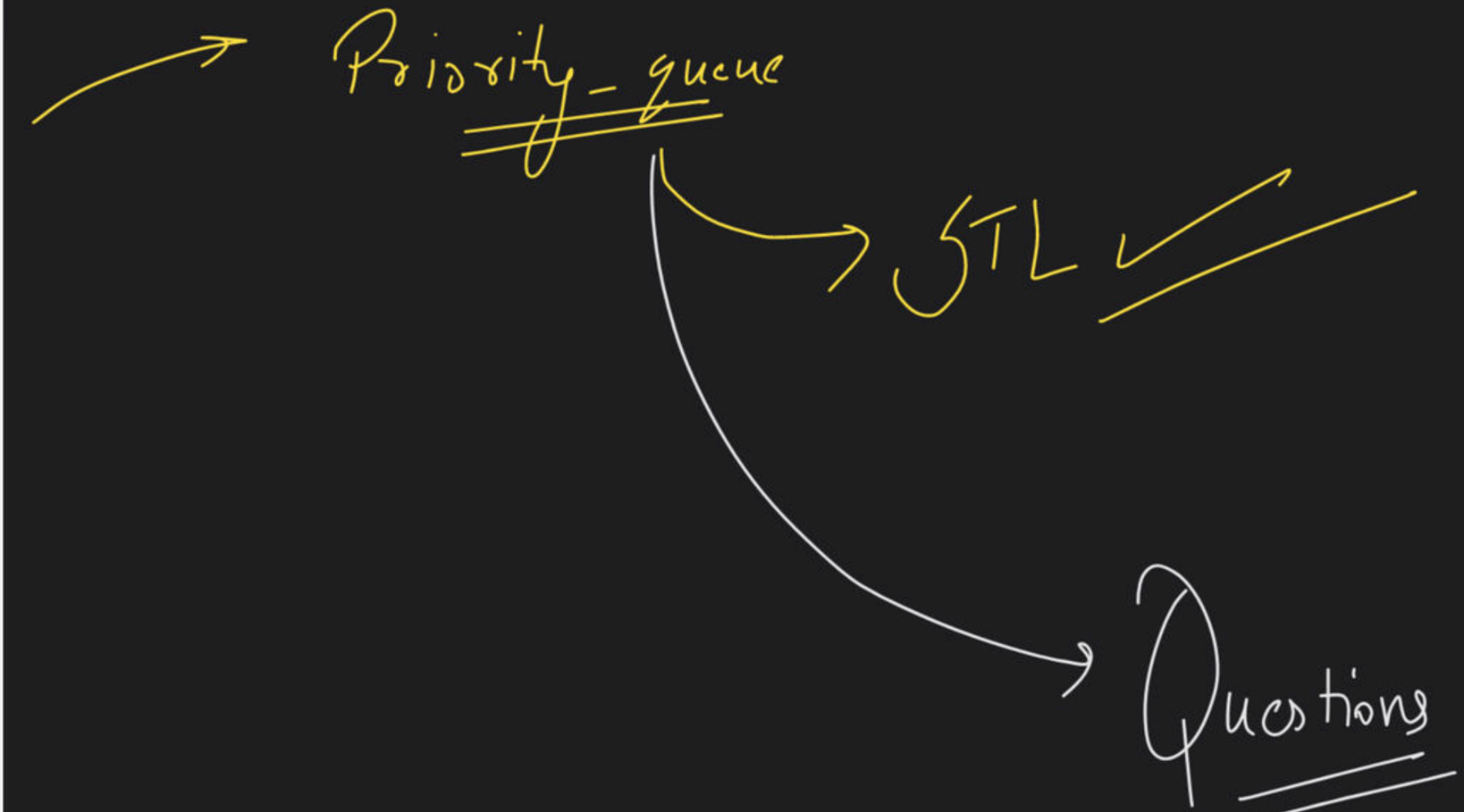
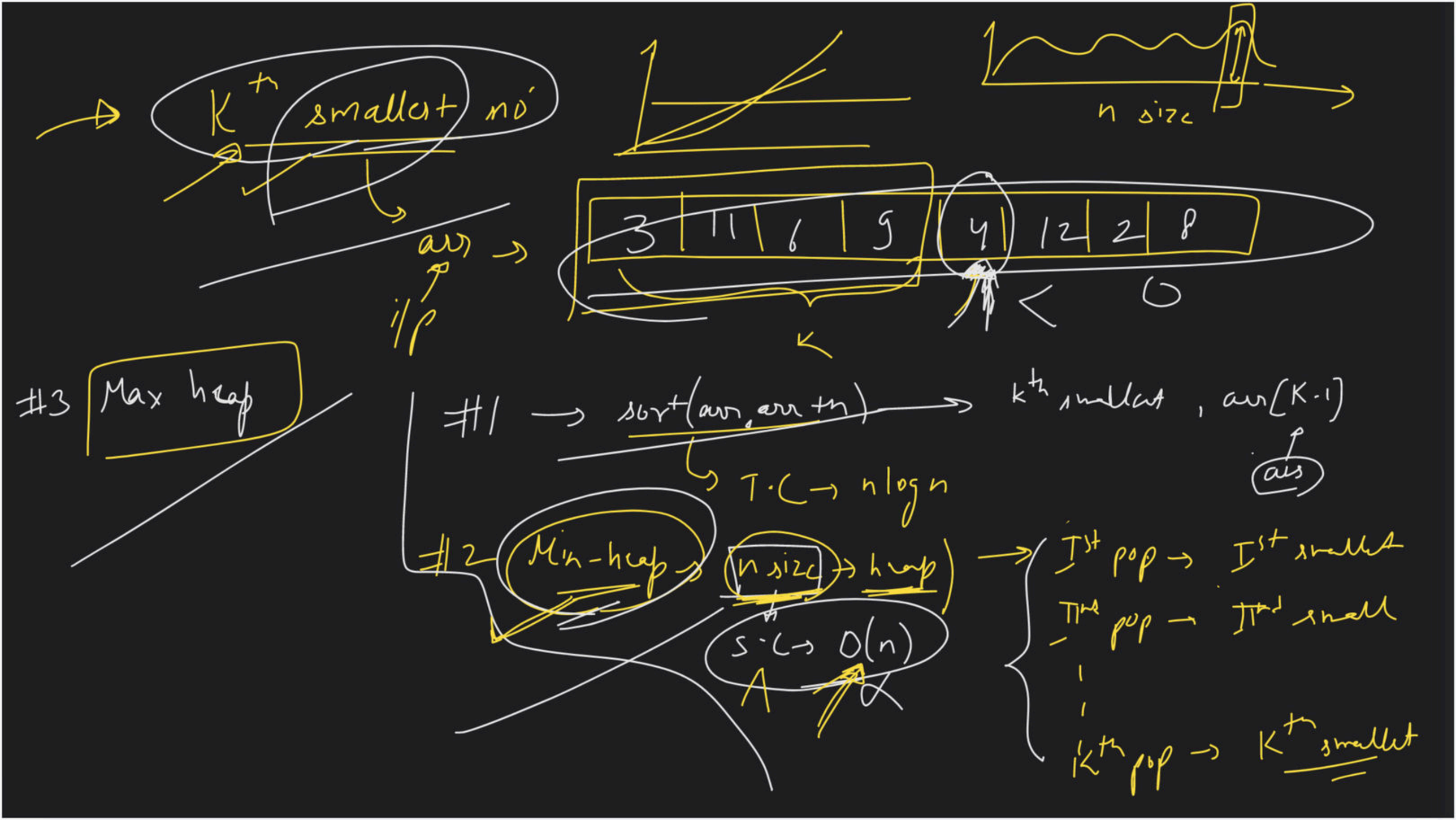
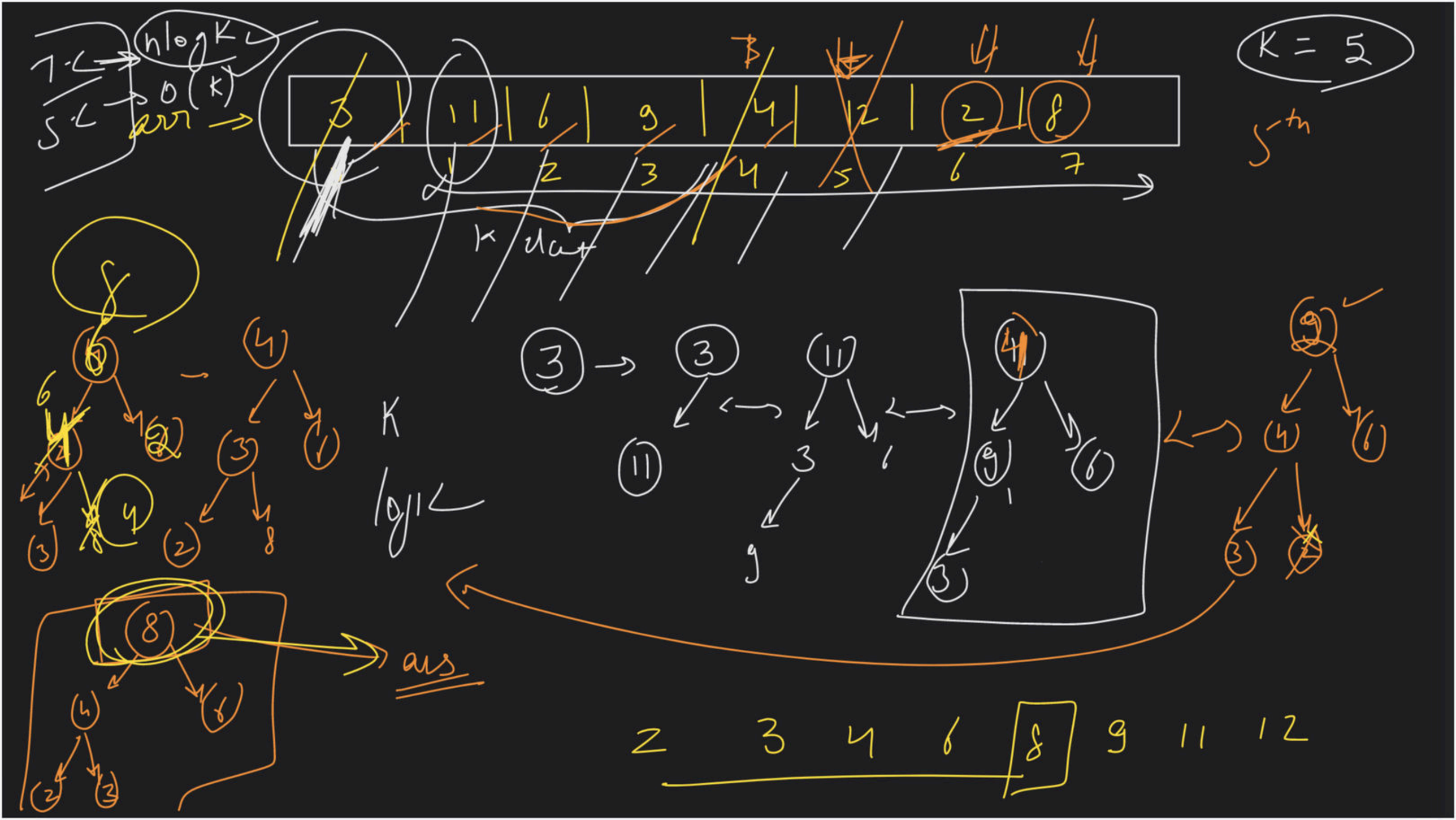
Heaps Class - 2

