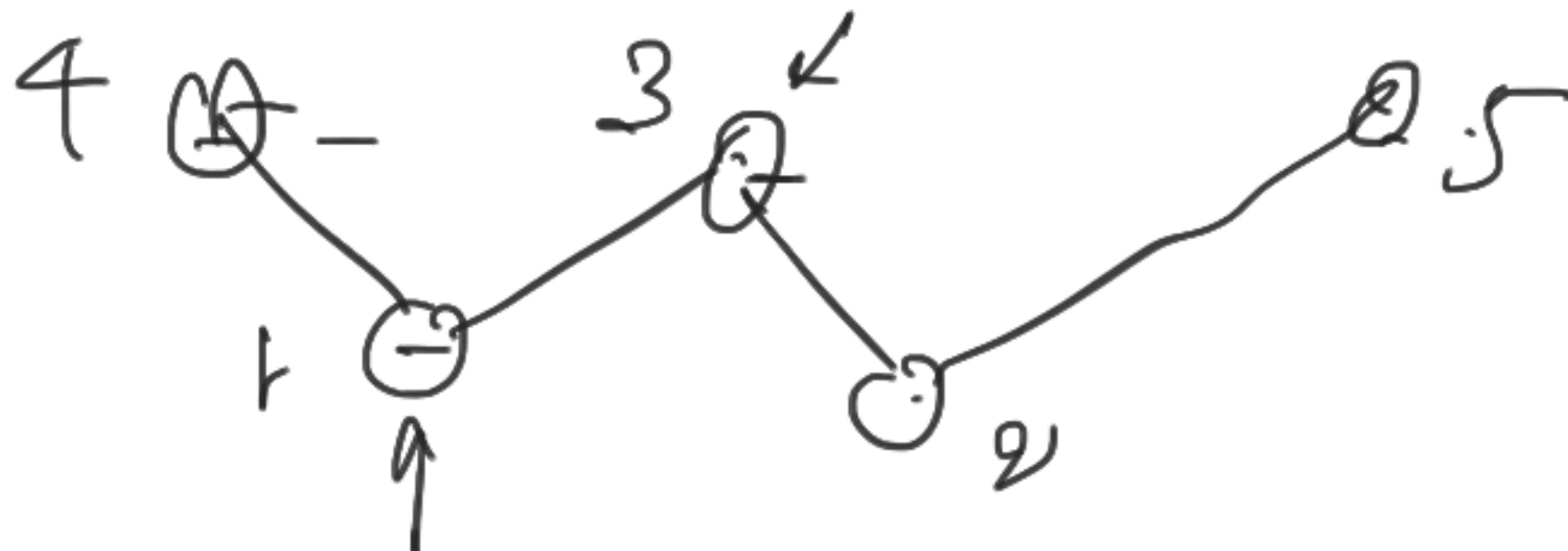
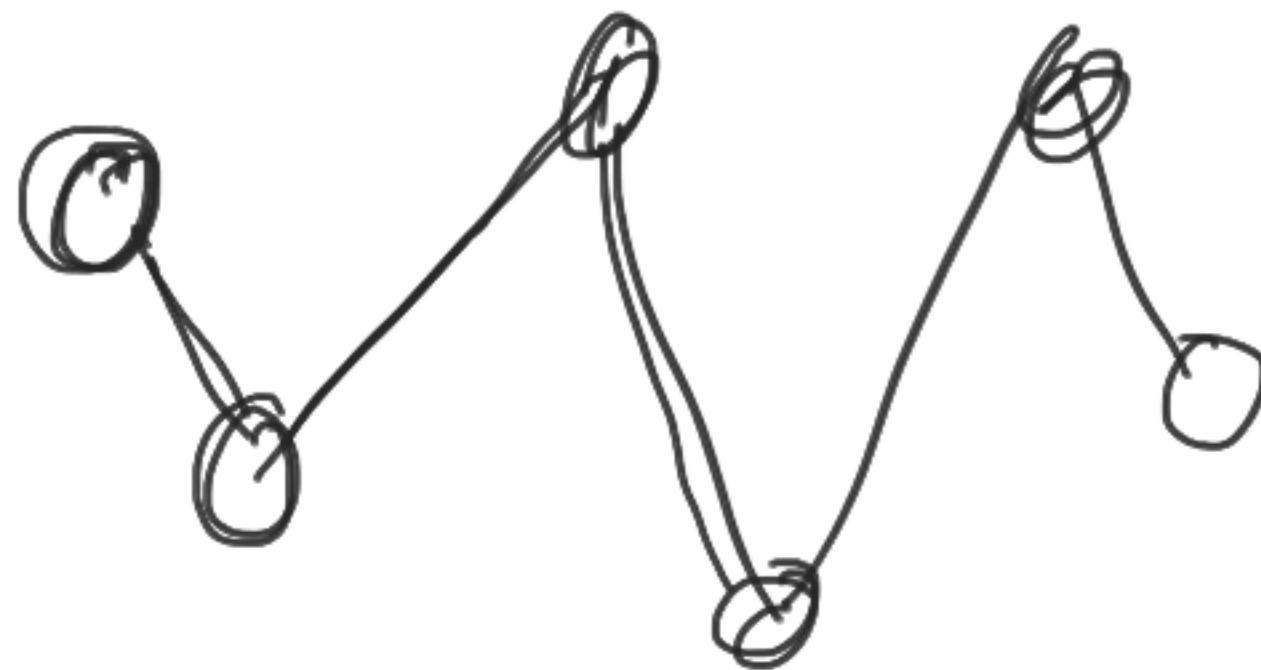
wave array

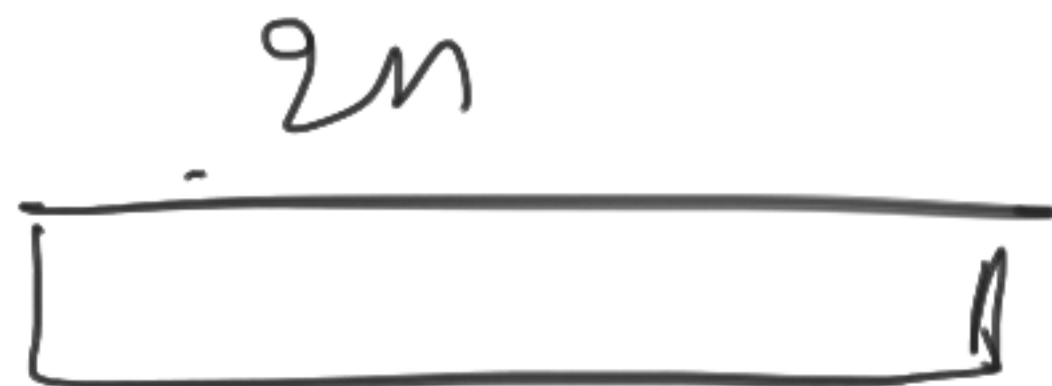
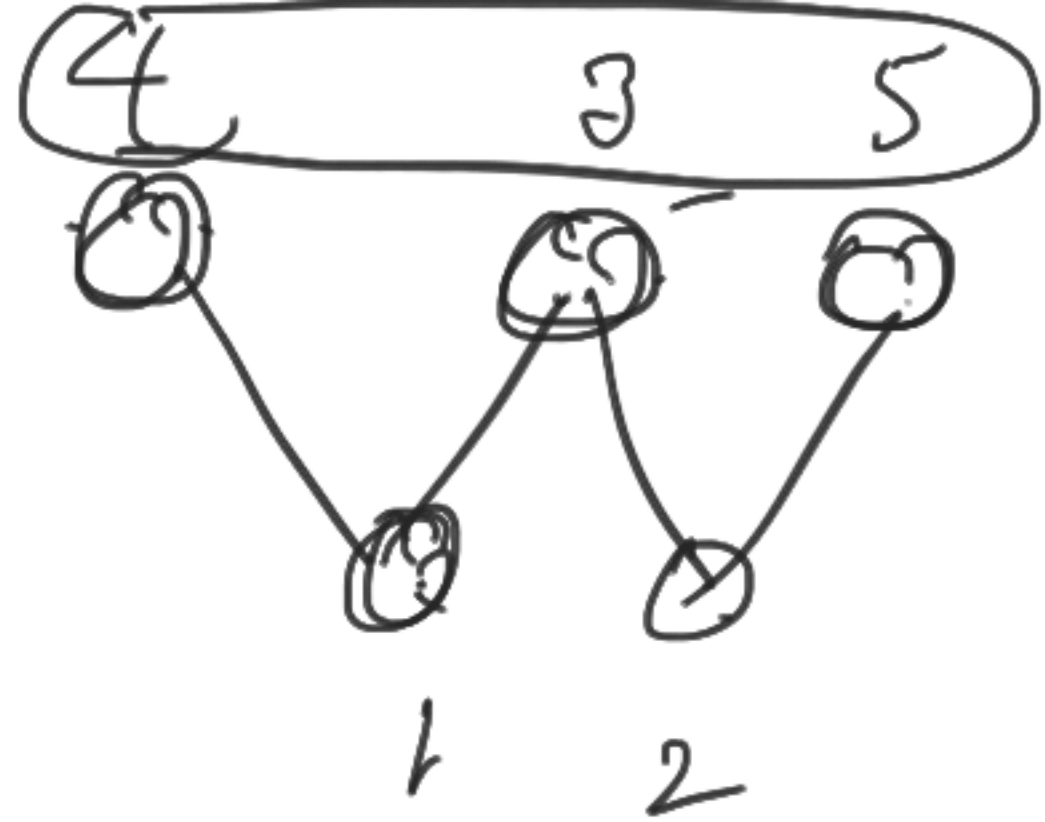


