$\alpha = n/m \Rightarrow O(\alpha) \Rightarrow$ 

$m = n/2 \quad \alpha = 2 \Rightarrow 2$

$m = n/4 \quad \alpha = 4 \Rightarrow 4$

$m = n/8 \quad \alpha = 8 \Rightarrow 8$

$n \rightarrow m \Rightarrow$

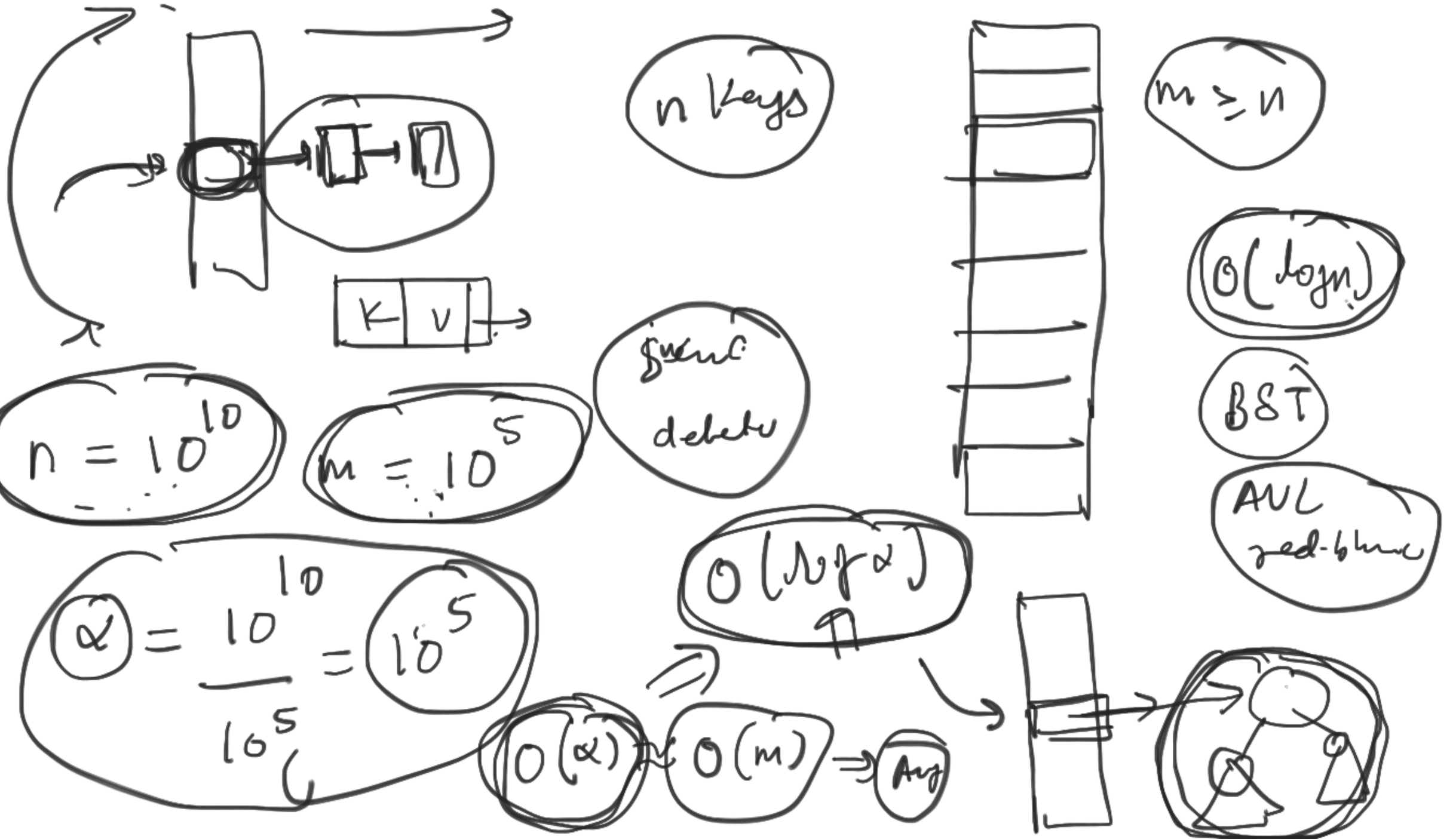
Time memory tradeoff