Special class

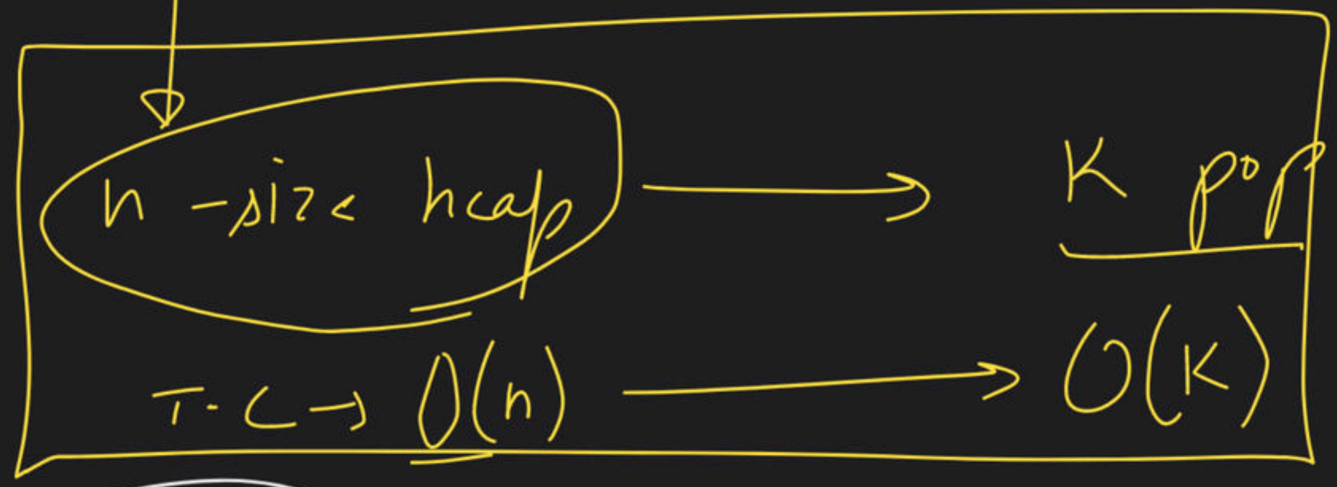
Love Babbar • May 24, 2023

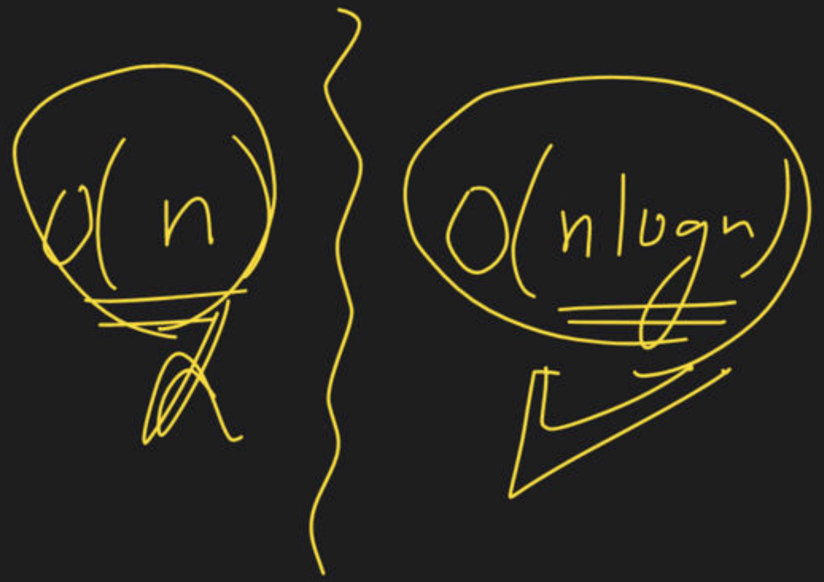

Priority_queue
STL
Questions


K th smallest no
n size
arr
i/p
3
11
6
9
41
12
2
8
0
#3 Max heap
#1 → sort(arr, arr+n) → kth smallest , arr[K-1]
arr
T.C → nlogn
#2 Min-heap → (n size → heap)
S.C → O(n)
A
1st pop → 1st smallest
IInd pop → IInd small
Kth pop → Kth smallest


nlogK
O(K)
K = 5
K th
logic
aus
2 3 4 6 8 9 11 12

#2 Min heap





#3 Max-heap

first K elemt -> max heap

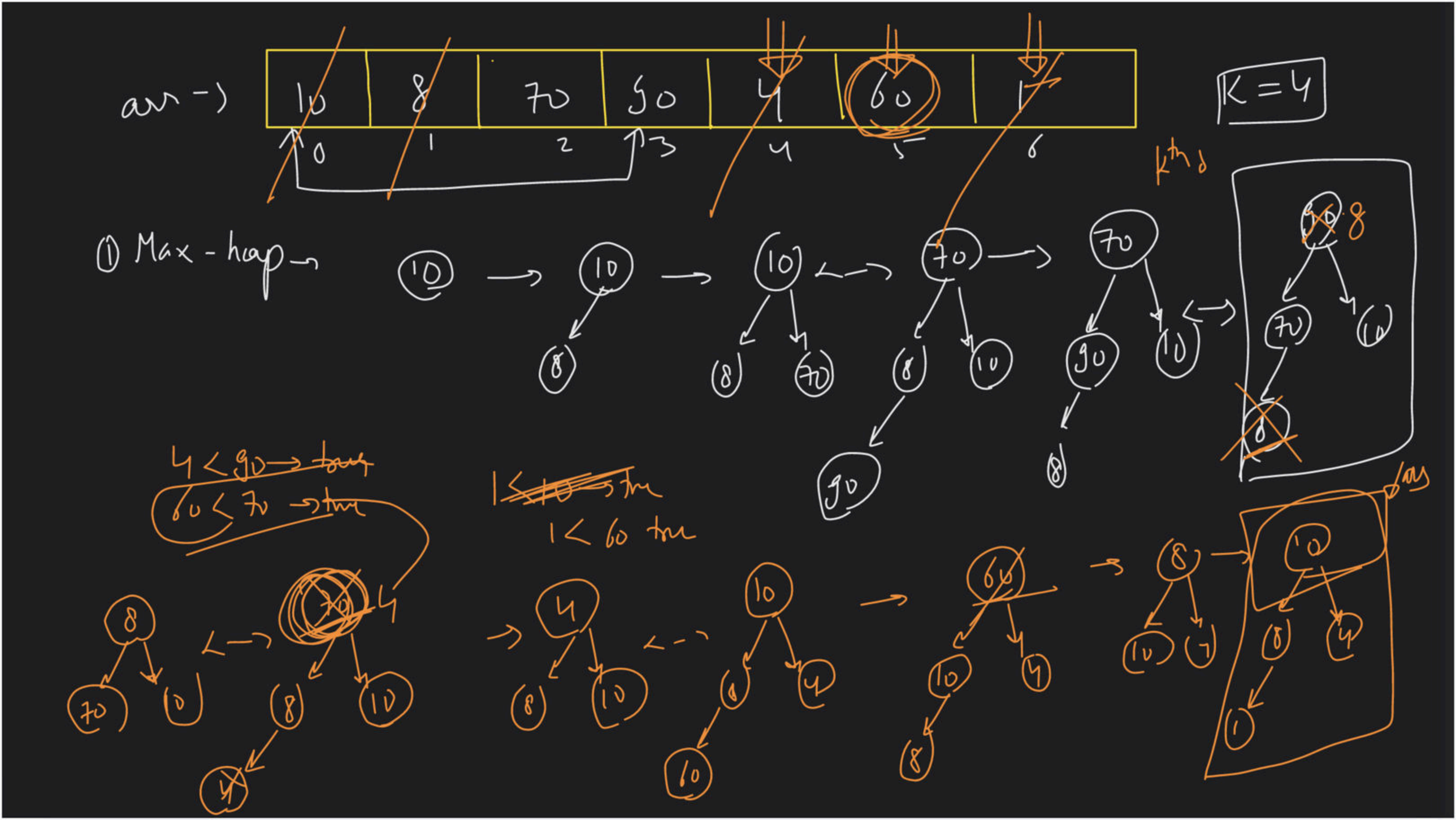
k size -> O(K)