A12



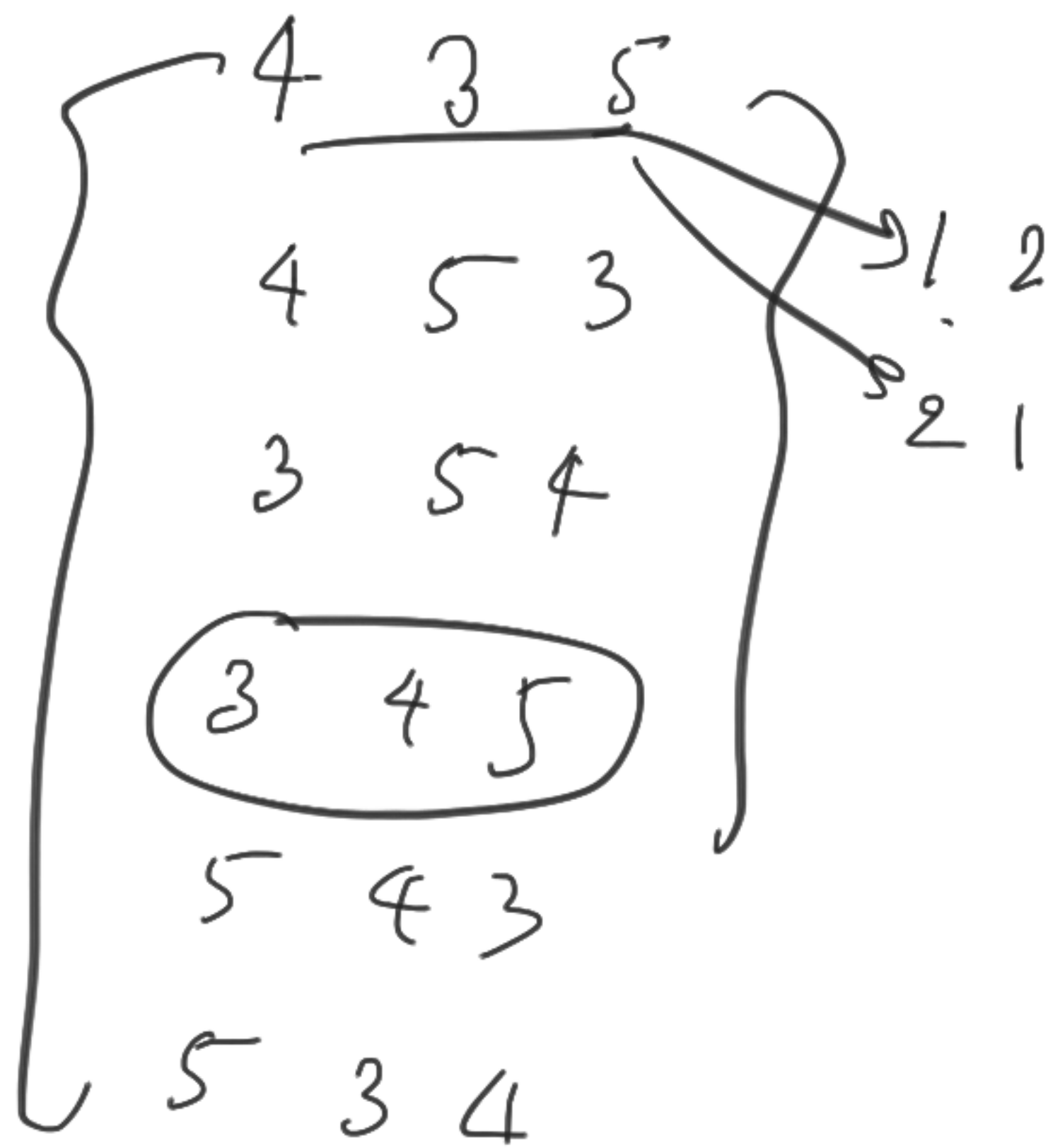
\Rightarrow

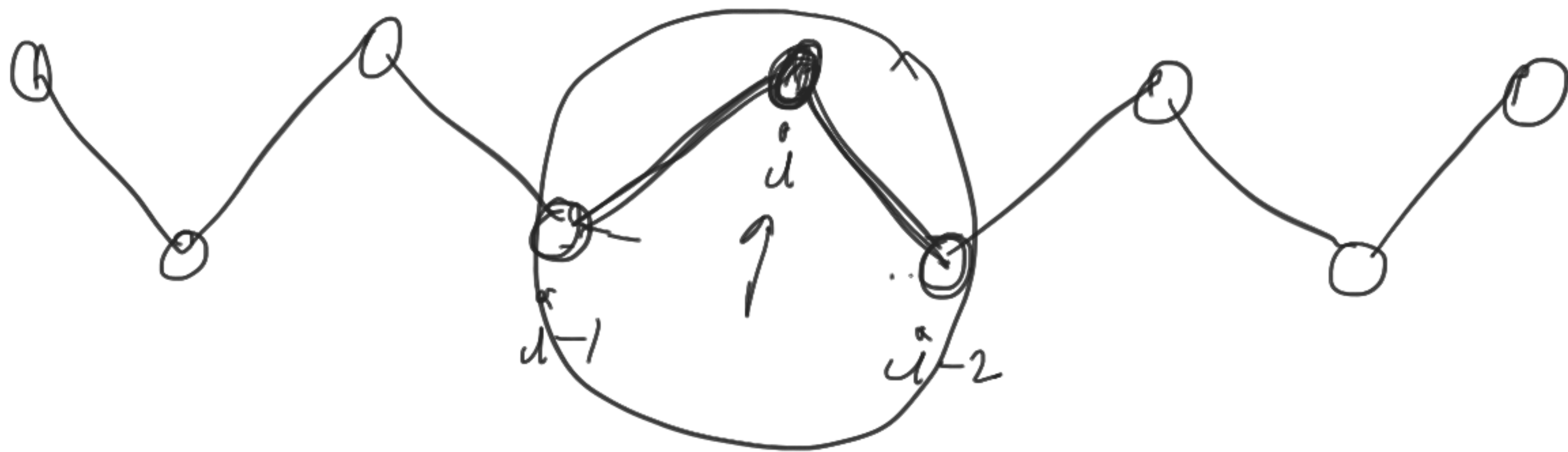
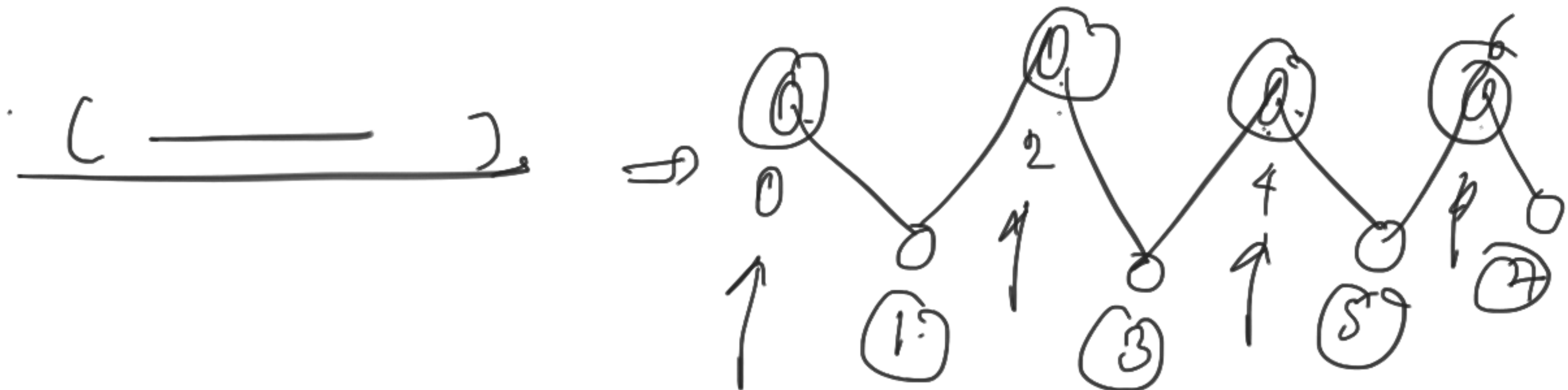