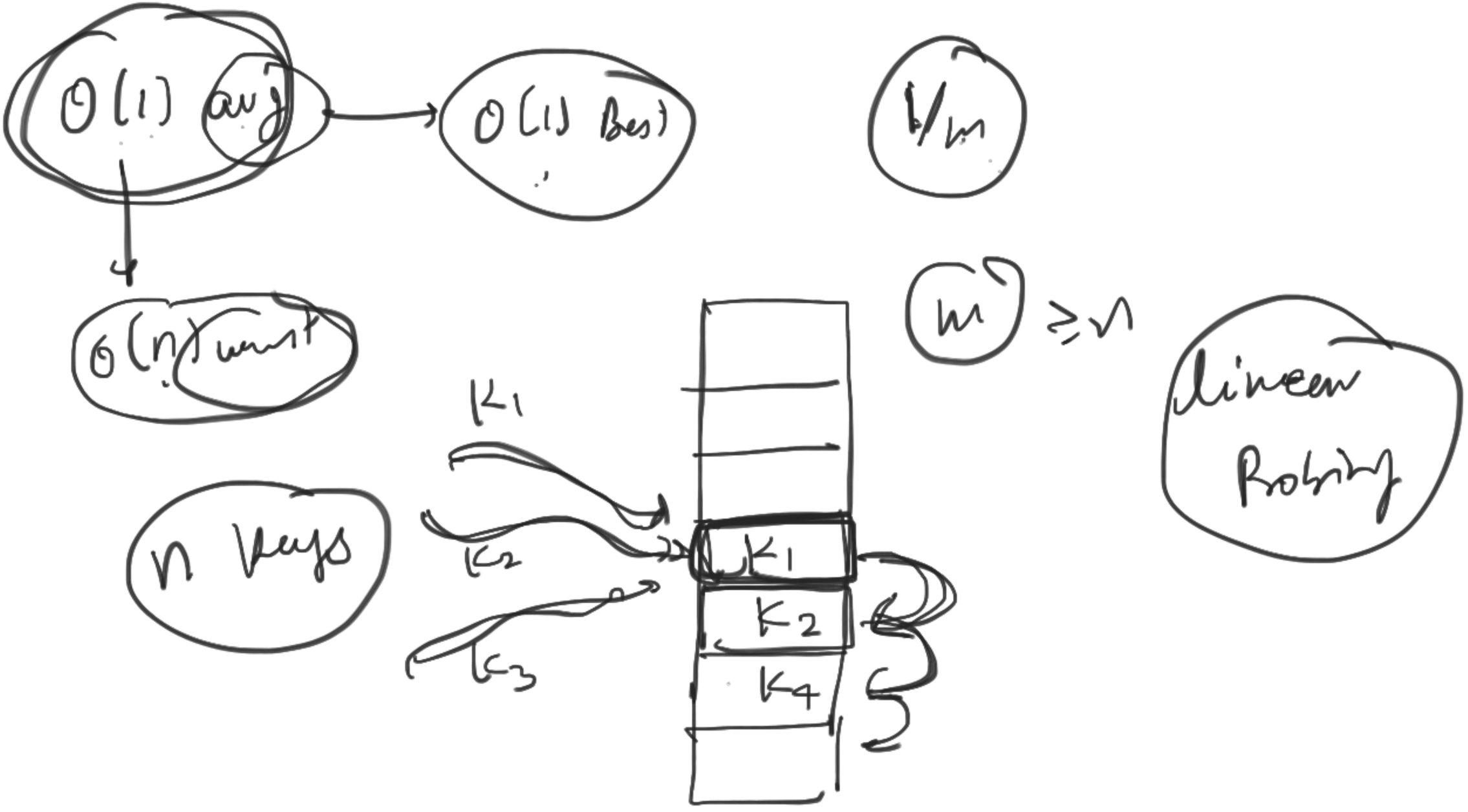
$m \geq n$

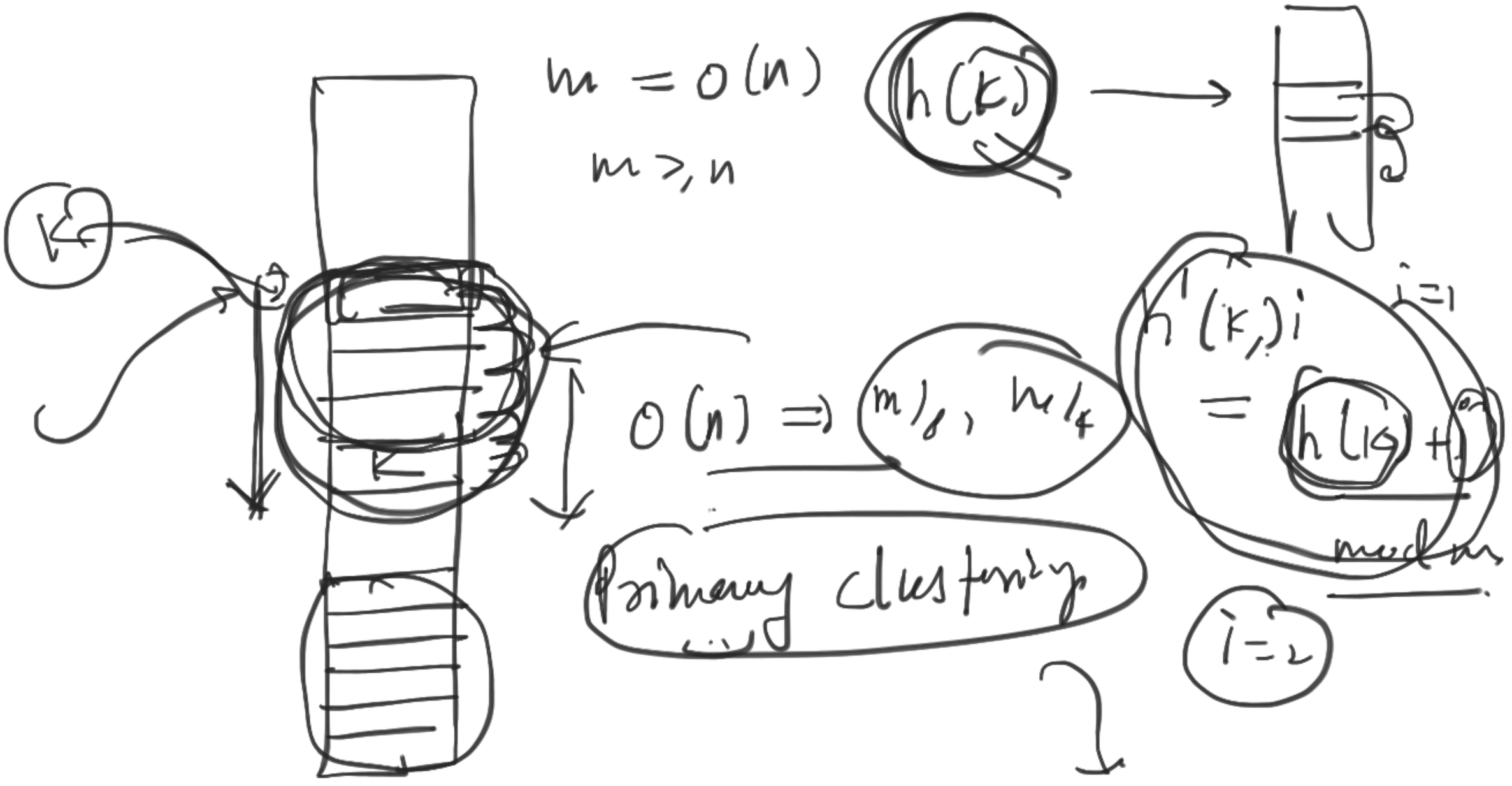
List

Array

open addressing


n keys
m ≥ n
K V
$n = 10^{10}$
$m = 10^5$
func
delete
O(log n)
BST
AVL
red-black
$\alpha = \frac{10^{10}}{10^5} = 10^5$
O(log α)
O(α)
O(m)
Avg