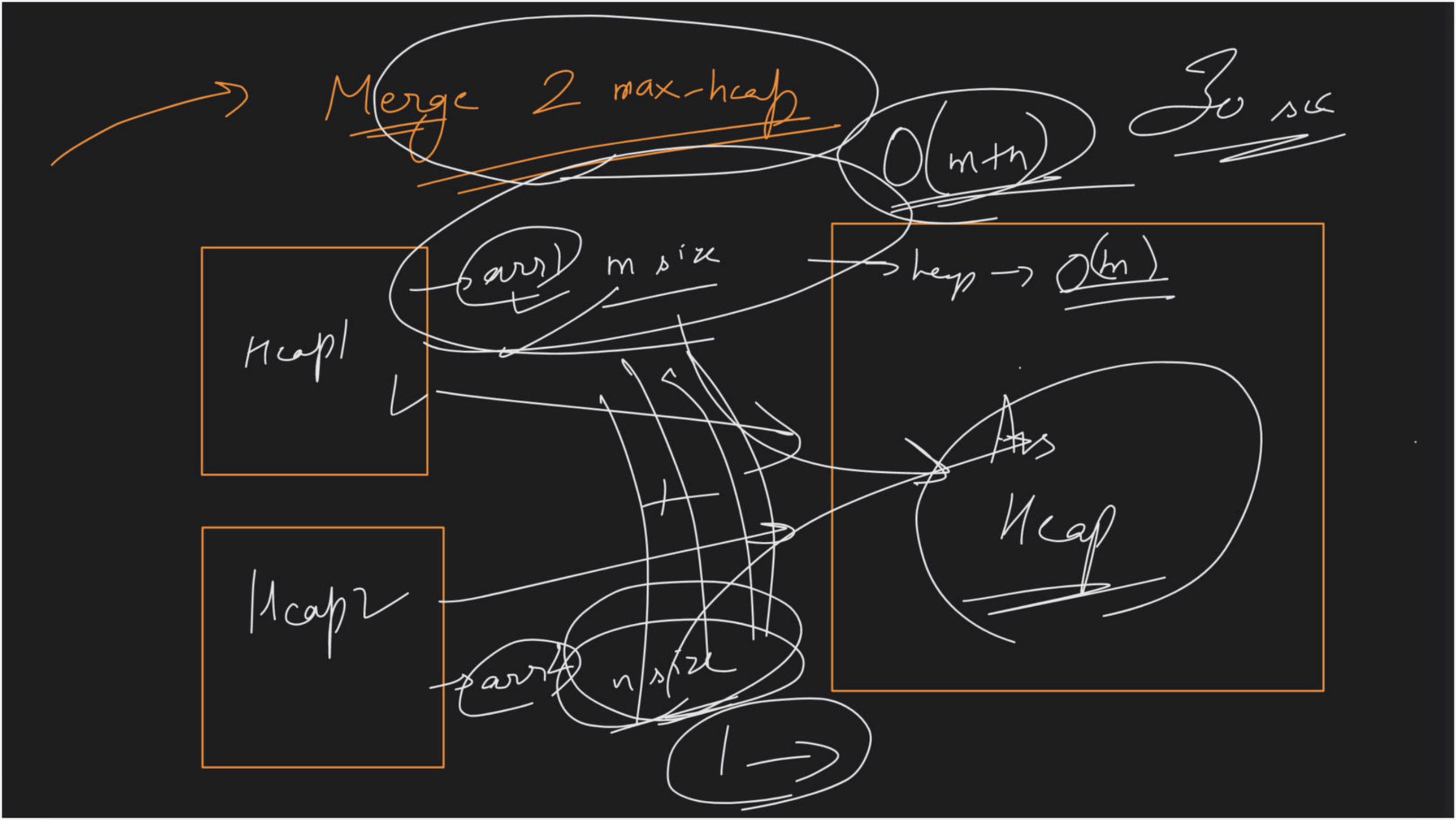
Max heap
↳
~~K~~ K -size -> K small elemts -> top -> K^th smallest elemt

insert -> newElemt < heap.top()

heap.pop()

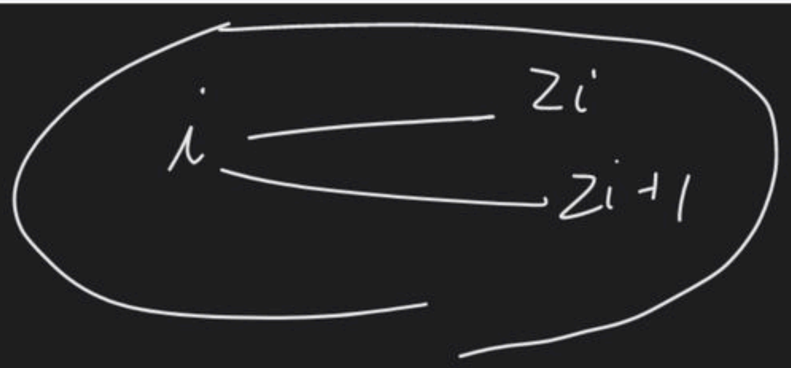
heap.insert(newElem)

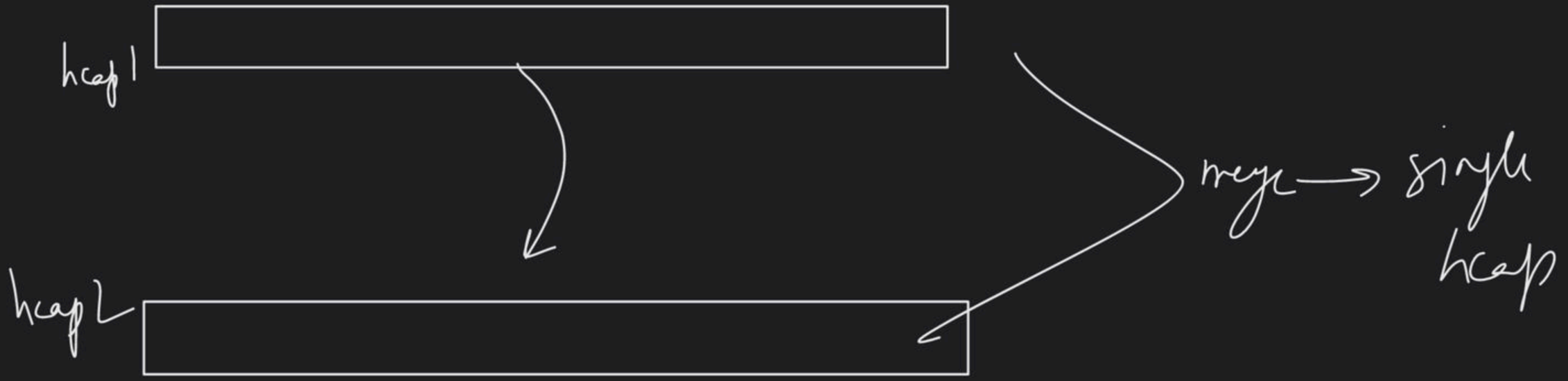

arr -)
10
8
70
90
4
60
1
0
1
2
3
4
5
6
K=4
① Max - heap →
kth
4 < 90 → true
60 < 70 → true
1 < 60 true
1 < 10 true


Merge 2 max-heap
O(m+n)
So sic
arr1
m size
heap → O(m)
Heap1
Ans
Heap
Heap2
arr2
n size
1