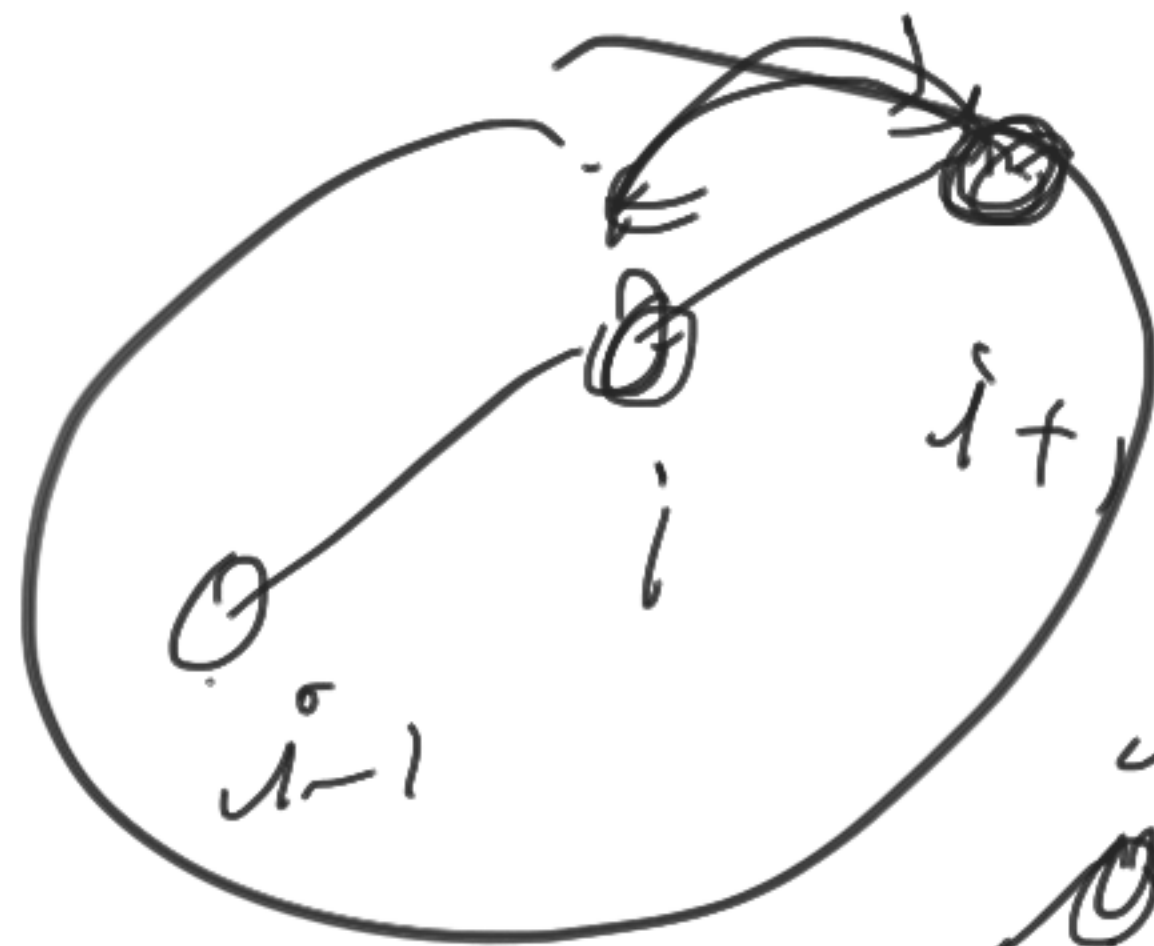
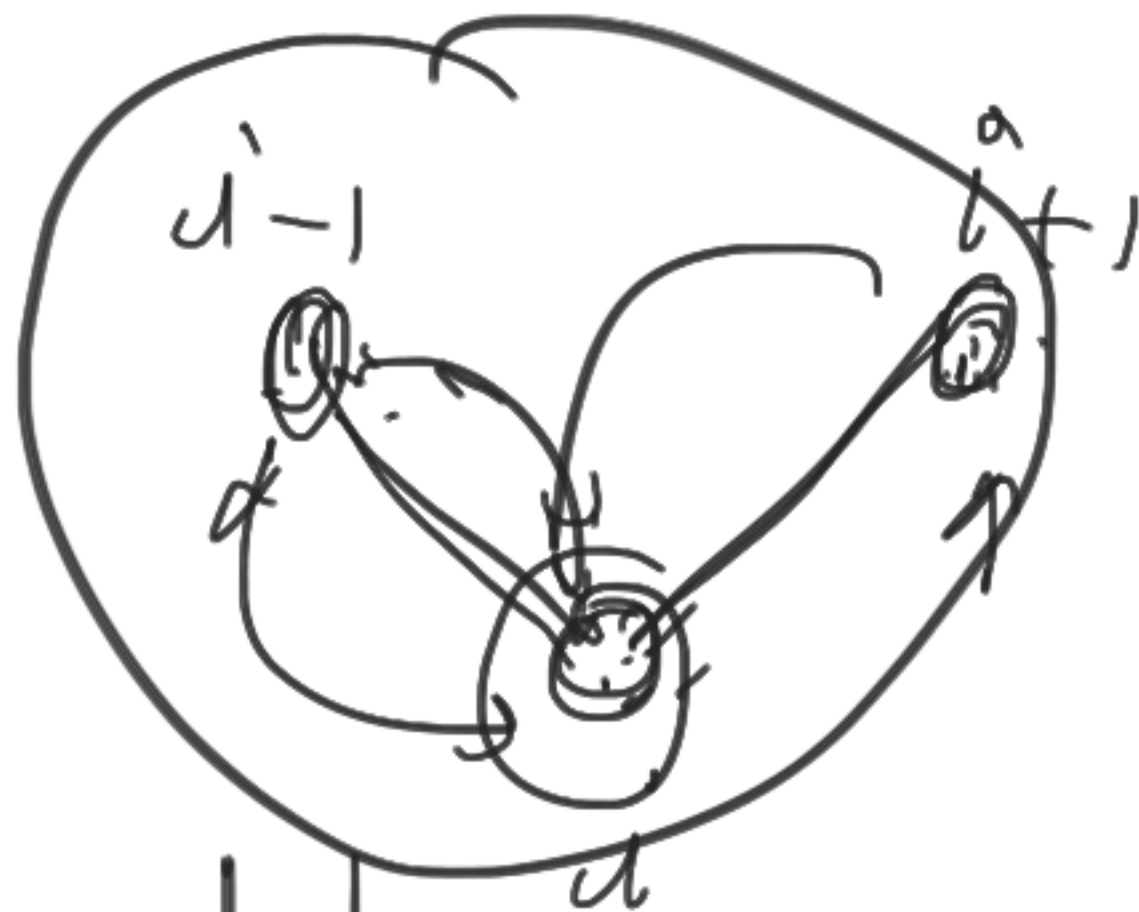
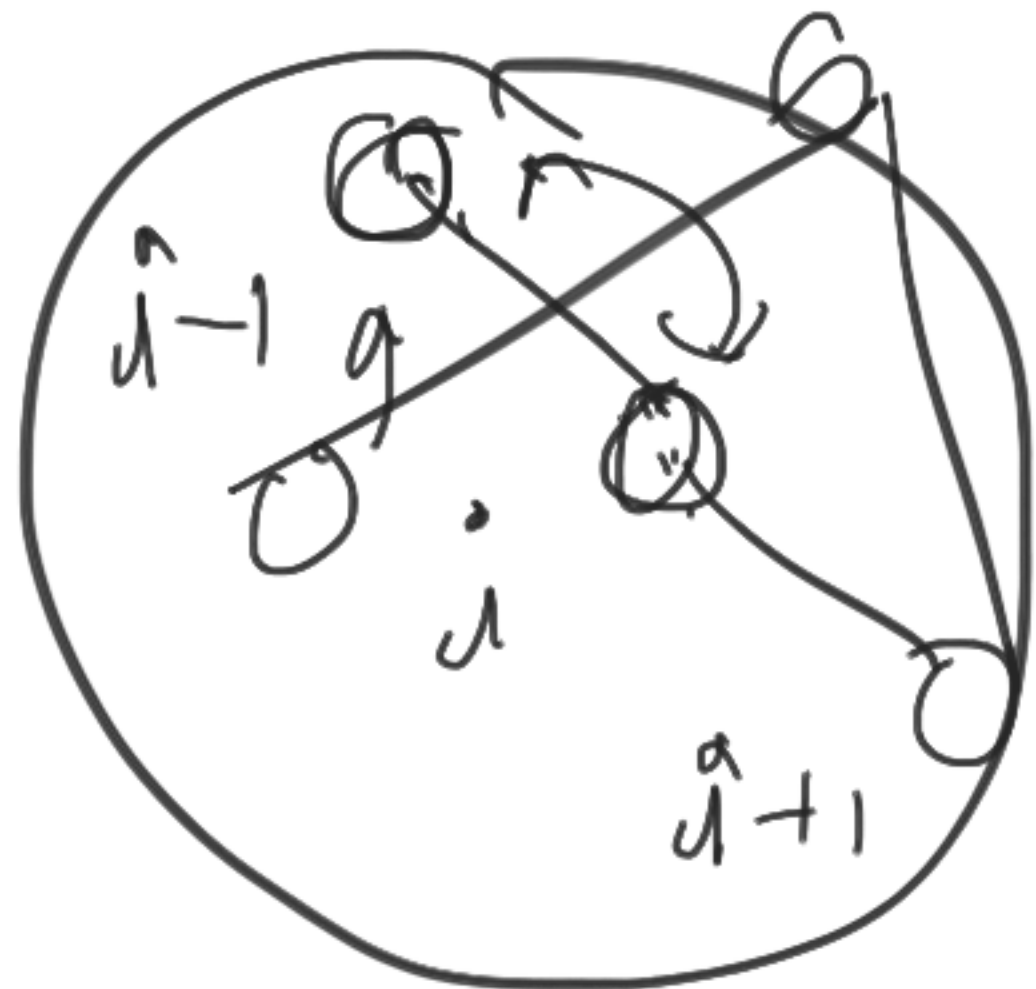