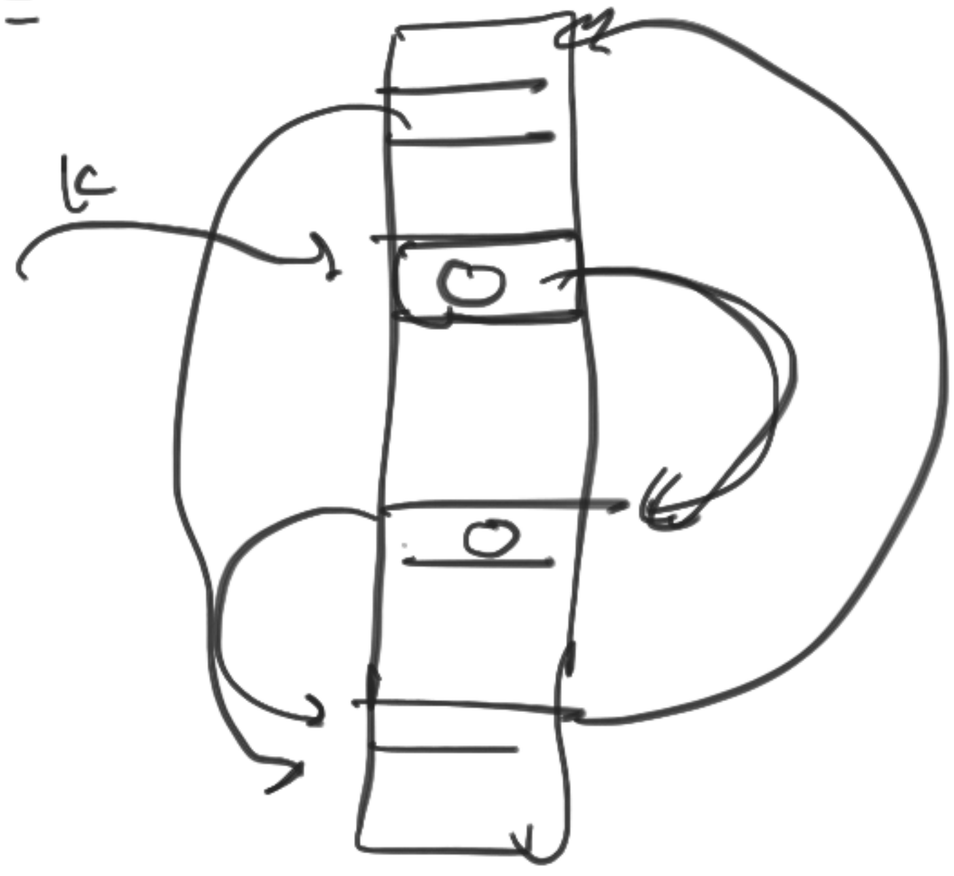

O(1) avg
O(1) Best
1/m
O(n) worst
m ≥ n
n keys
Linear Probing
K1
K2
K3
K1
K2
K4


m = O(n)
m >, n
h(k)
k
O(n) ⇒
m/8, m/4
Primary clustering
h'(k,)i
i=1
=
h(k)
i=2

$$h'(k,i) = (h(k) + i^2) \bmod m$$

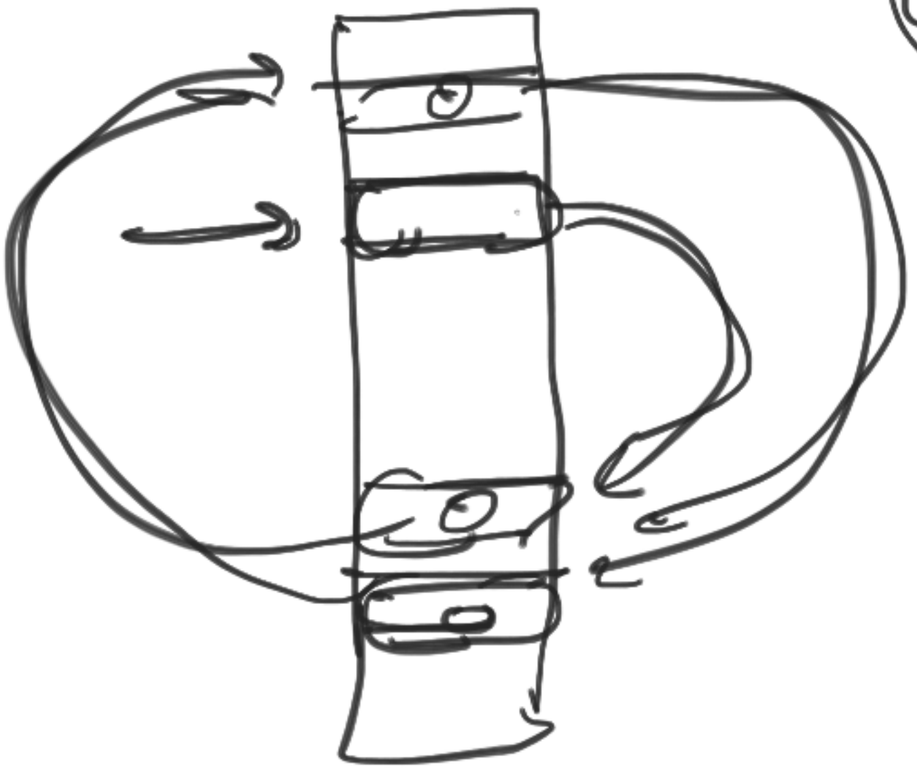
quadratic hashing

$i =$



$$h'(k,i) = h(k) + h_0(k,i) \bmod m$$

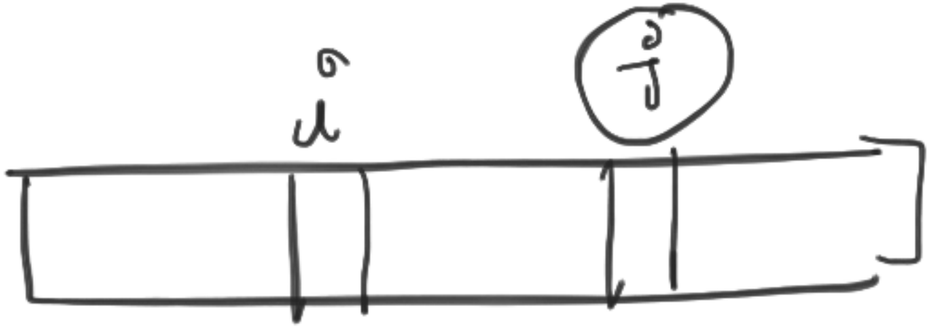