H/W

Merge 2 max heap

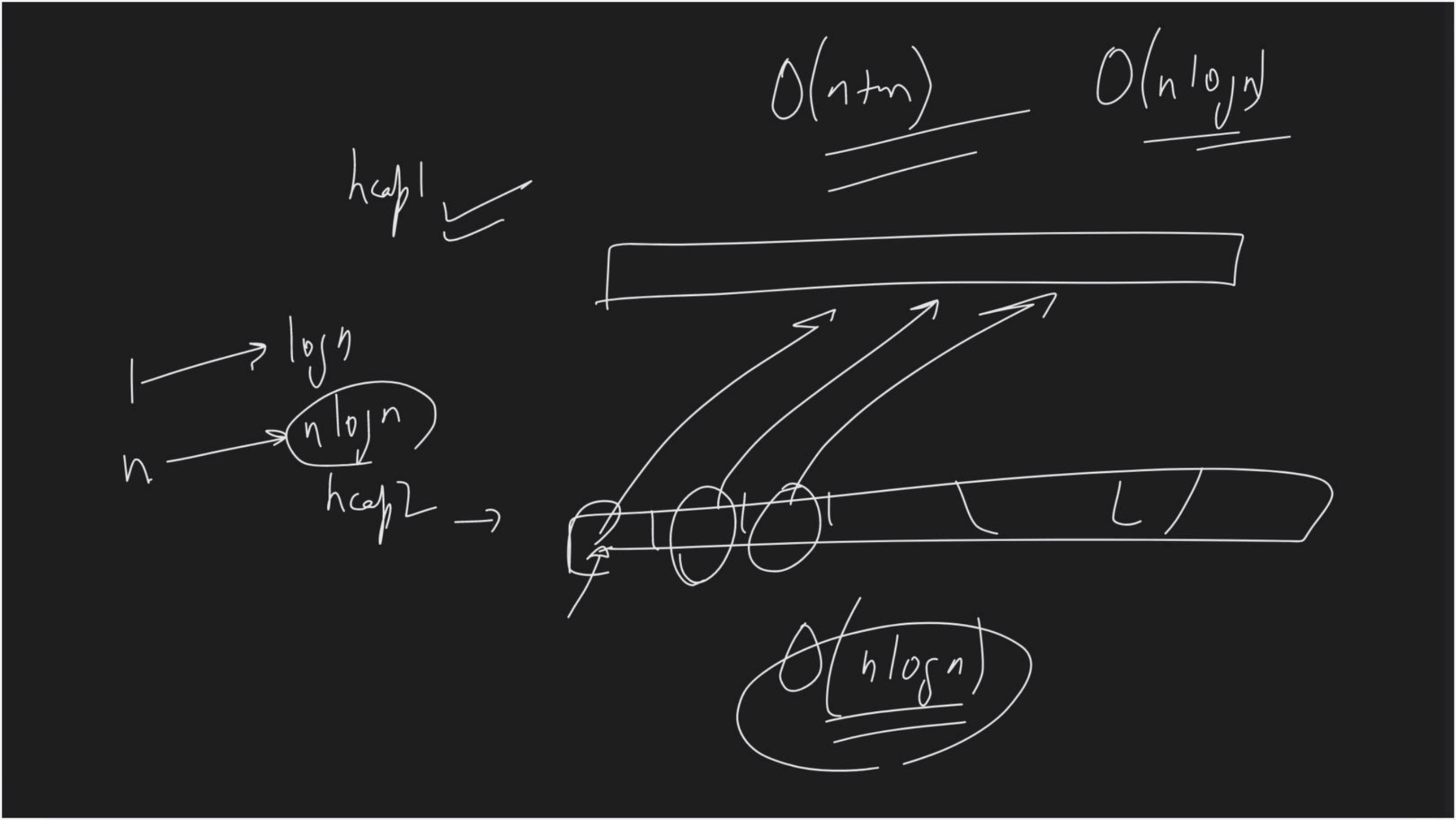


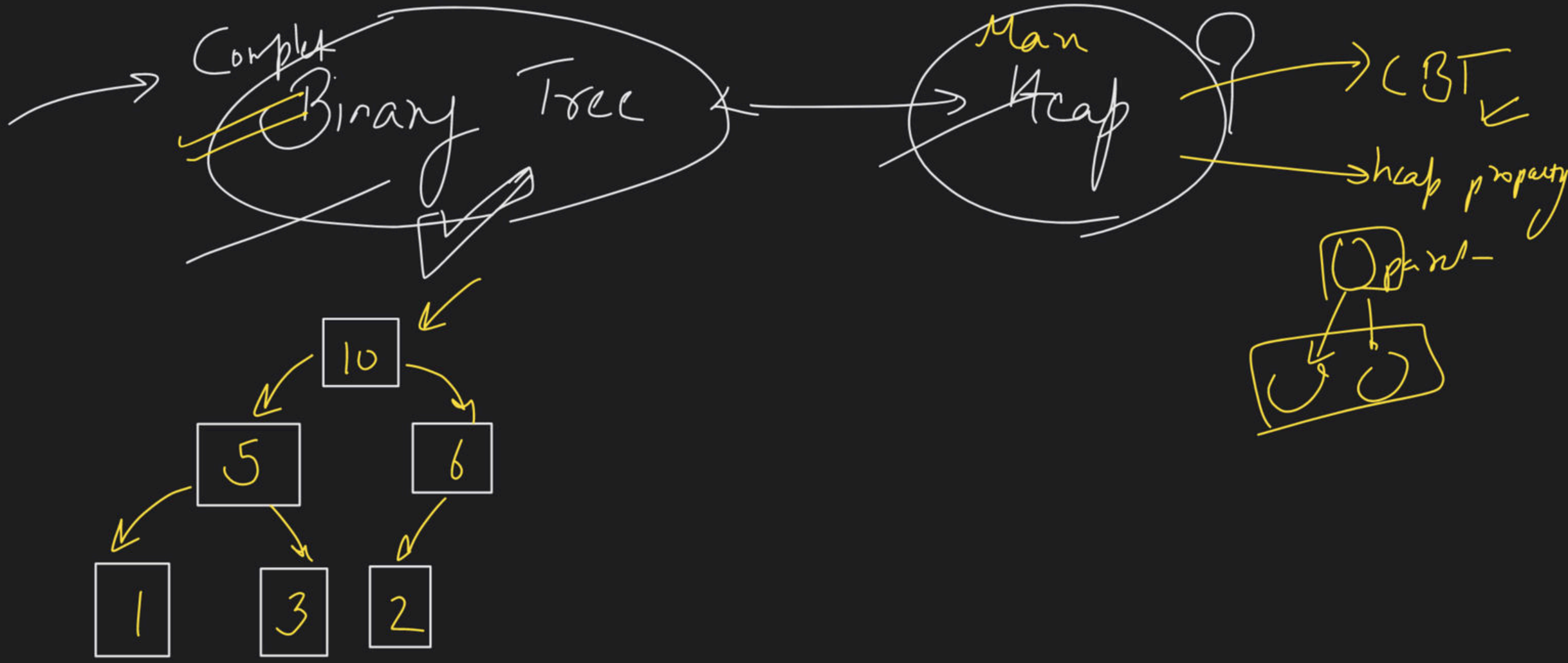


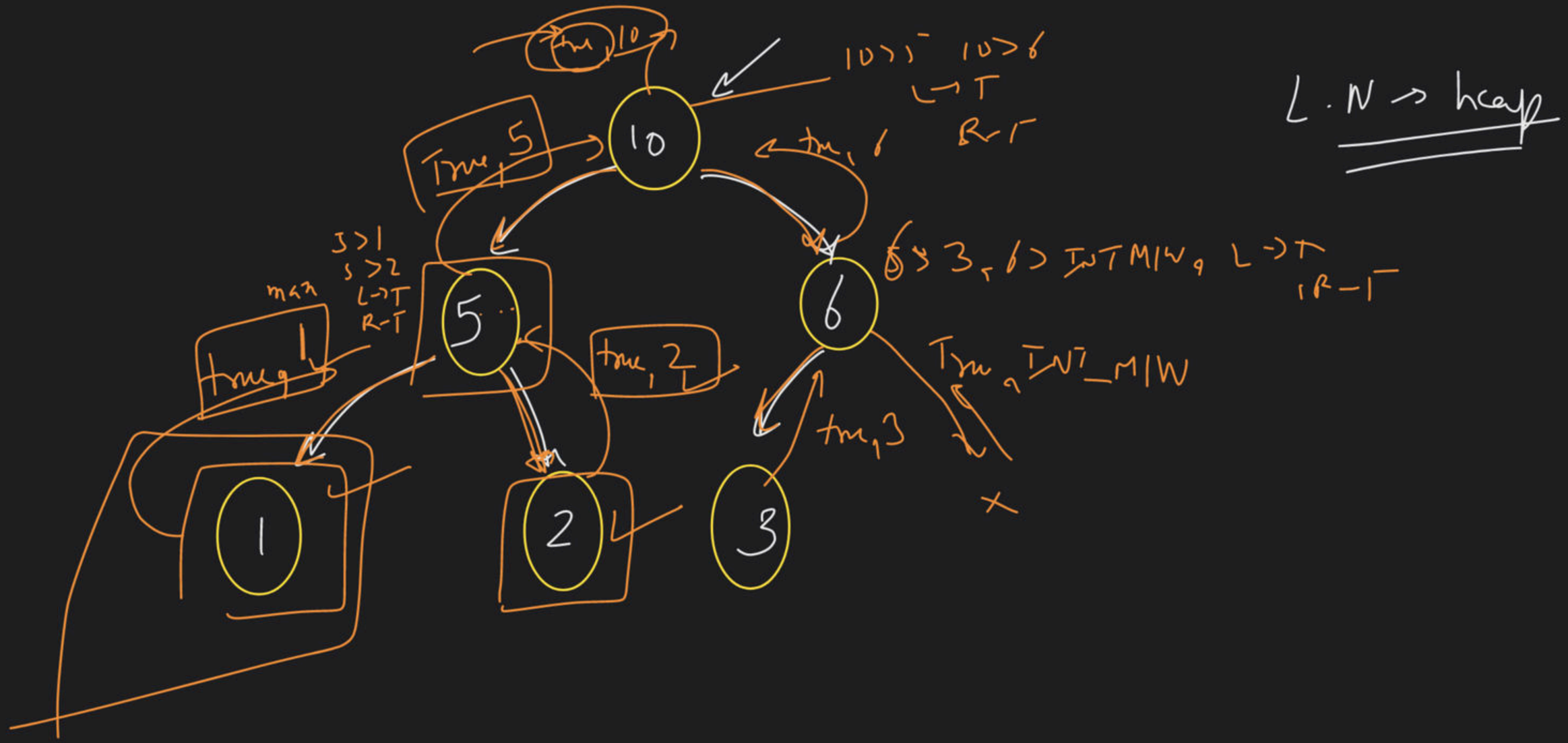
(1) arr → merge (heap1, heap2) → O(m+n)