n points $\Rightarrow \lfloor n \rfloor$

n vertices $\Rightarrow \lfloor n \rfloor \Rightarrow (\lfloor n \rfloor)^2$

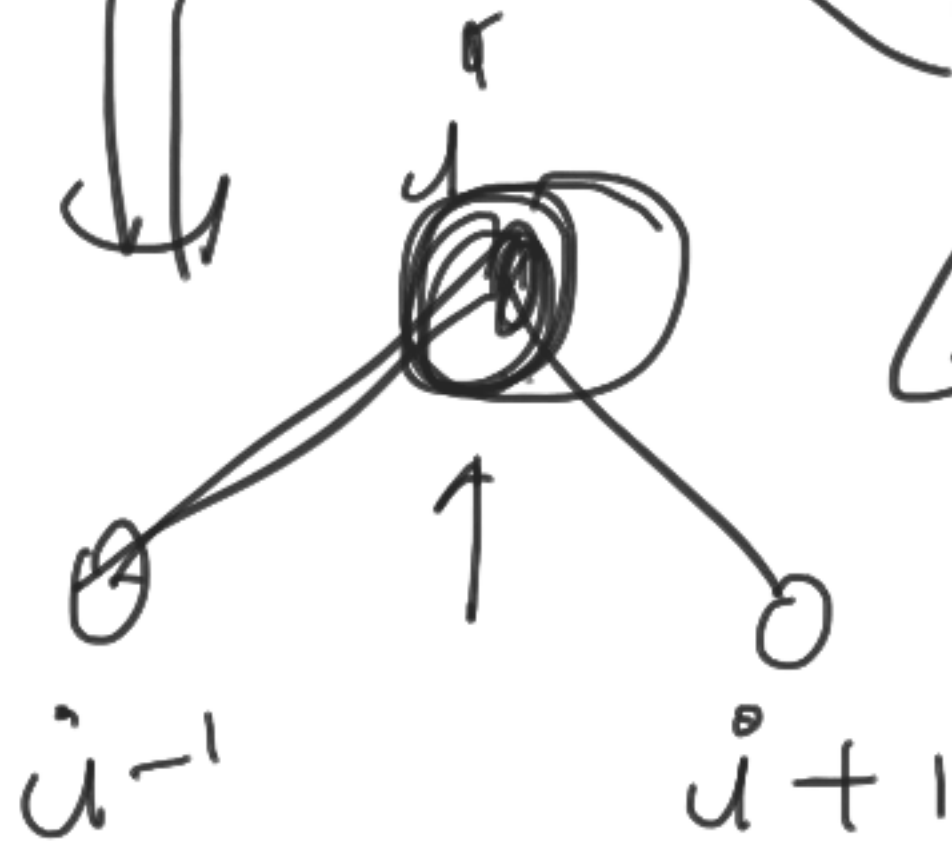






\approx

\Downarrow



\Downarrow

\Rightarrow

$0 \quad 2 \quad 4 \quad i-1 \quad \dots \quad i+1$



$$(A[i-1] \leq A[i] \geq A[i+1])$$

↑

↑

↑

$$O(n)$$

$i-1$ i $i+1$



$$i = 0 \text{ to } n-1$$

$$i = i + 2$$

$$i = n-1$$

$$i > 0 \ \& \ i < n-1$$

$$\text{if } (A[i] < A[i-1])$$

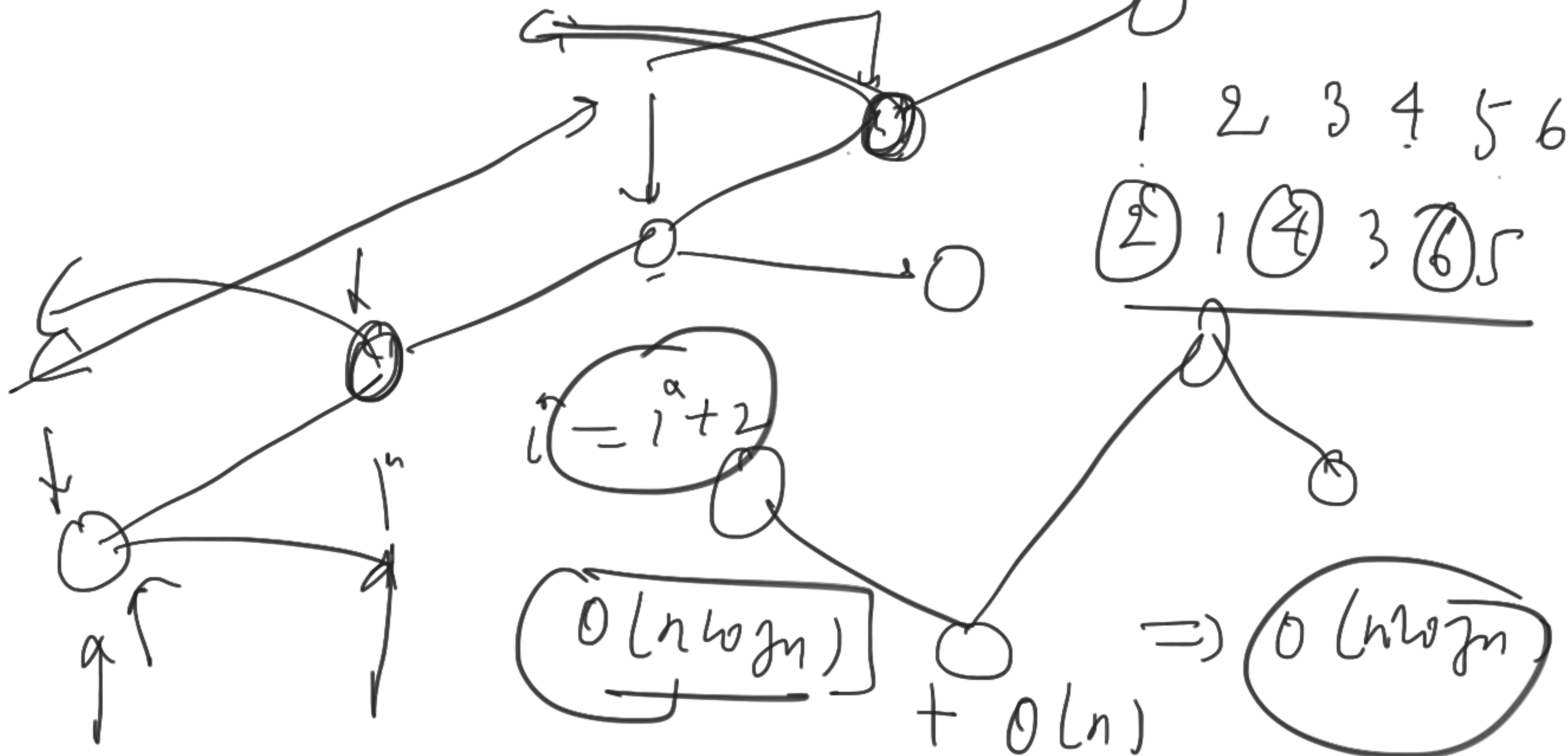
$$\text{Swap}(i, i-1)$$

$$i < n-1$$

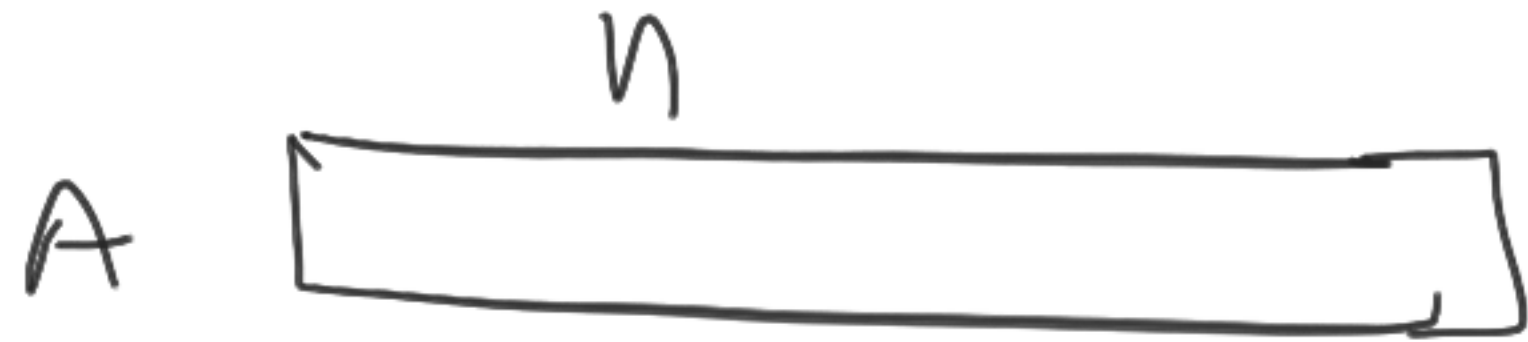
$$\text{if } (A[i] < A[i+1])$$

$$\text{Swap}(i, i+1)$$

Sort the array

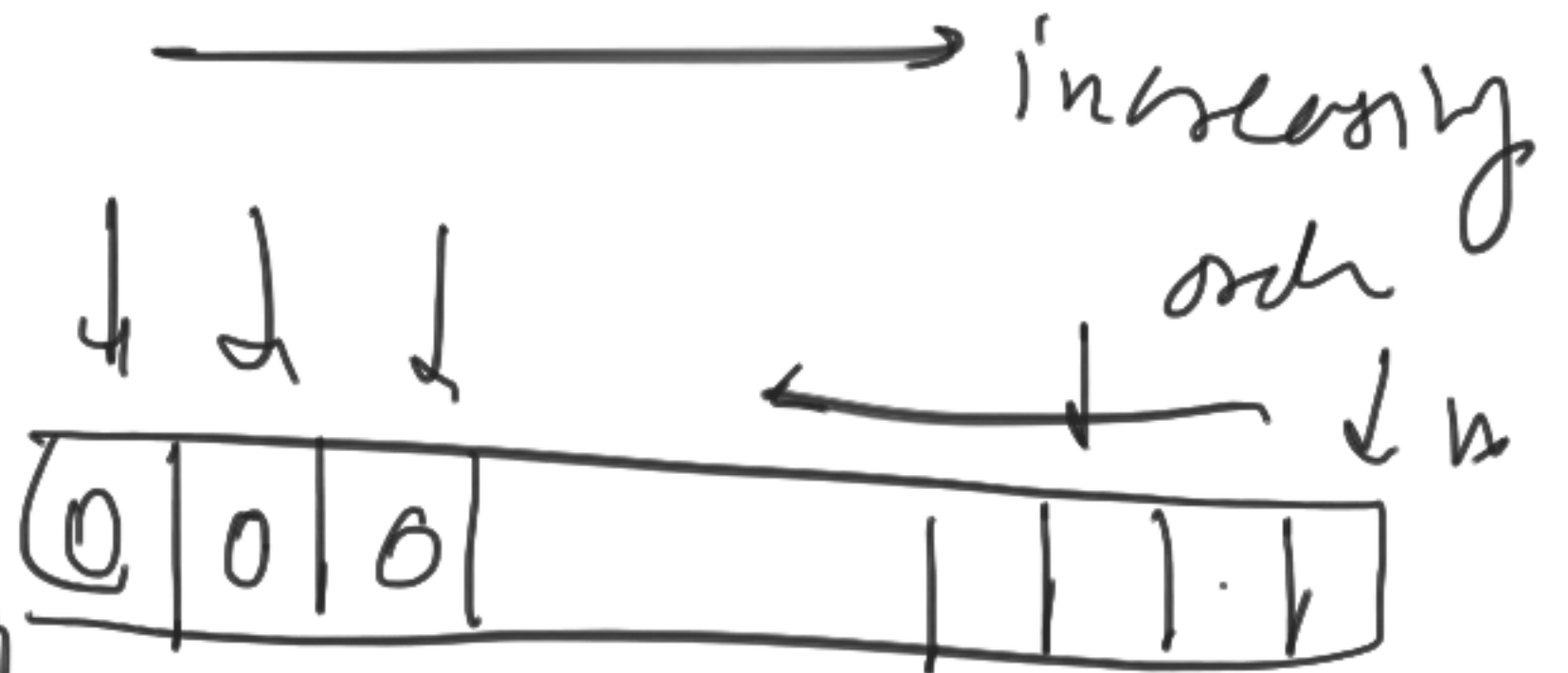


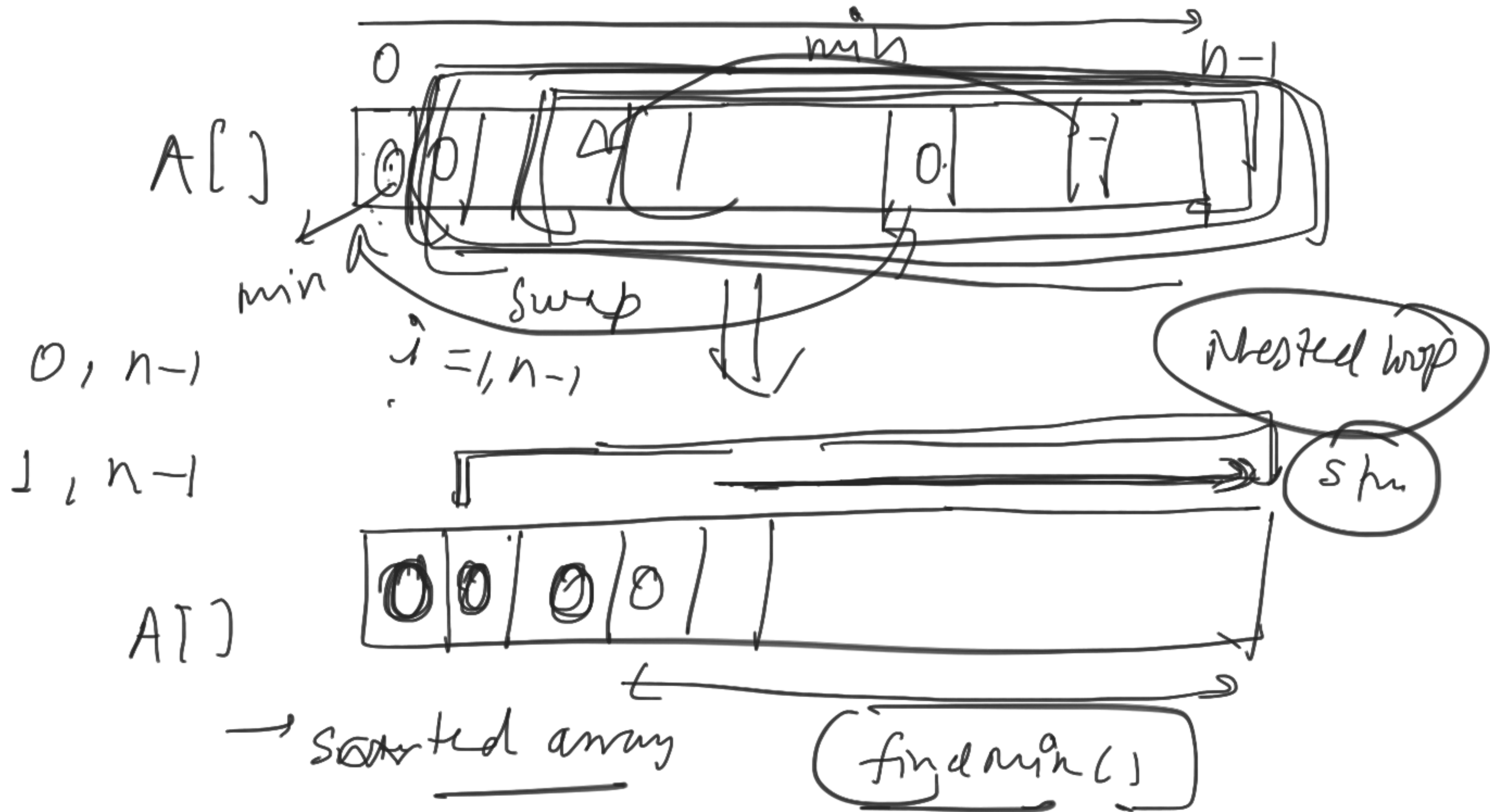
Selection sort
insertion sort



$$A[i-1] \leq A[i] \leq A[i+1]$$

$$A[0] \leq A[1] \leq \dots \leq A[n-1]$$