double hashing



O(1) avg

5 4 3 2 1

max difference array



min

max

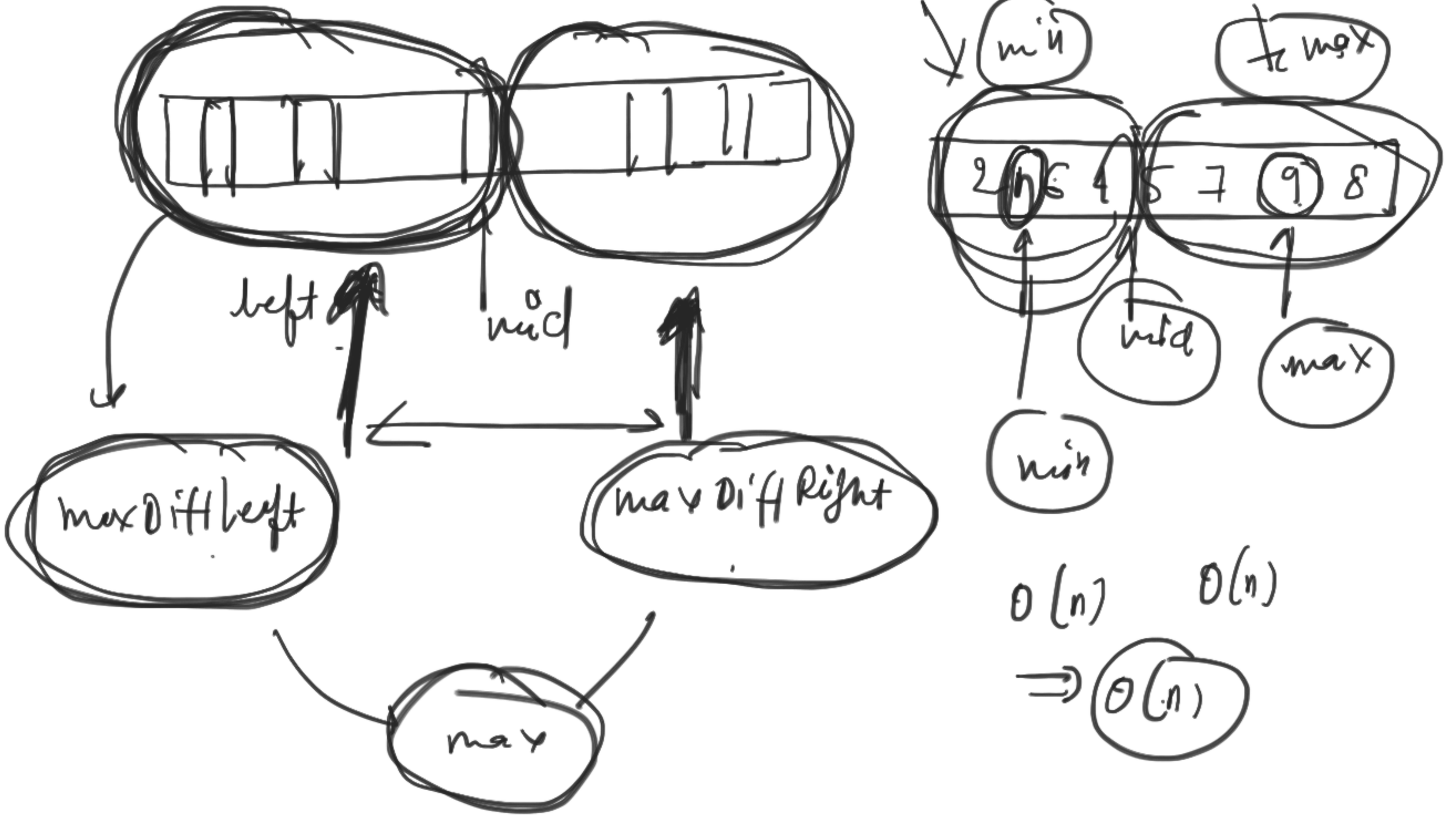
recursion

$j \geq i$

$A[j] - A[i]$

max

$A[j] > A[i]$

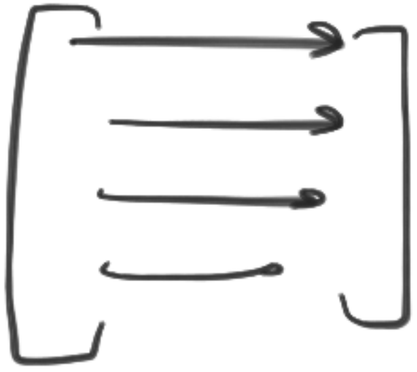

left
mid
maxDiffLeft
maxDiffRight
max
min
max
2 6 5 4 5 7 9 8
mid
max
min
O(n) O(n)
O(n)

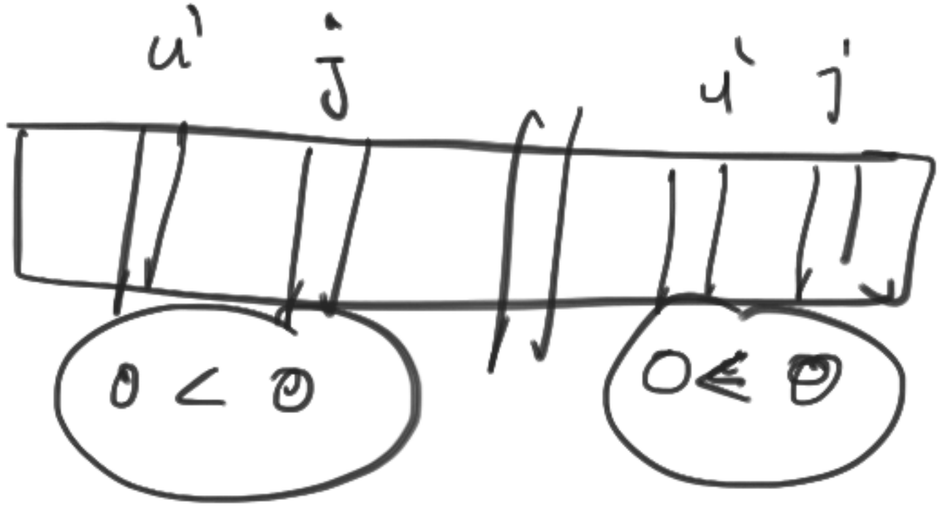
$n \longrightarrow 2\left(n/2\right) + O(n) + O(1)$