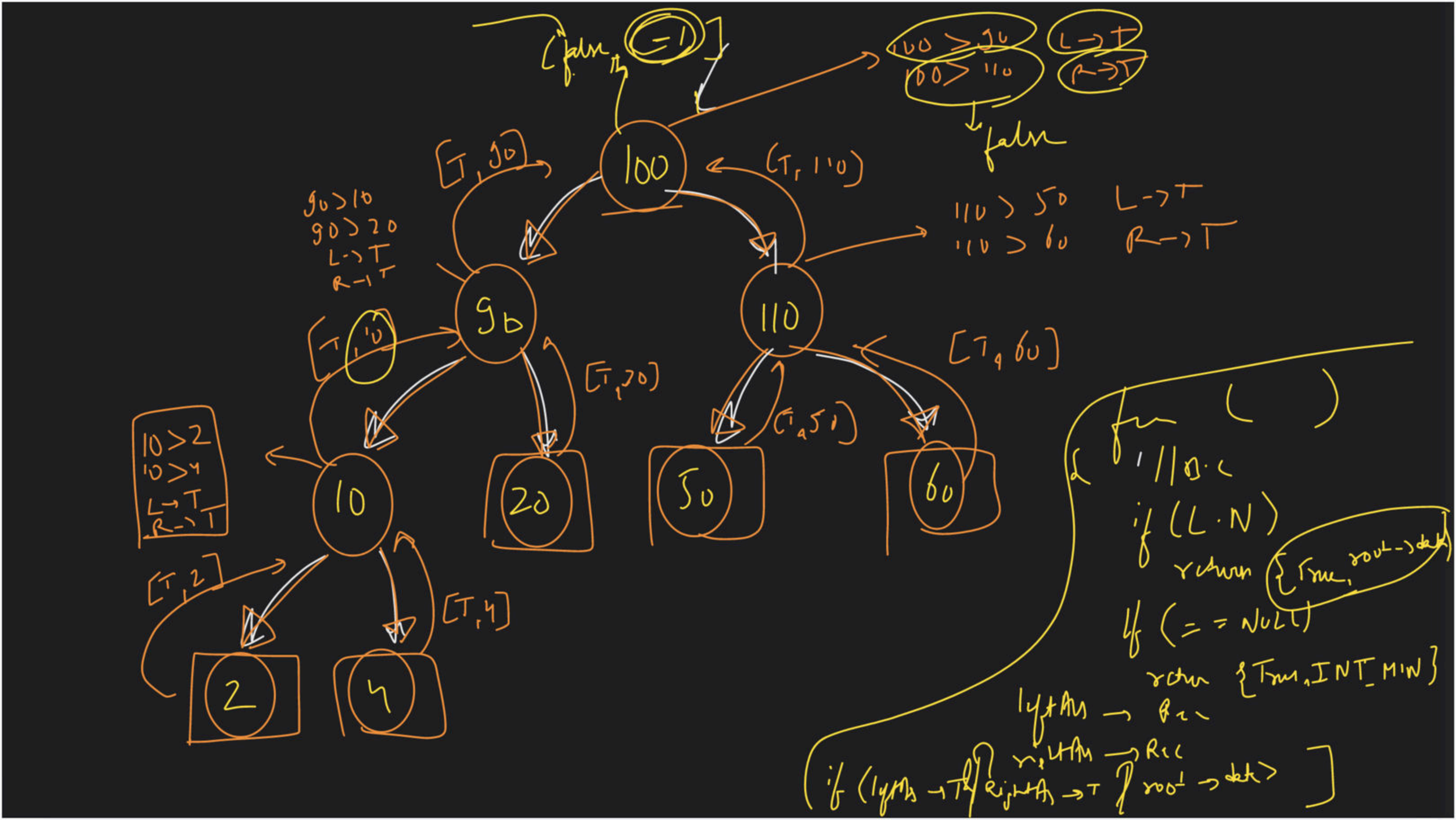
(2) ~~heapify~~ Buildheap → O(m+n)

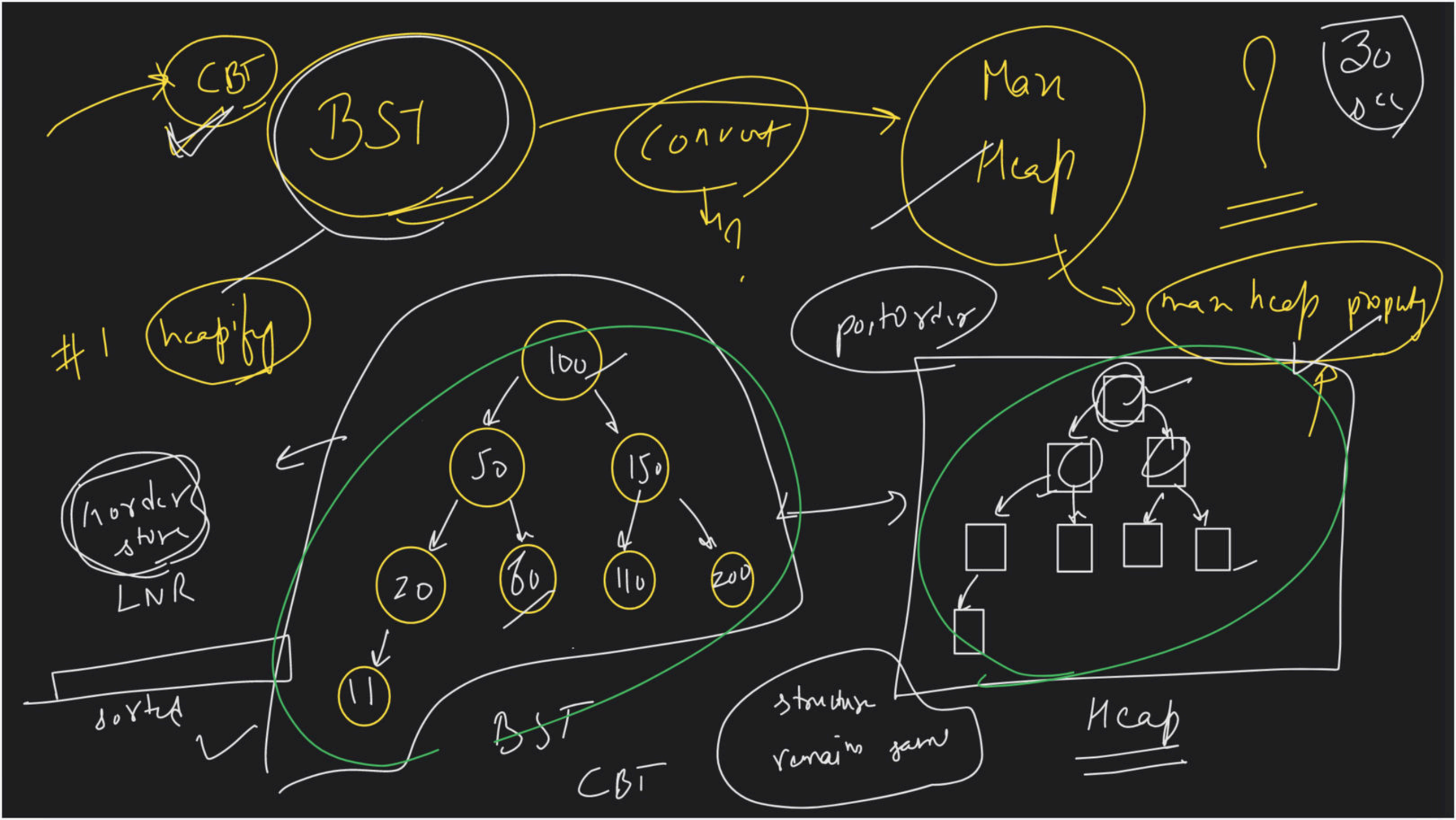
T.C

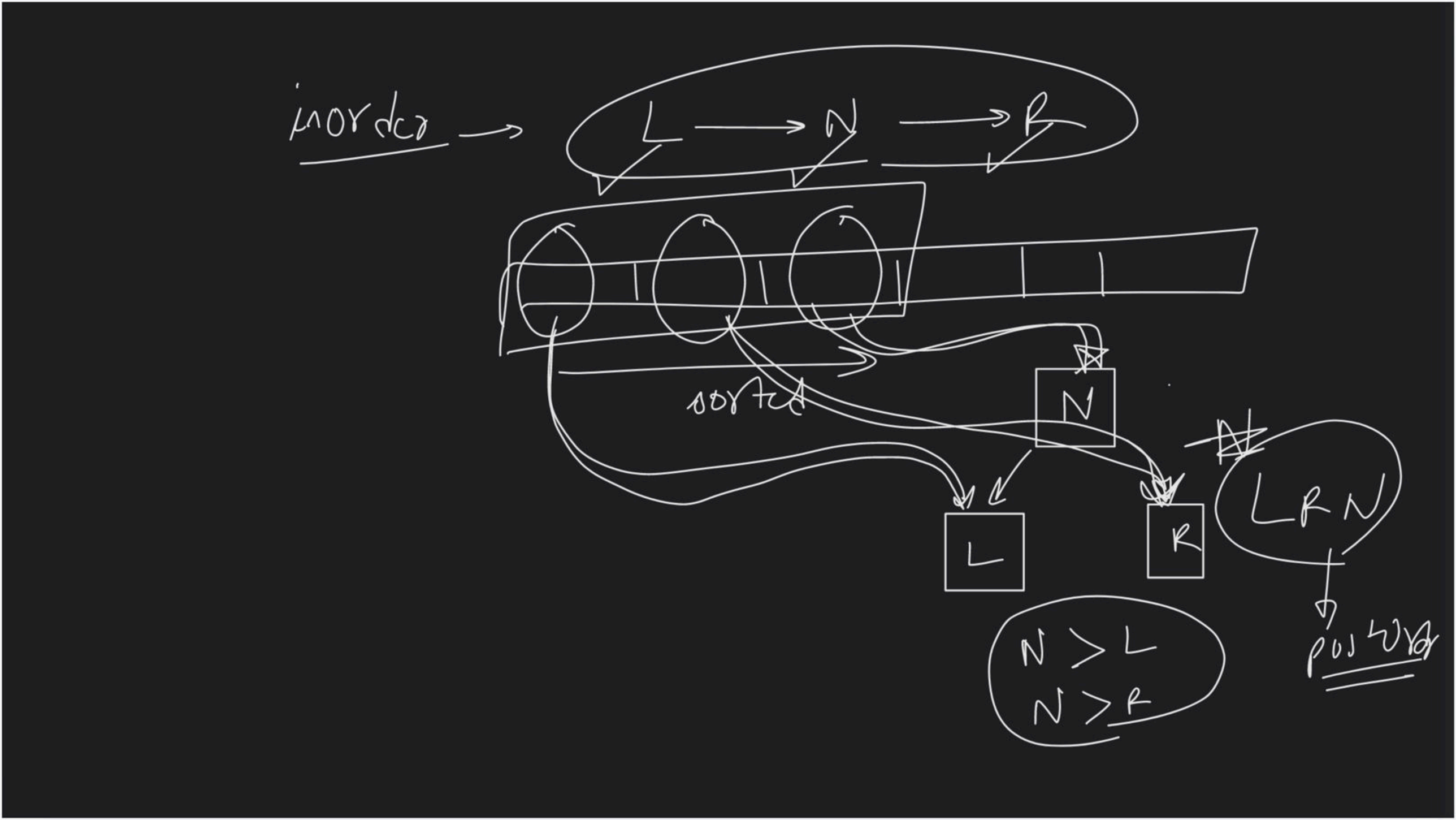

O(ntm)
O(nlogn)
heap1
1
logn
n
nlogn
heap2
L
O(nlogn)