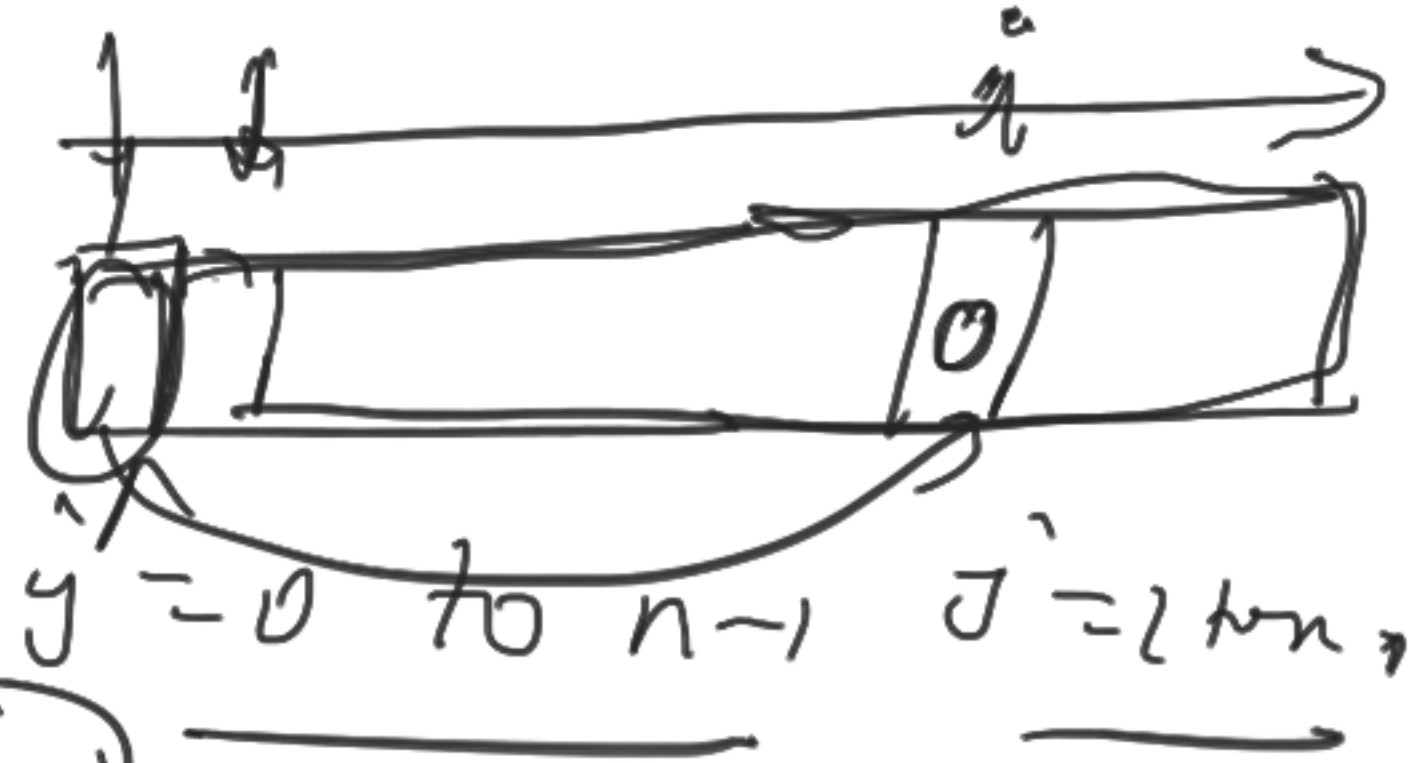




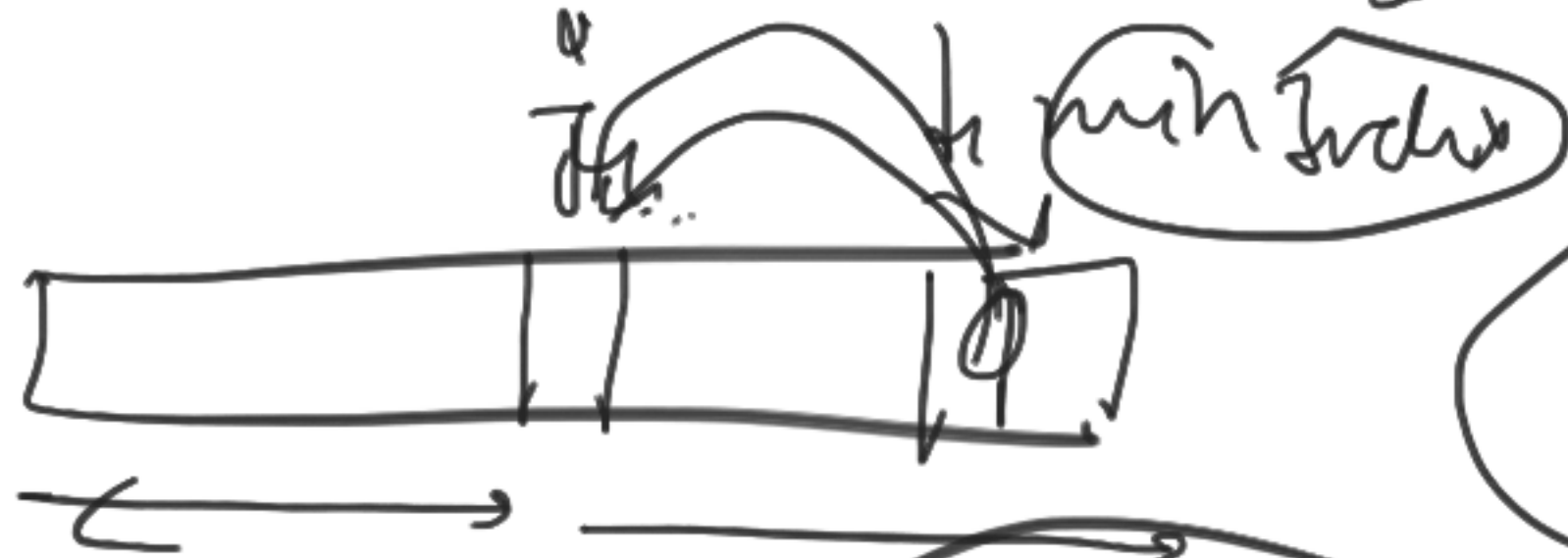
for ($i = 0$ to $n-2$)

($l+1$ to $n-1$)

for ($j = i$ to $n-1$)



($n-1$) loop

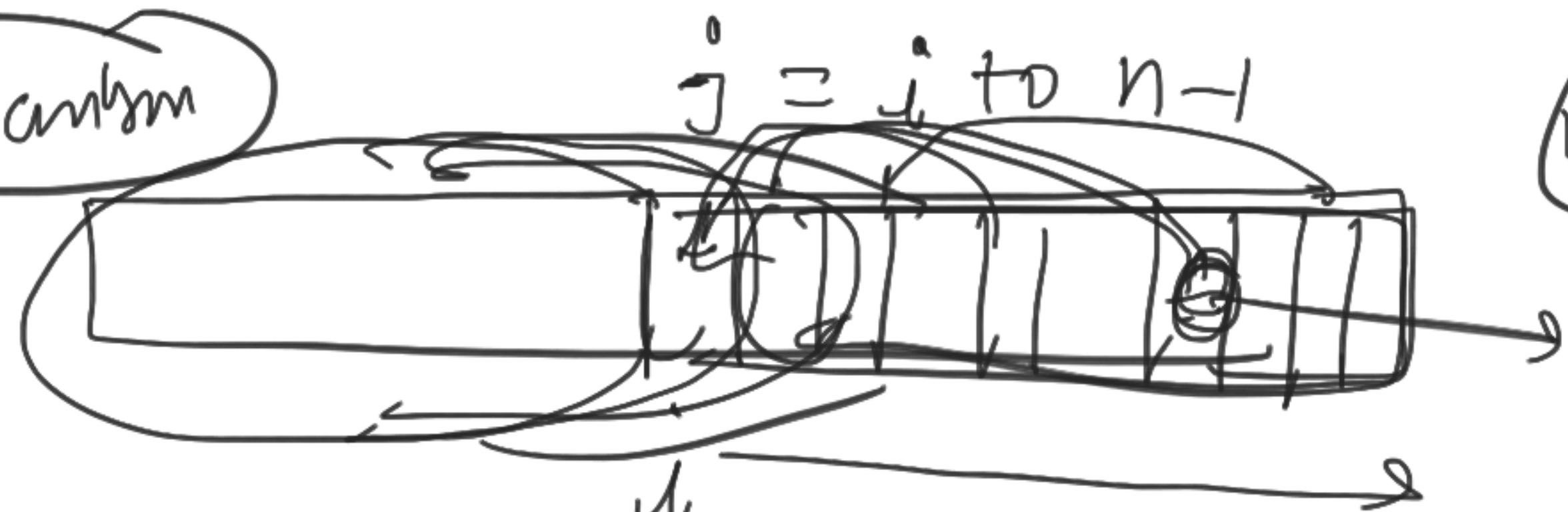


Comparison



Swapping

Comparison



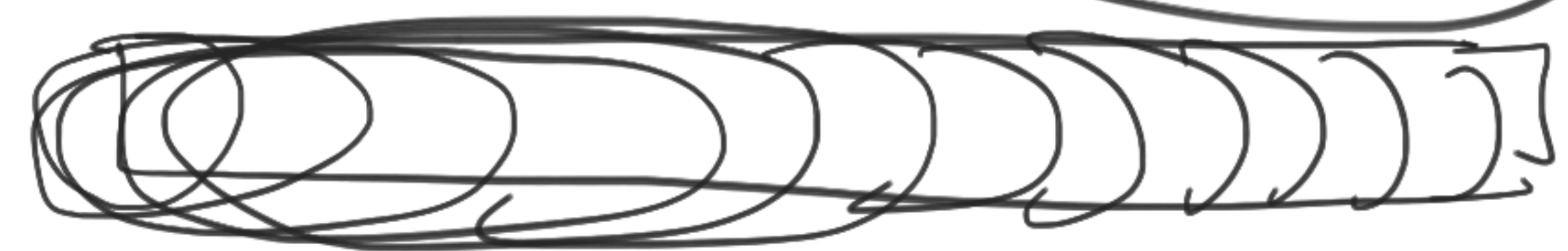
$\text{minIndex} = i$

minIndex

time complexity j

n swapping

$n-i$ \rightarrow $O(n^2)$



$\frac{n(n+1)}{2}$

$n + n-1 + n-2 + \dots + 1$

Why sorting? \Rightarrow in-depth

Sorted \rightarrow binary search

$O(\log n)$

\Rightarrow Recursive

\Rightarrow Iterative

Single loop \swarrow
Two Pointers

Process the data

inserts



$\Downarrow O(n \log n)$



Sorted



merge sort

$O(n \log n)$



merging

$O(n)$

✓✓

counting sort

$O(n)$

\Rightarrow

Direct address
table

heap sort

\Rightarrow



Partition

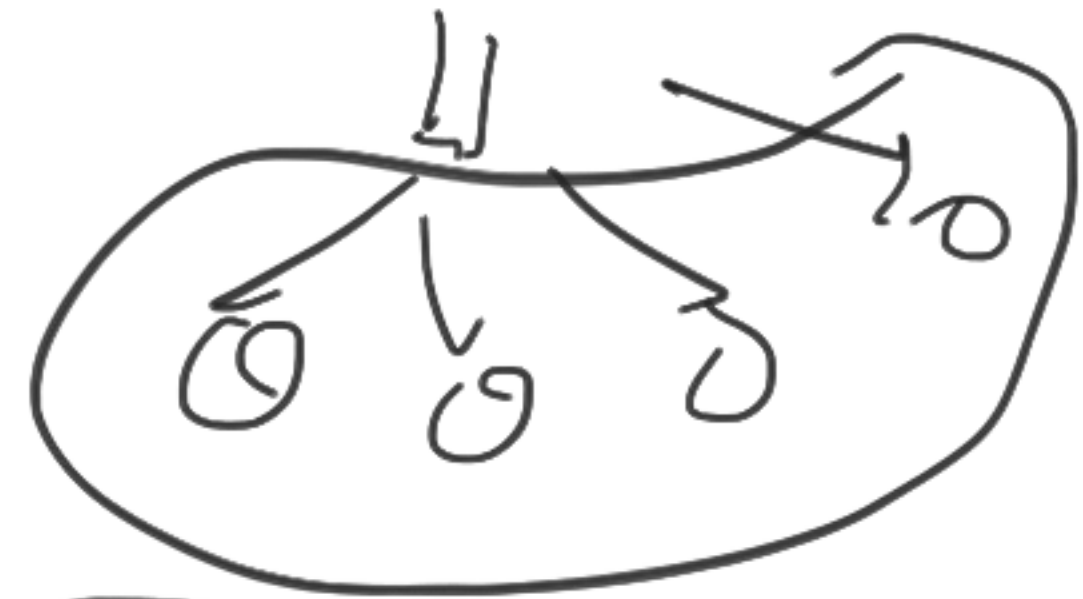


Two-point

Divide (merge sort)