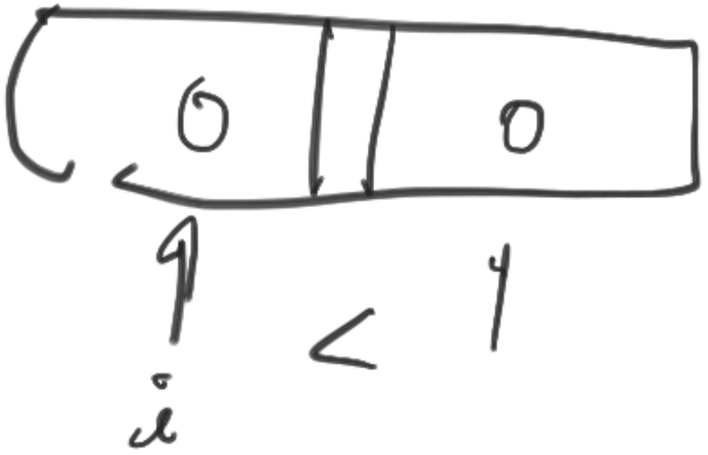
$$T(n) = 2T\left(n/2\right) + O(n)$$

merge sort

$$T(n) = O(n \log n)$$

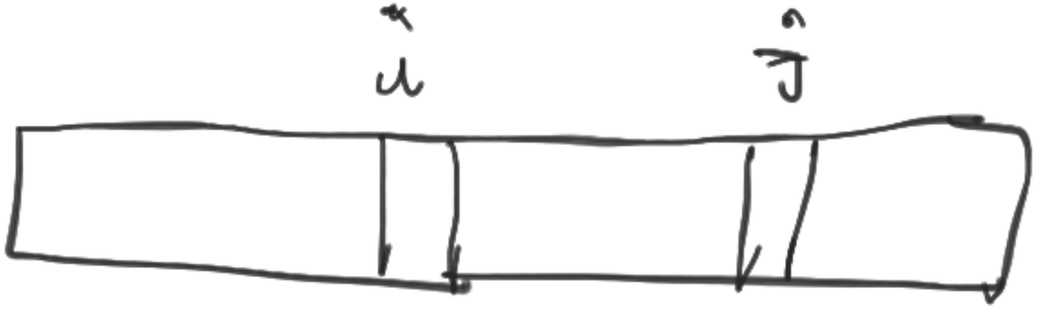




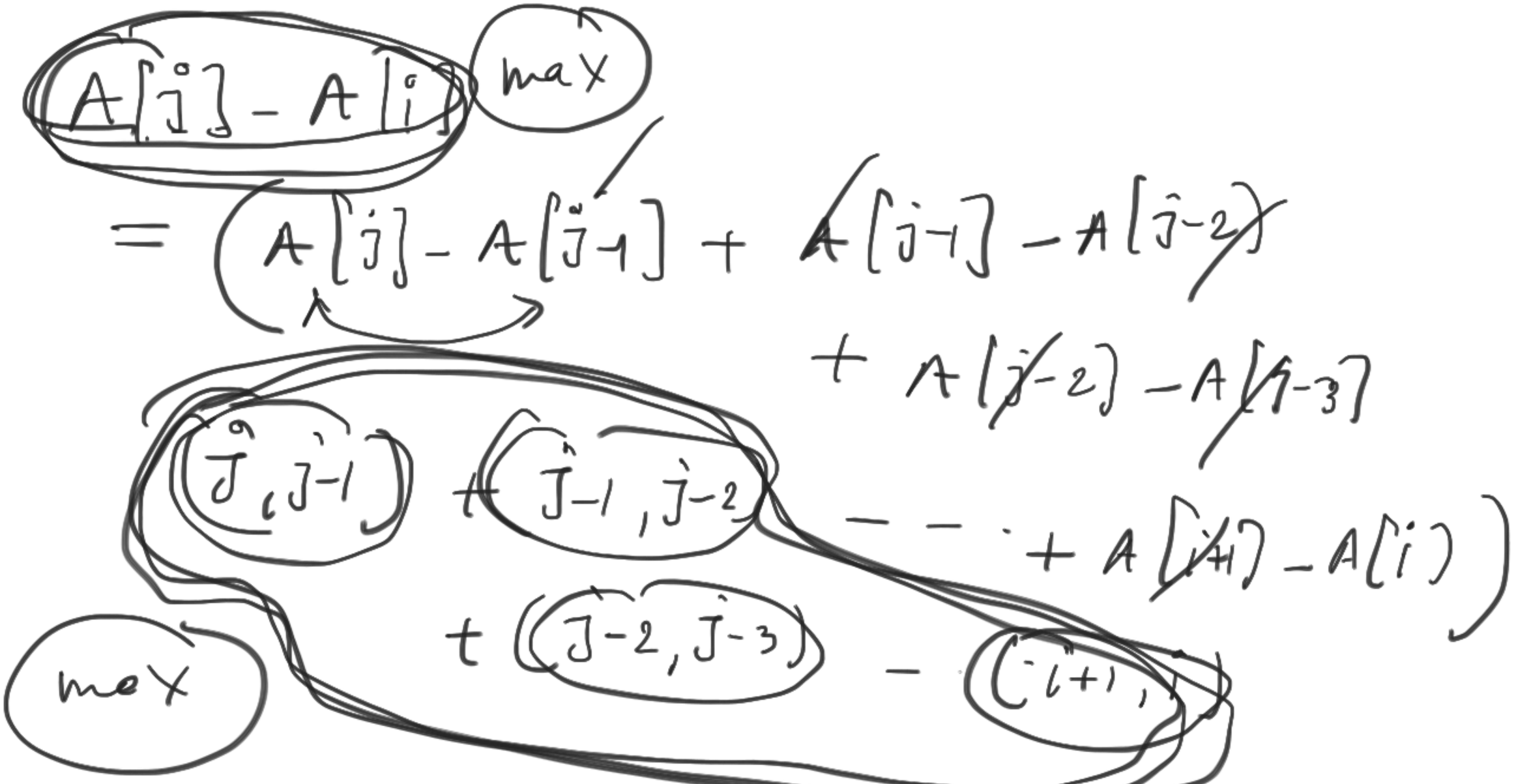