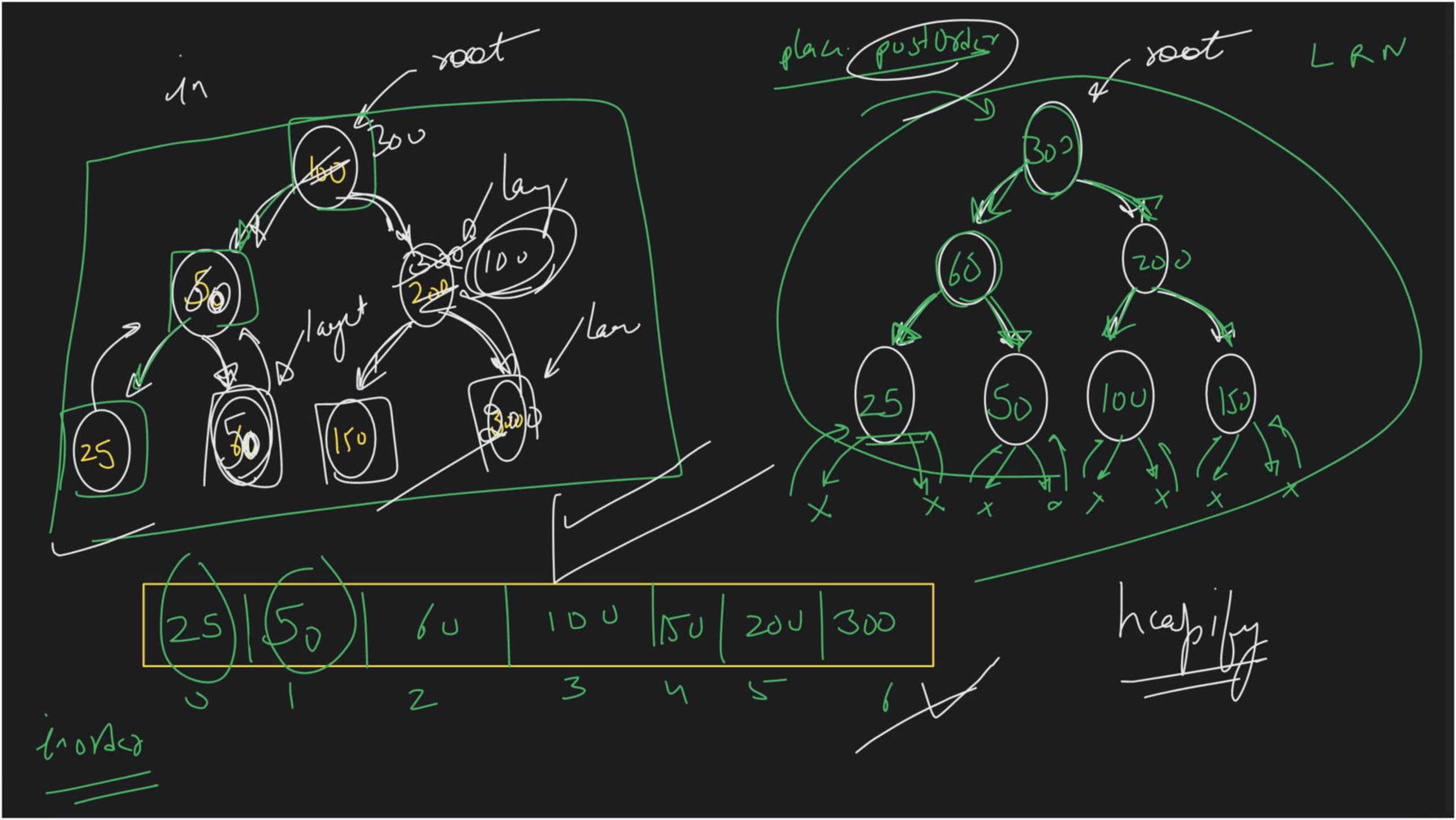

Complete
Binary Tree
Max
Heap
CBT
Heap property
parent
10
5
6
1
3
2

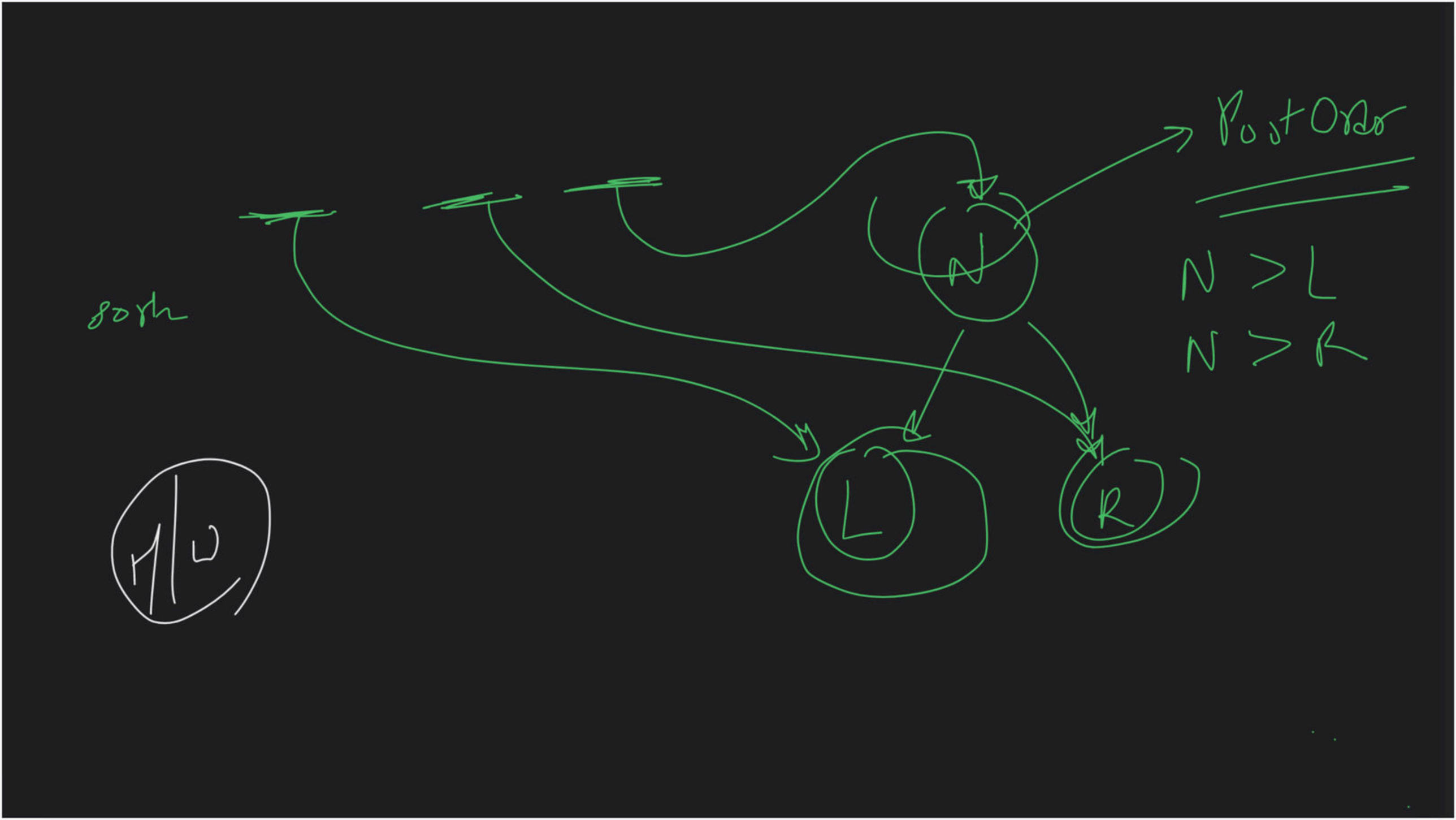

10
10 > 5
L -> T
R -> T
true, 5
true, 6
5 > 1
5 > 2
L -> T
R -> T
true, 1
true, 2
6 > 3, 6 > INT MIN, L -> T
R -> T
true, INT_MIN
true, 3
1
2
3
x
L . N -> heap