Binary search

quick sort

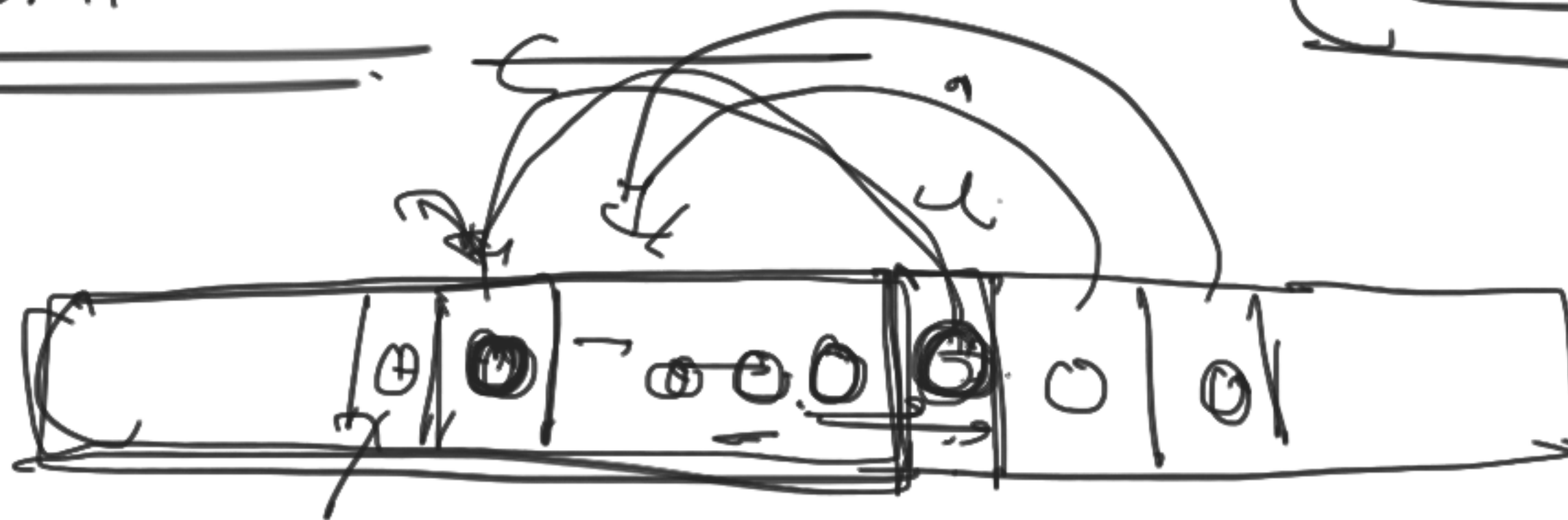


Insertion

26



Insertion Sort



$O(1)$



Sorted

→

+1

Unsorted part

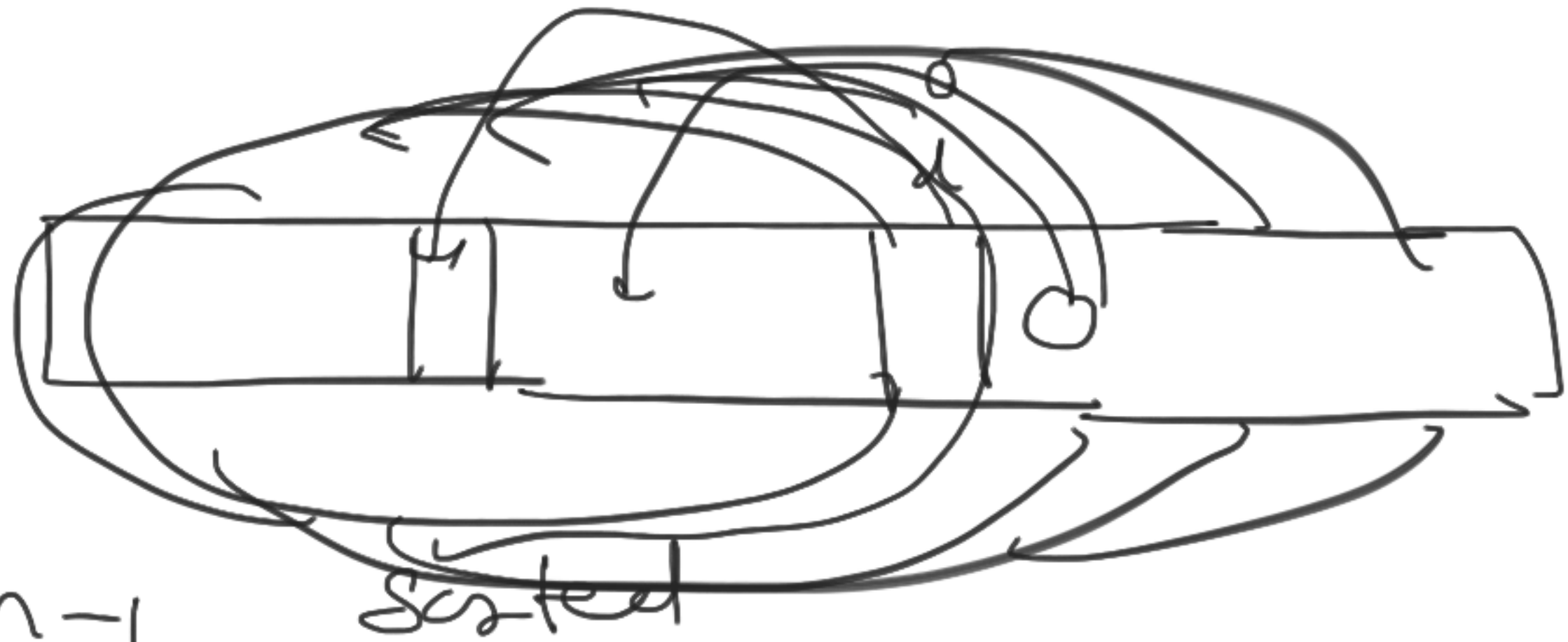
⇒

1

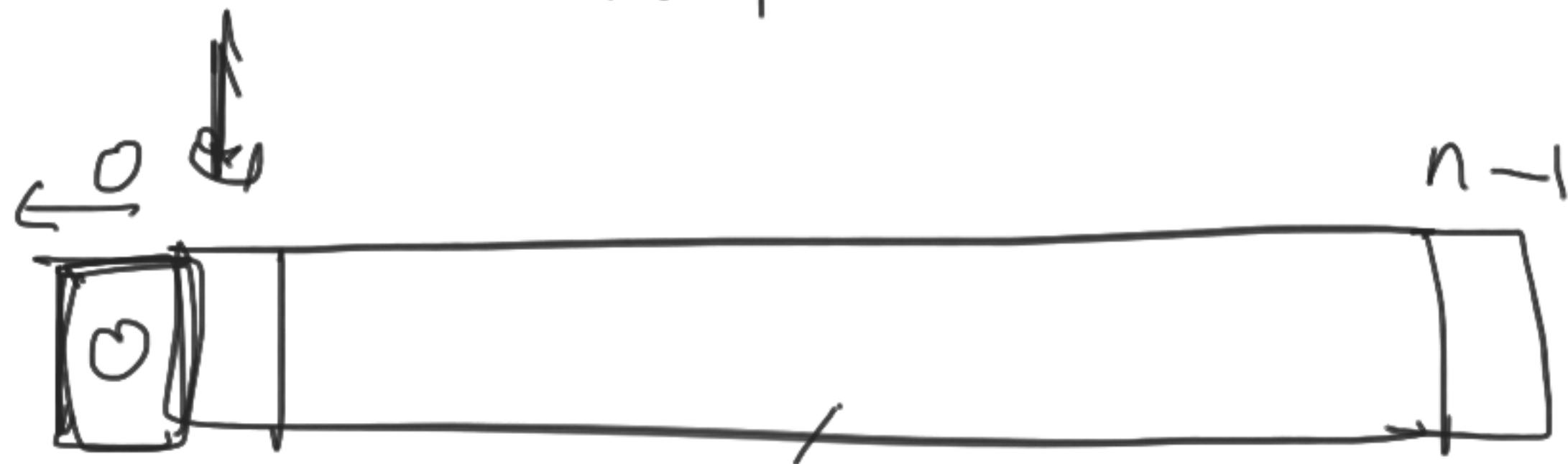
→

i

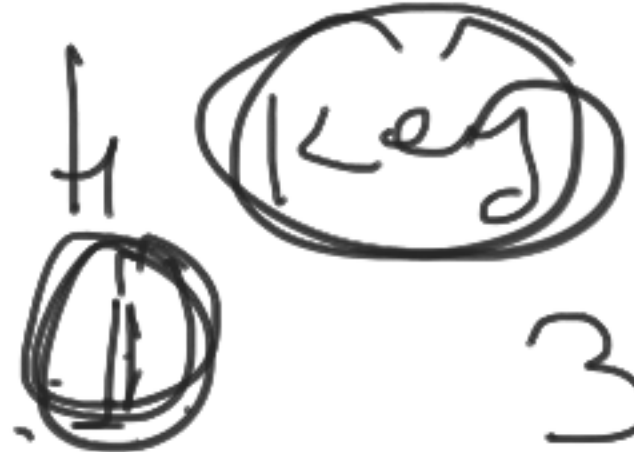




$d = 1 \text{ to } n-1$



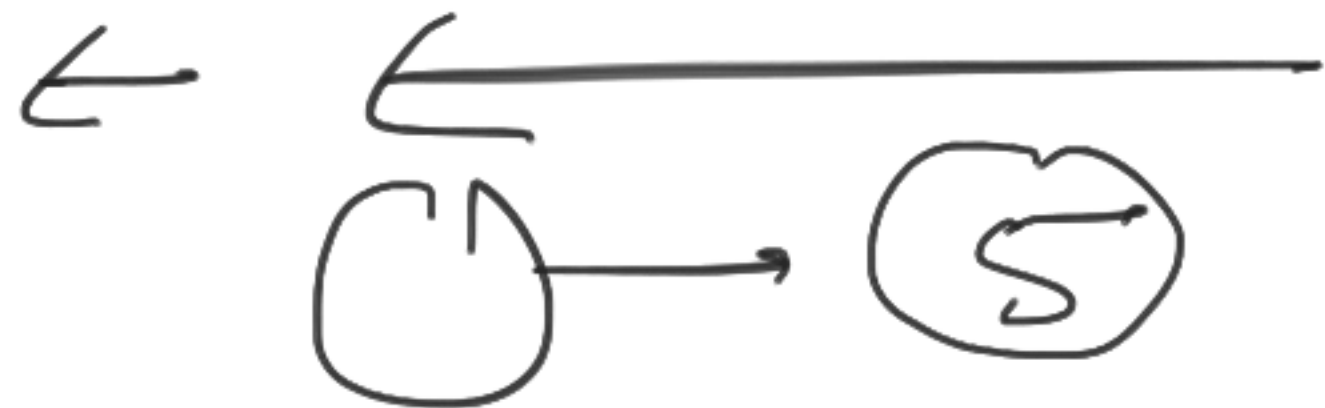