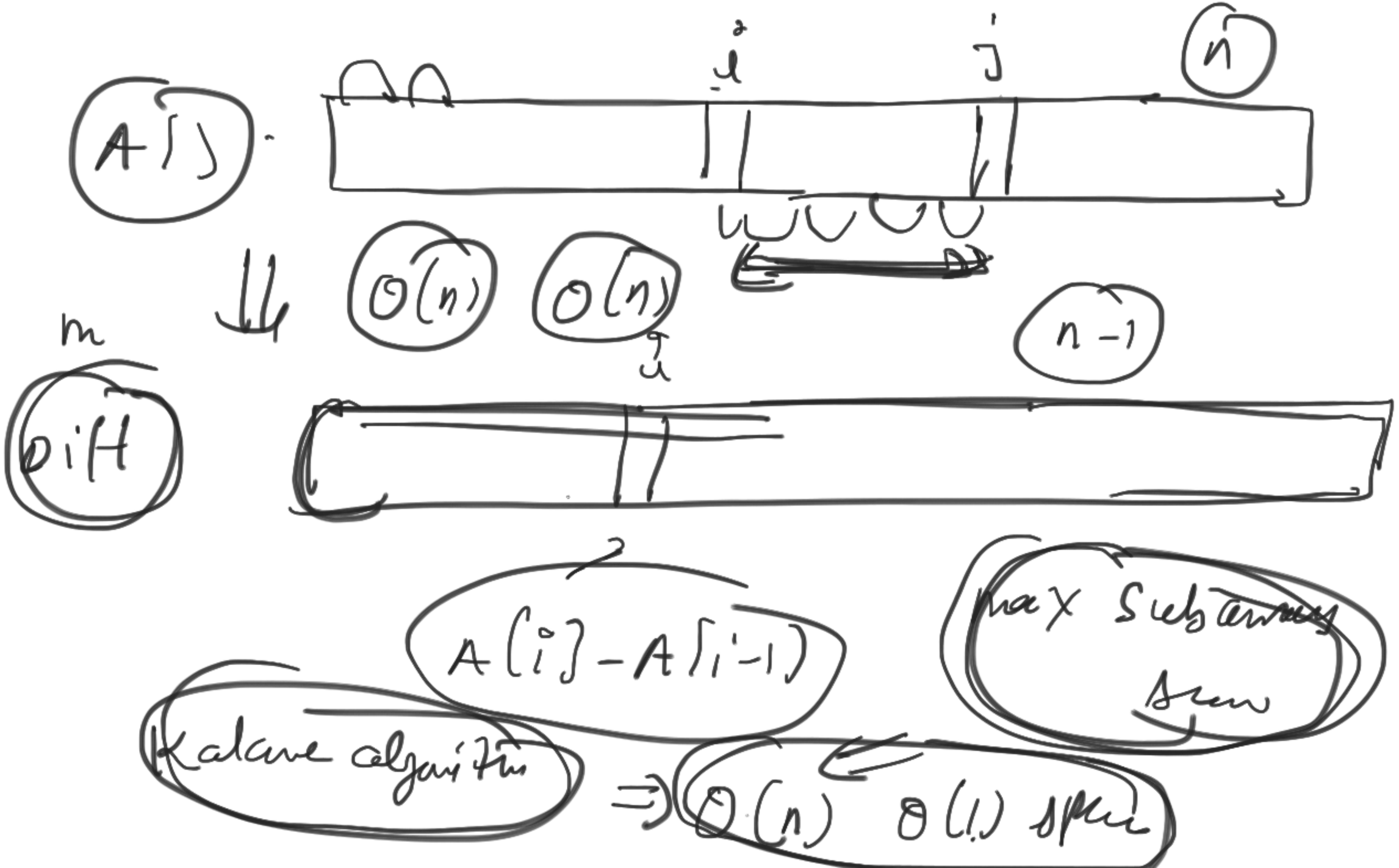
$i < j$

$A[i] < A[j]$



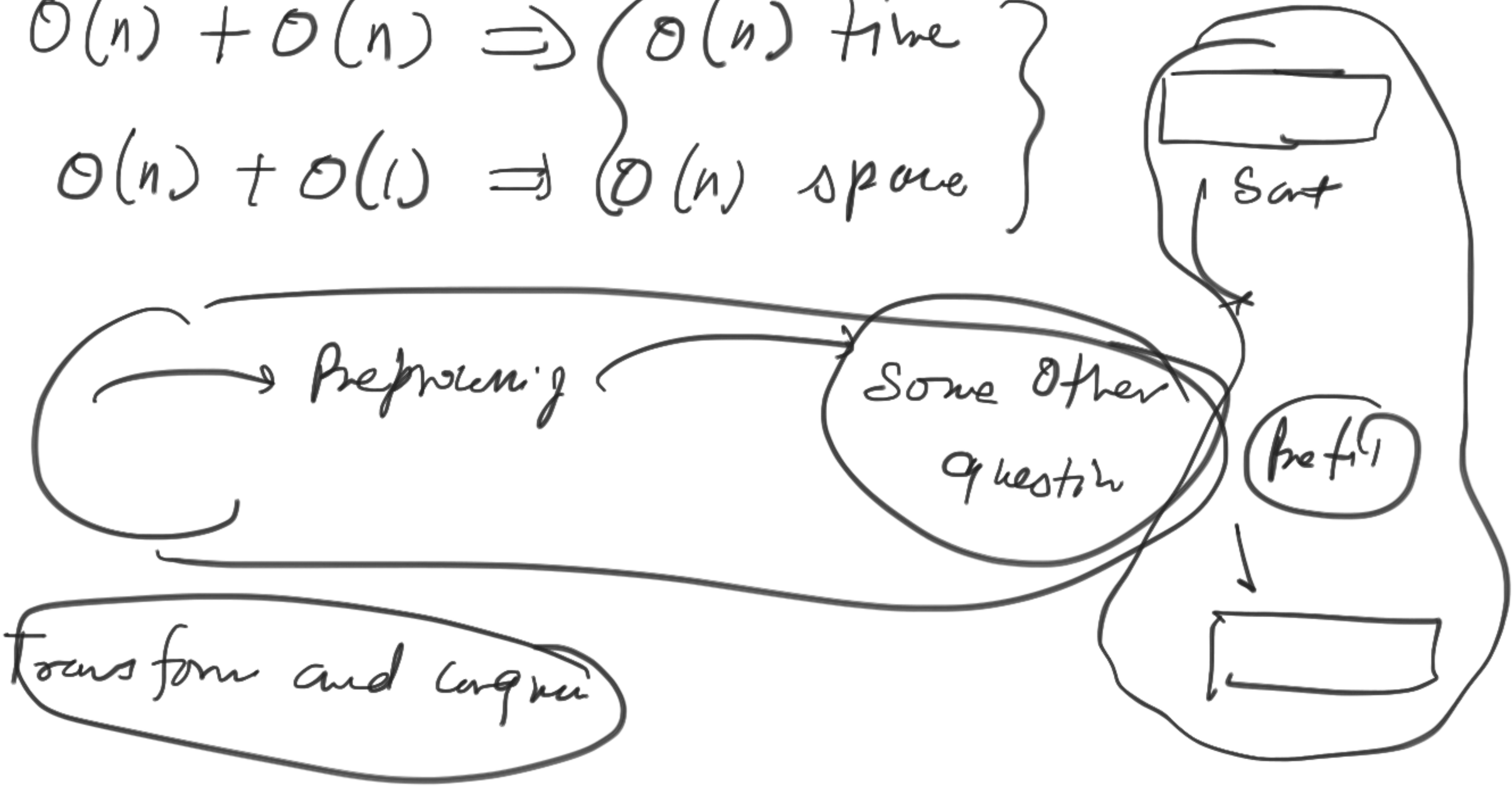
$A[j] - A[i] =$

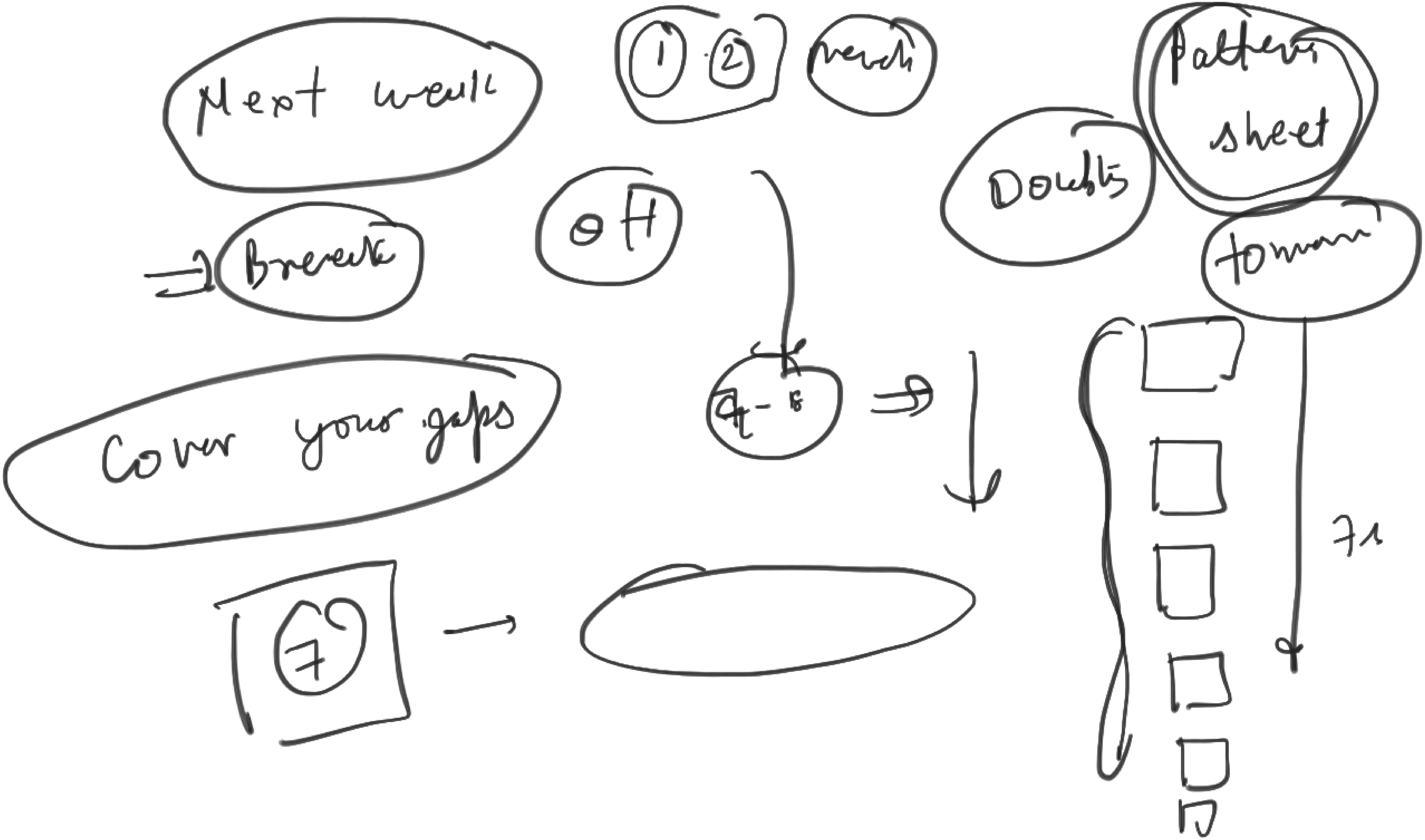

A[j] - A[i]
max
= A[j] - A[j-1] + A[j-1] - A[j-2]
+ A[j-2] - A[j-3]
(j, j-1) + (j-1, j-2) - - - + A[i+1] - A[i])
+ (j-2, j-3) - (i+1, i)
max


A[]
i
j
n
O(n)
O(n)
m
n-1
diff
A[i]-A[i-1]
max Subarray Sum
Kadane algorithm
O(n) O(1) space