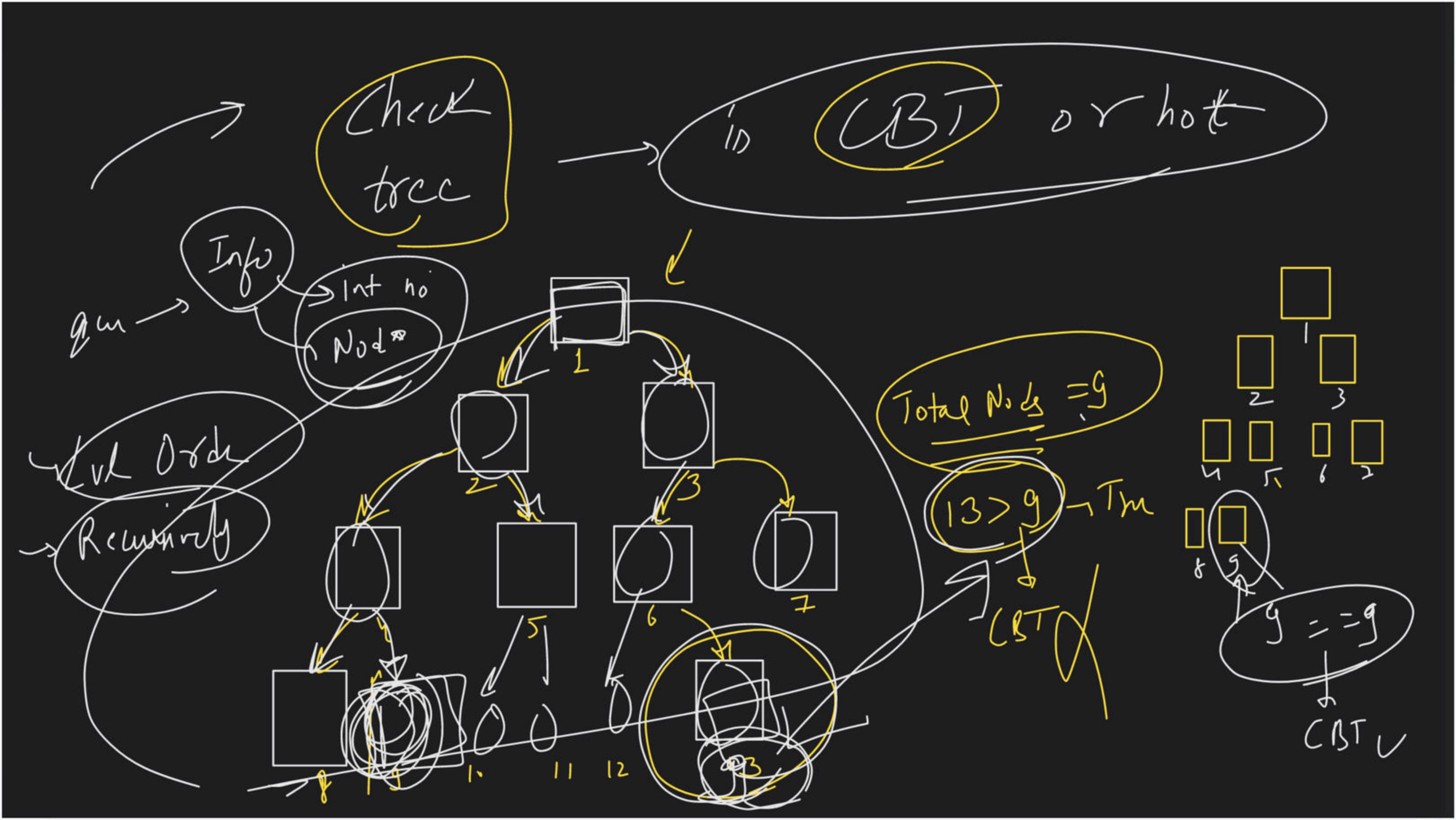

100
90
110
10
20
50
60
2
4
[T, 90]
[T, 110]
90>10
90>20
L->T
R->T
[T, 10]
[T, 20]
10>2
10>4
L->T
R->T
[T, 2]
[T, 4]
110>50
110>60
L->T
R->T
[T, 60]
[T, 50]
100>90
90>110
L->T
R->T
false
[false, -1]
fun ( )
if (L·N)
return {true, root->data}
if (== NULL)
return {true, INT_MIN}
if (leftMin -> T && rightMin -> T)
root -> data

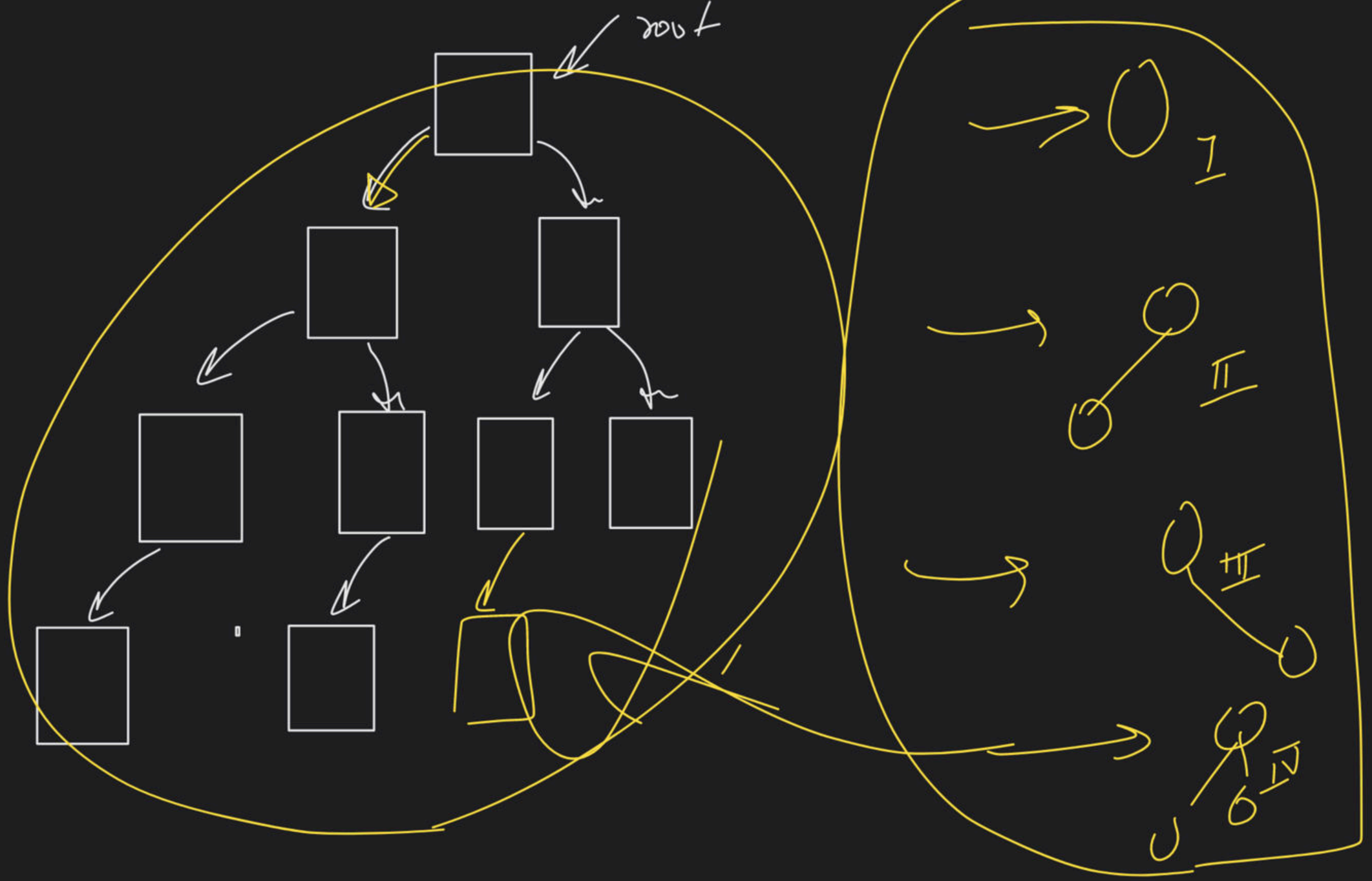

CBT
BST
convert
Max Heap
?
30
#1
heapify
inorder
LNR
100
50
150
20
80
110
200
11
sorted
BST
CBT
postorder
max heap property
structure
remains same
Heap