S



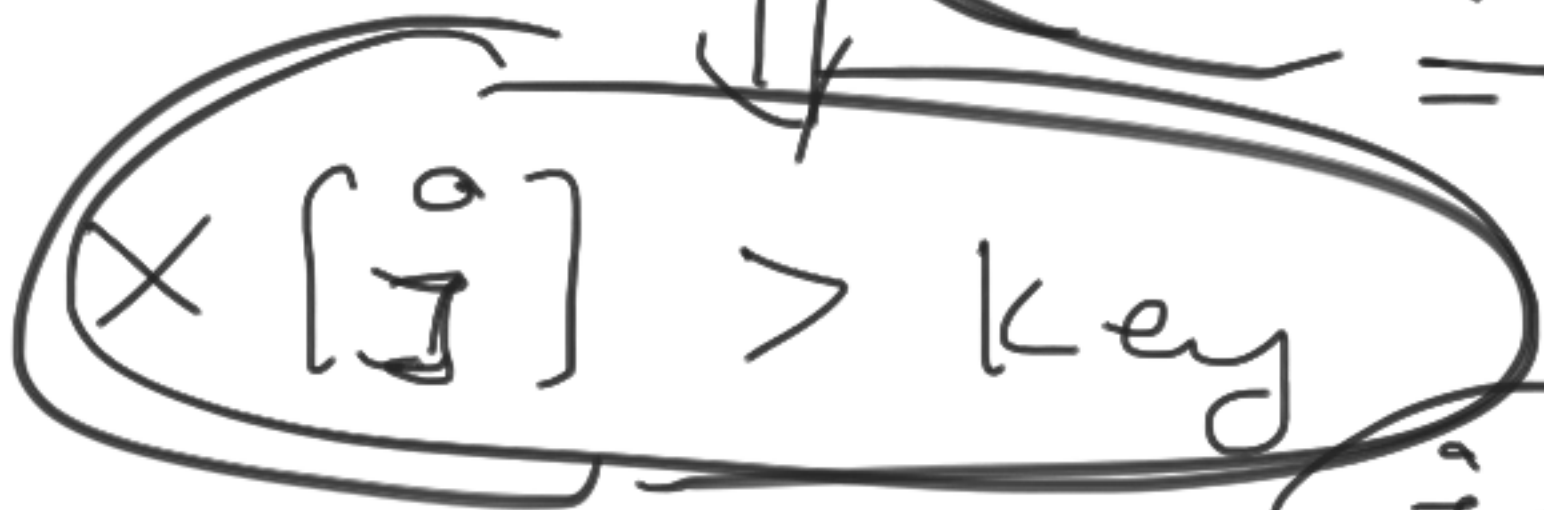
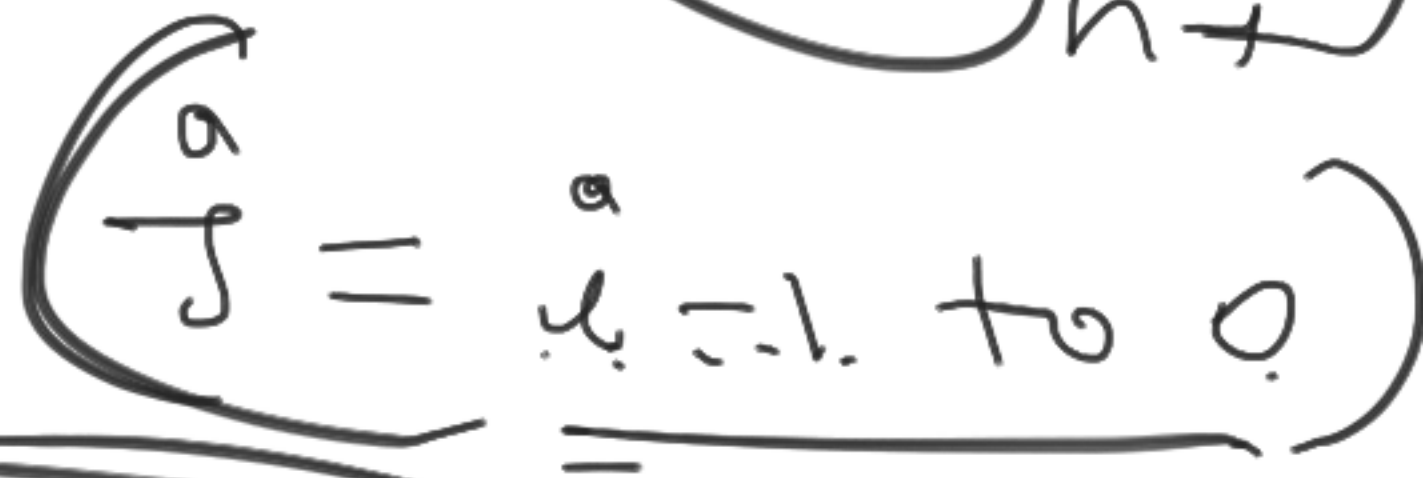
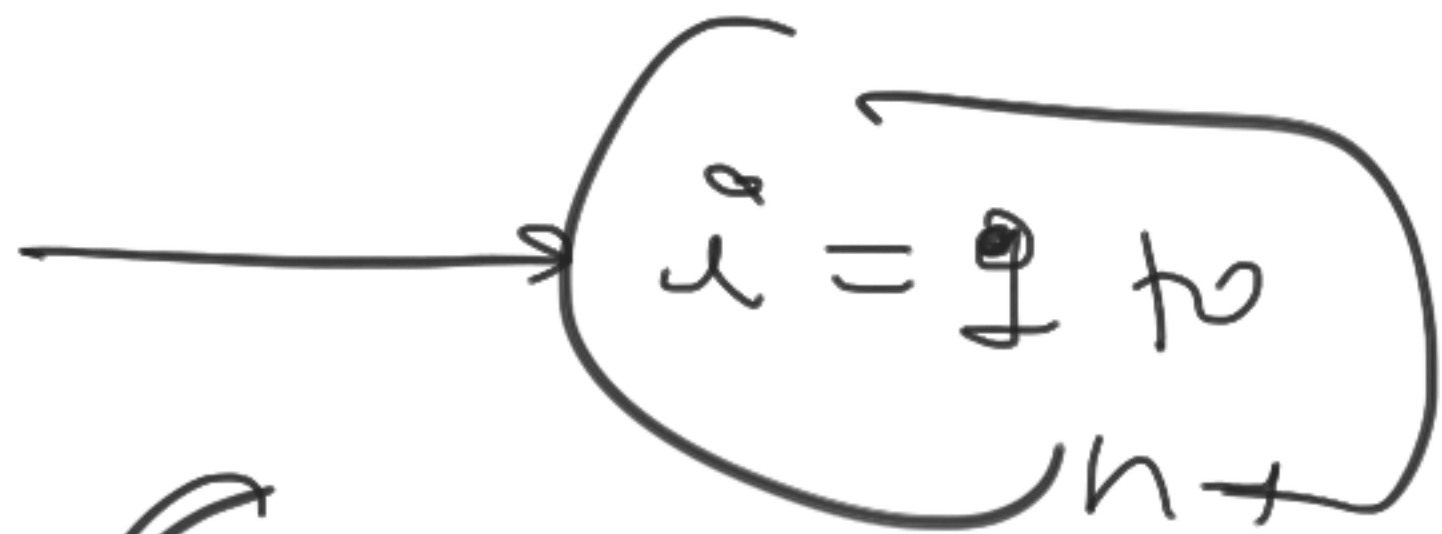
3

10

6



20



$X[j+1]$

$j = j-1$ $X[j]$

for ($i = 1$ to $n-1$)

{ $key = x[i]$

$j = i - 1$

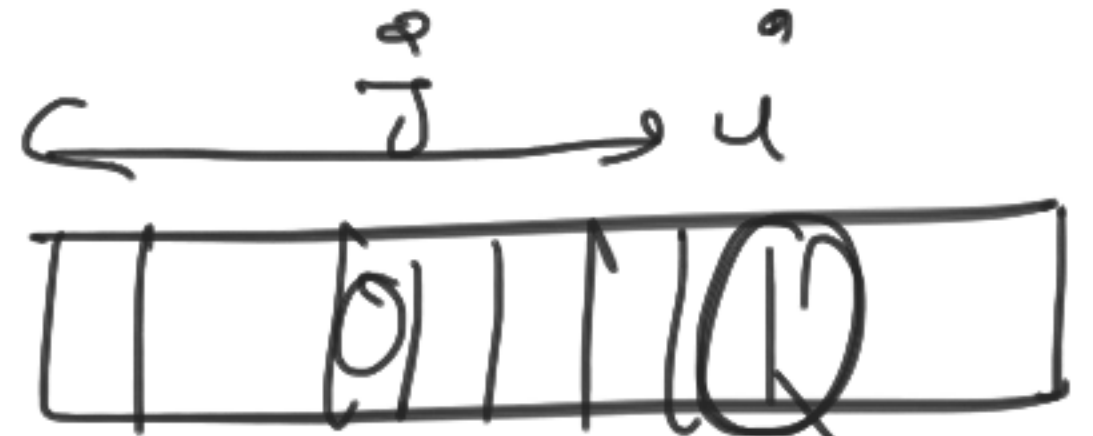
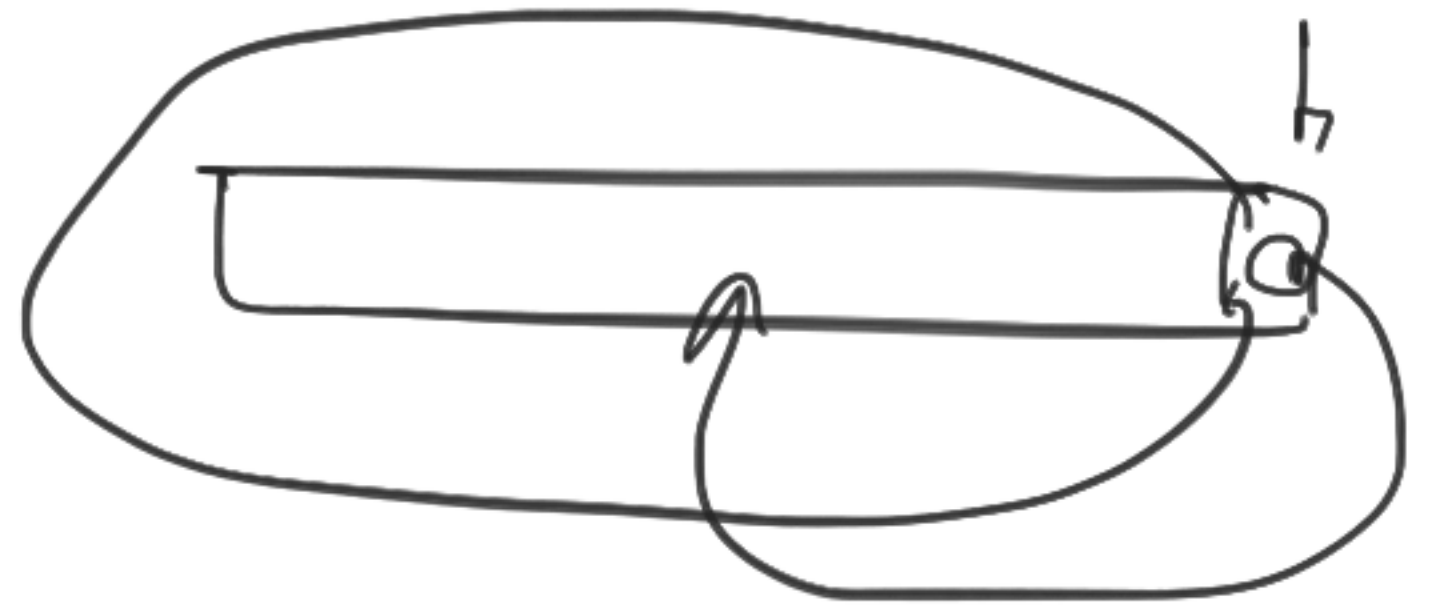
while ($j \geq 0$ &&