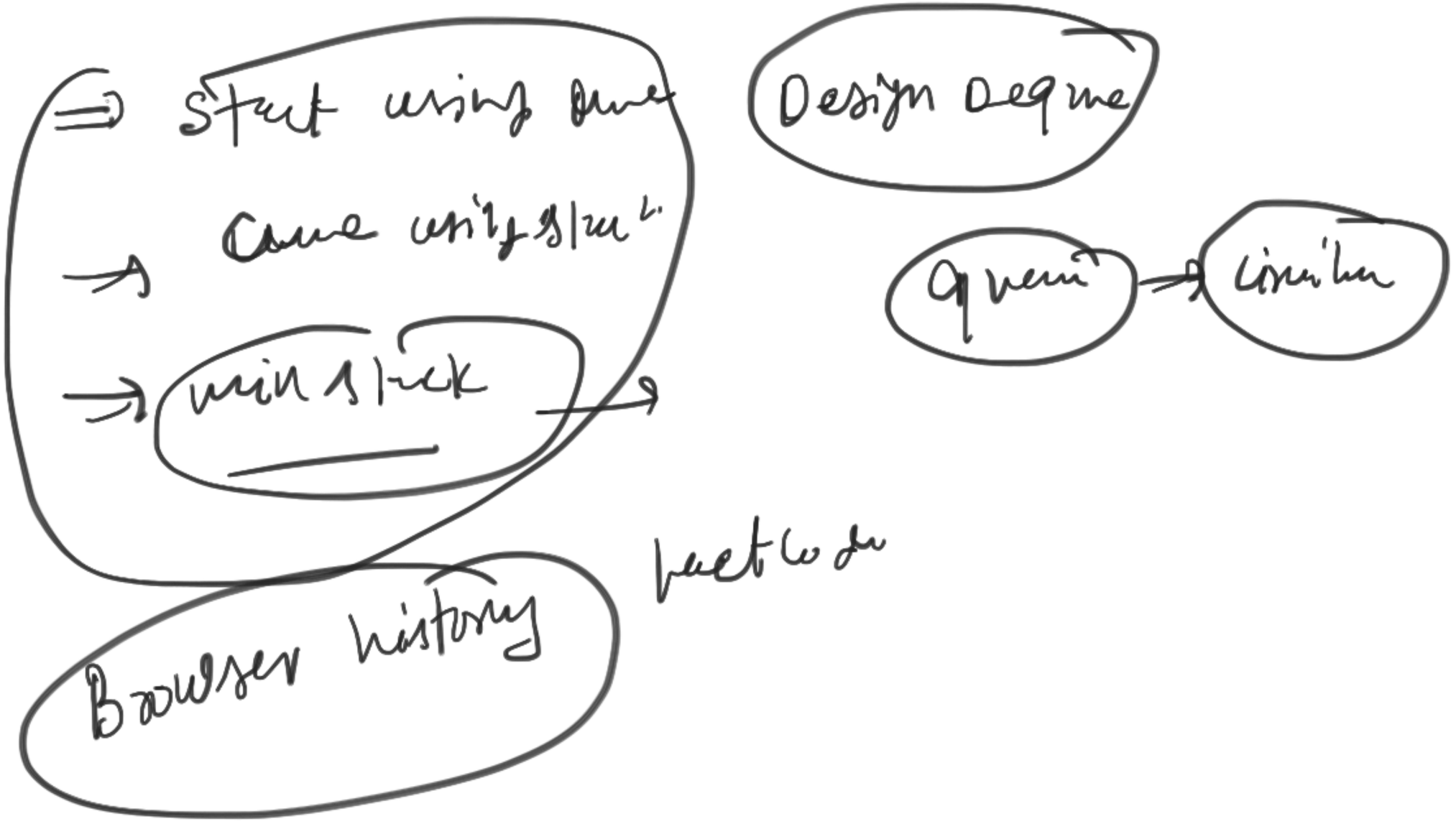
$O(n) + O(n) \Rightarrow$ $O(n)$ time

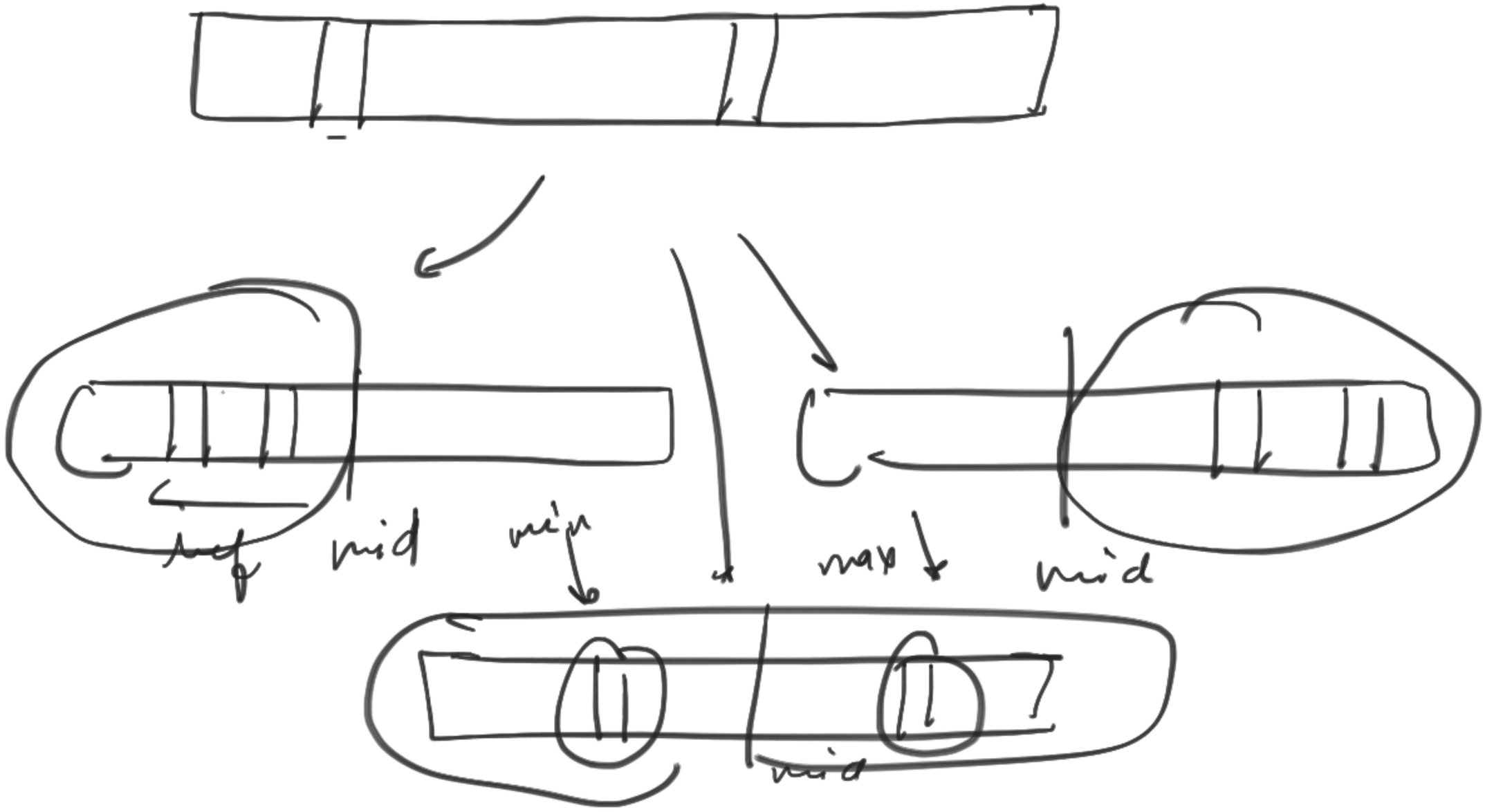
$O(n) + O(1) \Rightarrow$ $O(n)$ space





Next week
Break
Cover your gaps
7
Doubts
Pattern sheet


Start using Stack
Design Deque
Queue using Stack
Queue
Circular
min Stack
Browser history
Leetcode


mid
min
max
mid
mid