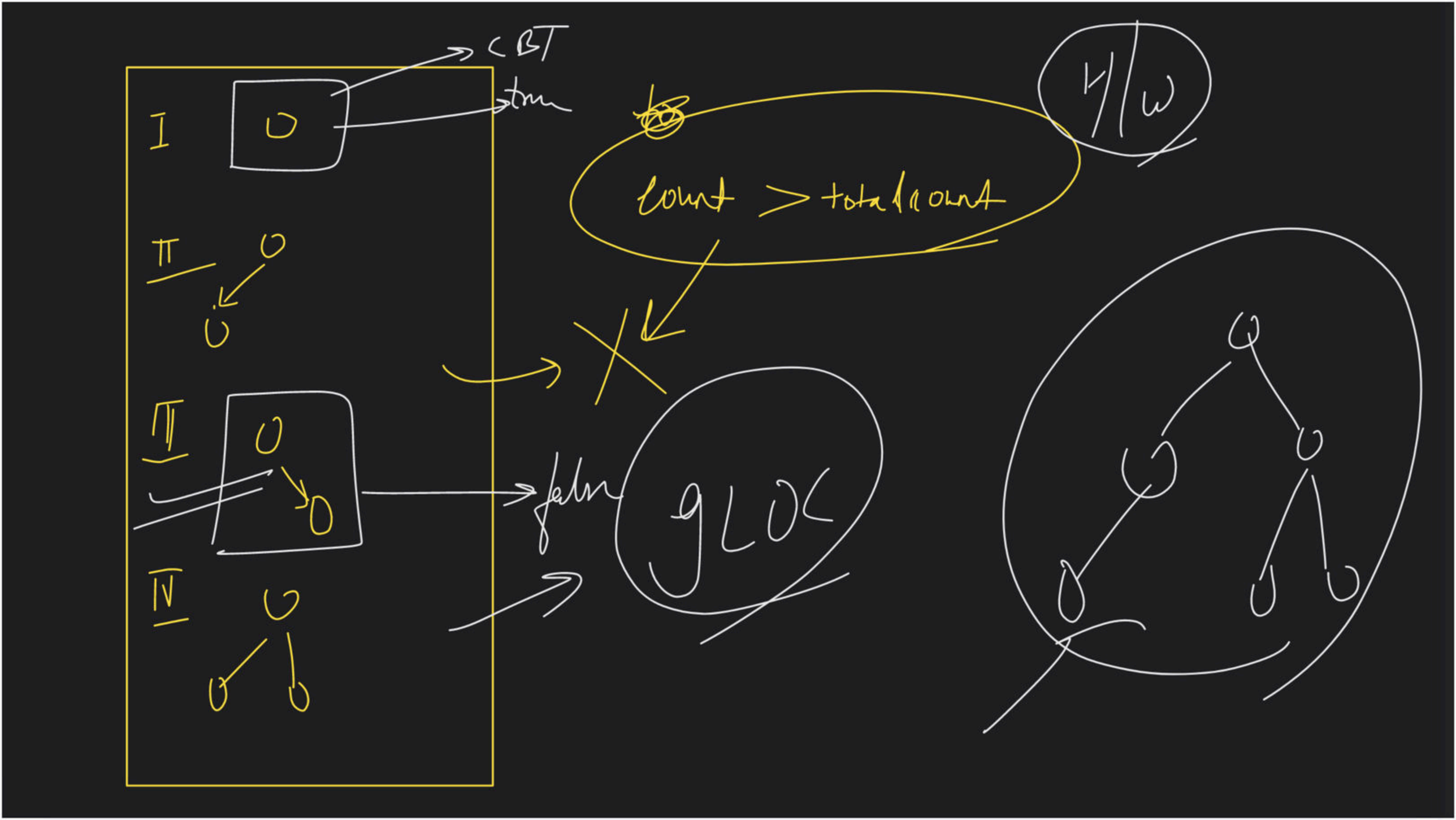

inorder
L
N
R
N
L
R
LRN
N > L
N > R

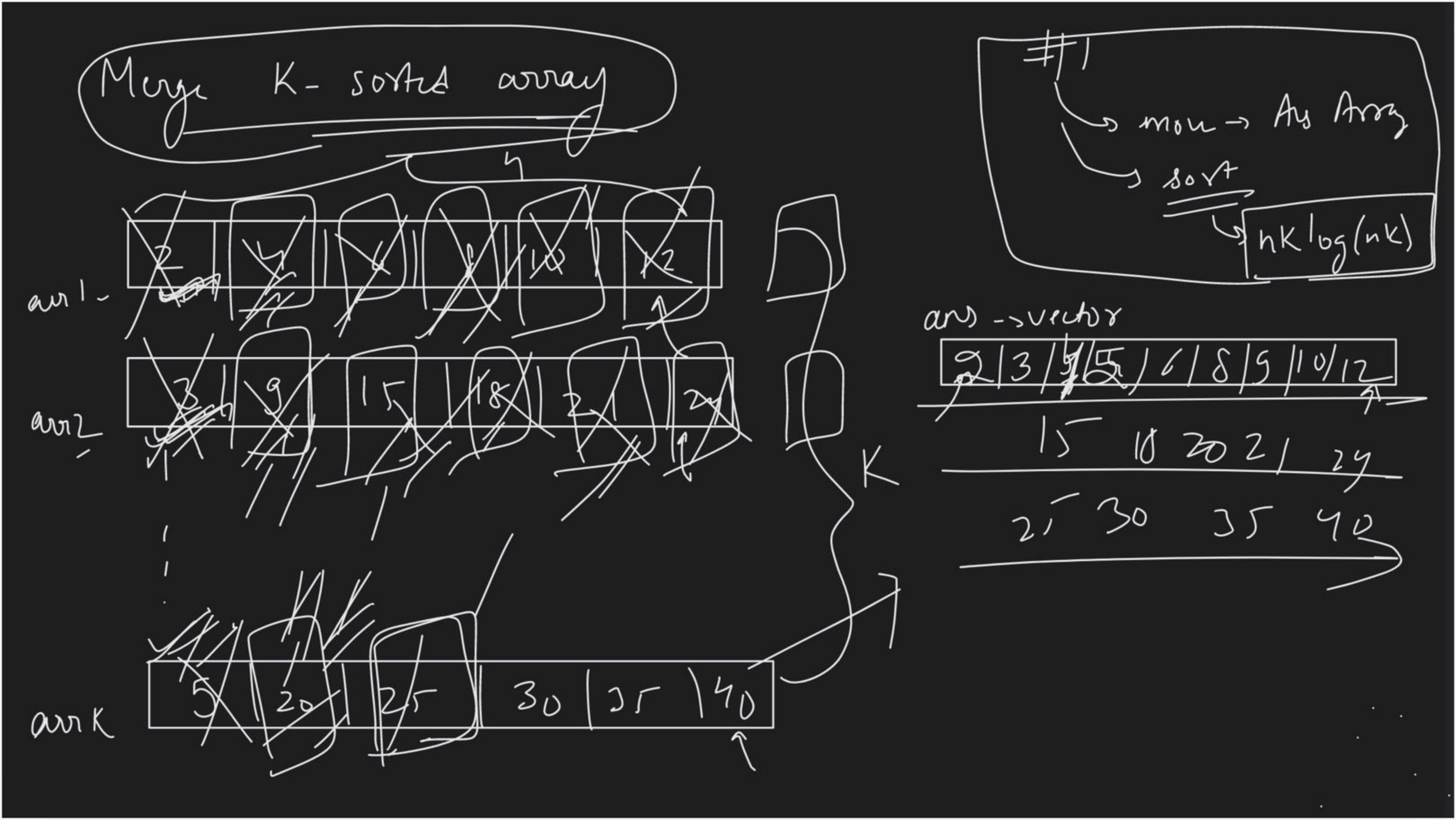

in
root
300
150
50
200
100
25
50
150
300
left
left
plan: postOrder
root
L R N
300
60
200
25
50
100
150
25
50
60
100
150
200
300
0
1
2
3
4
5
6
inorder
heapify

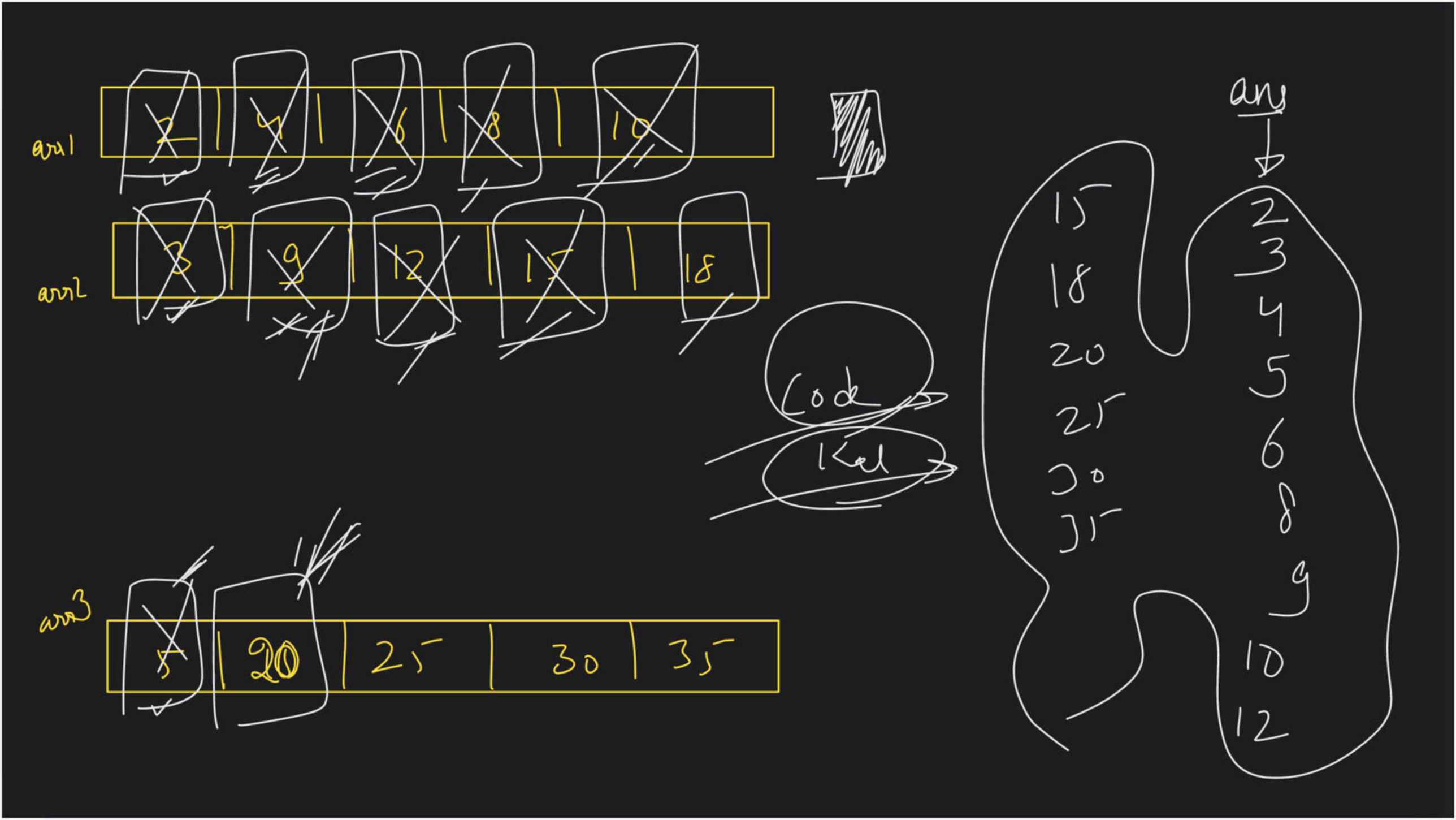

Post Order
N > L
N > R
both
N
L
R
H/D


Check
tree
is CBT or not
Info
Int no
Node
Level Order
Recursively
Total Nodes = 9
13 > 9
True
CBT
1
2
3
4
5
6
7
8
9
10
11
12
13
9 == 9
CBT


root
I
II
III
IV


CBT
tru
I
II
III
IV
count > totalRound
false
glOc


Merge K- sorted array
arr1 -
arr2 -
arrK
K
#1
→ mov → Any Array
→ sort
nk log(nk)
ans -> vector
9 13 15 16 18 19 10 12
15 18 20 21 24
25 30 35 40


arr1
2
4
6
8
10
arr2
3
9
12
15
18
arr3
5
20
25
30
35
code
Kal
any
15
18
20
25
30
35
2
3
4
5
6
8
9
10
12