{

$x[j+1] = x[j]$

$j = j - 1$

} $x[i+1] = key$



$x[j] > key$

$x[j] < key$

key

Data movement

② Worst Case
Best Case

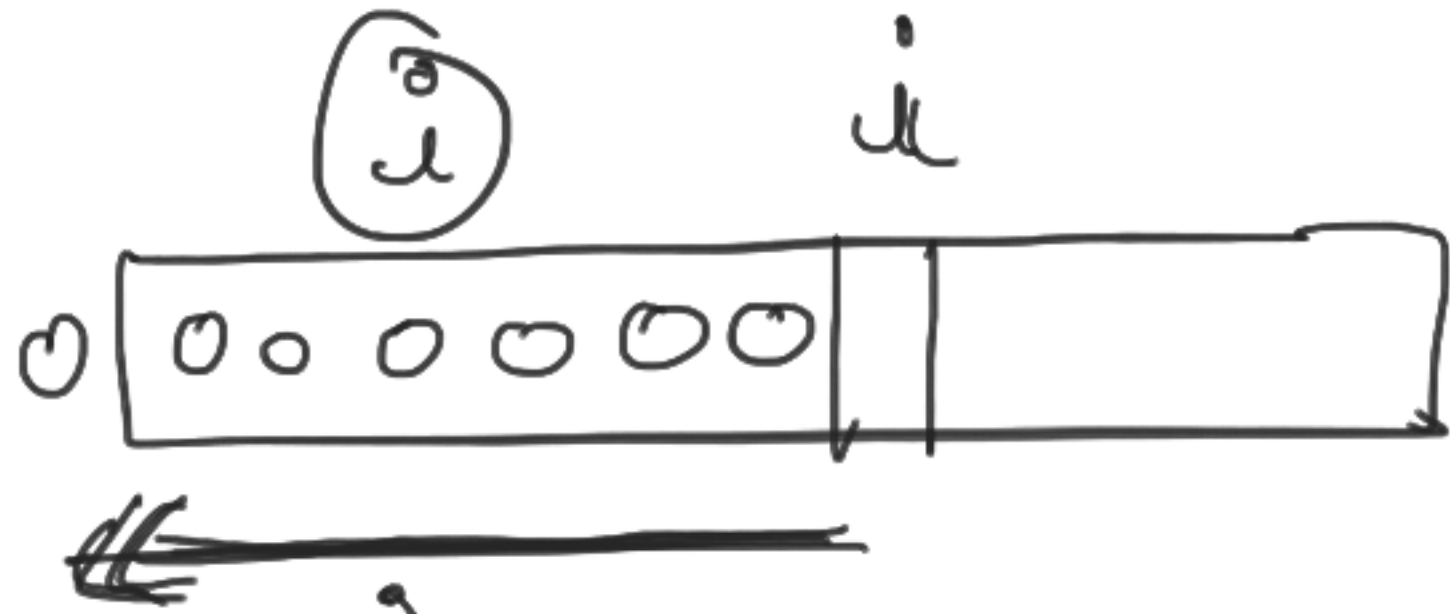
③ When should we use?

④

n inversions.

insertion sort

Comparative
shifting

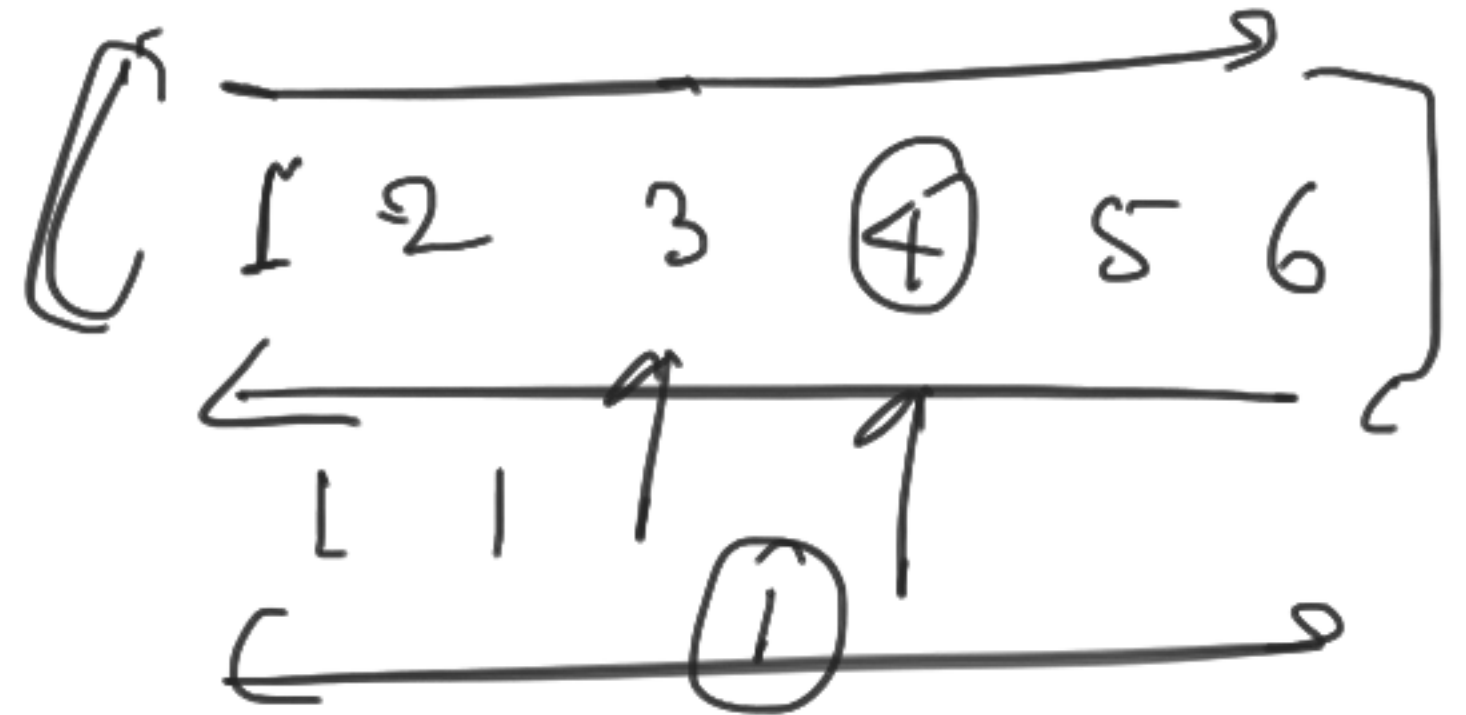
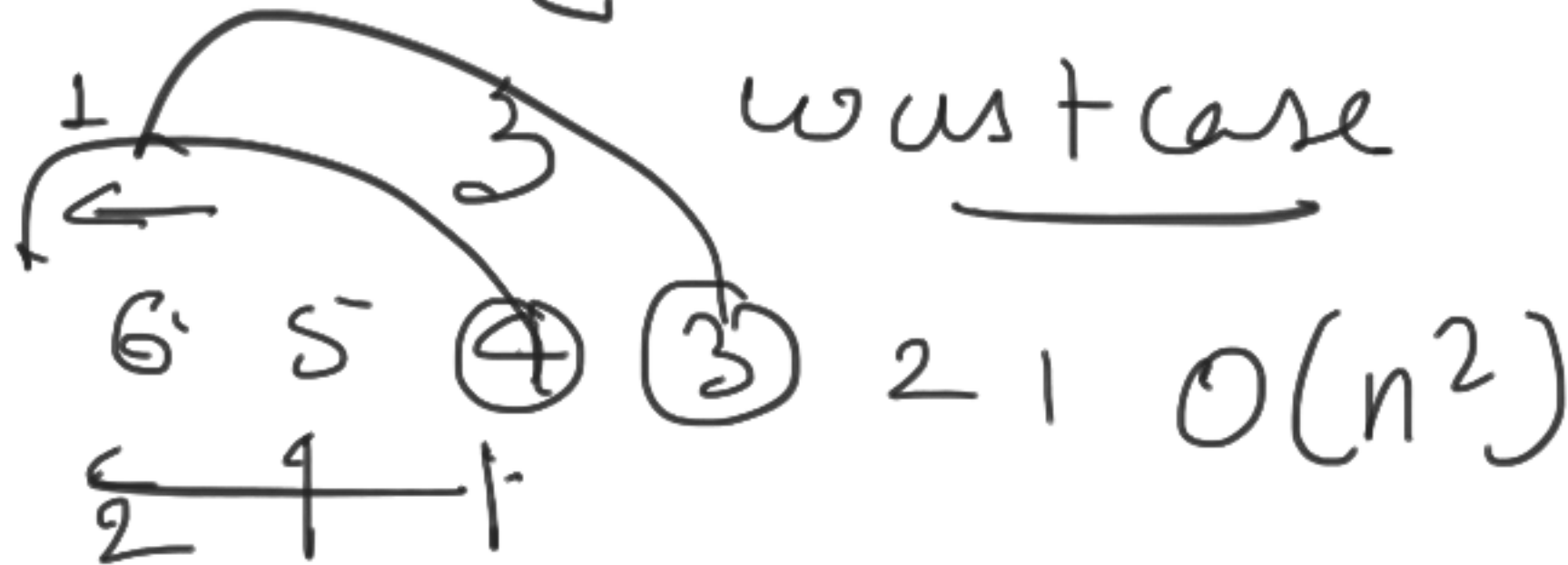


$$\Rightarrow 1 + 2 + \dots + n - 1$$

$$= \frac{n(n-1)}{2} \Rightarrow O(n^2)$$

i comp
 i shifting

worst case

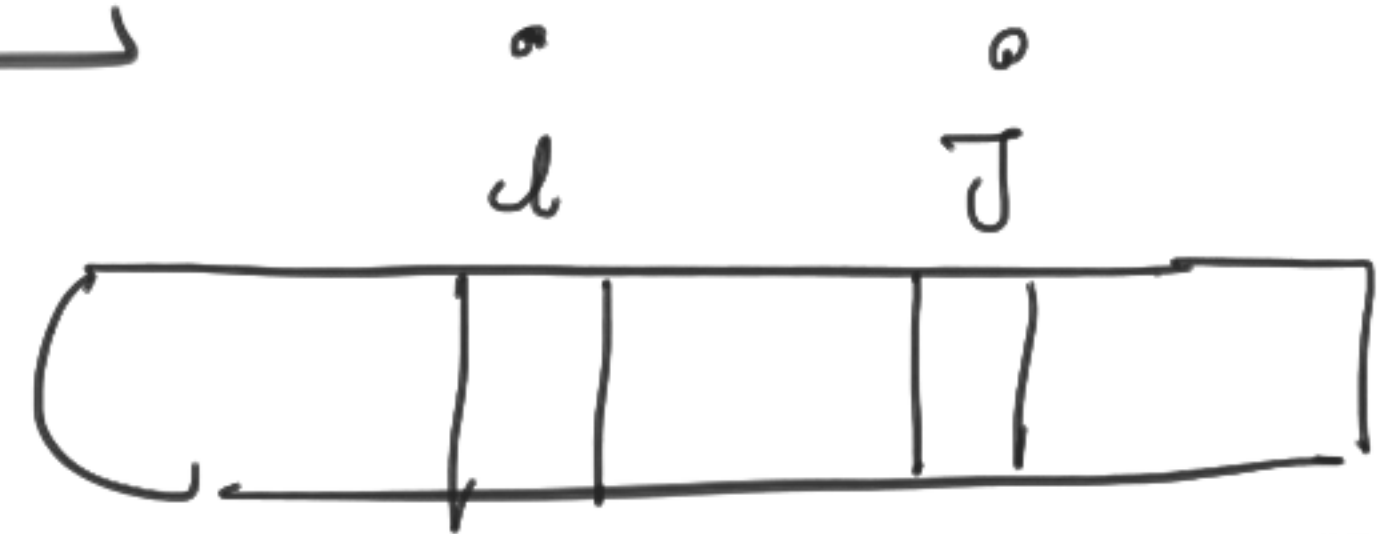


$O(n)$ n comp
 O shift

almost sorted \Rightarrow Partially sorted. $\Rightarrow O(n)$

$$\binom{0}{i, j}$$

inversion



if $A[i] > A[j]$

$i < j$

one inversion



(3) (1) (4)

(3, 2) (3, 1)
(4, 1) (2, 1)