CS & IT ENGINEERING

Algorithms

Heap Algorithms

Lecture No.- 01

By- Dr. Khaleel Khan sir

Recap of Previous Lecture

Topic — Sorting Techniques

Topic

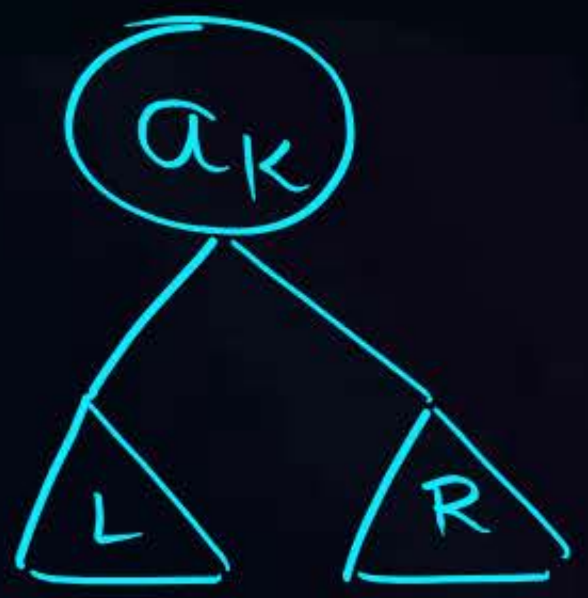
Topics to be Covered

Topic Heap Algorithms

Topic

Priority Queue :

Definition: A Heap is a complete binary tree with the property that the value at each node is at least as large as the values at its children (if they exist).



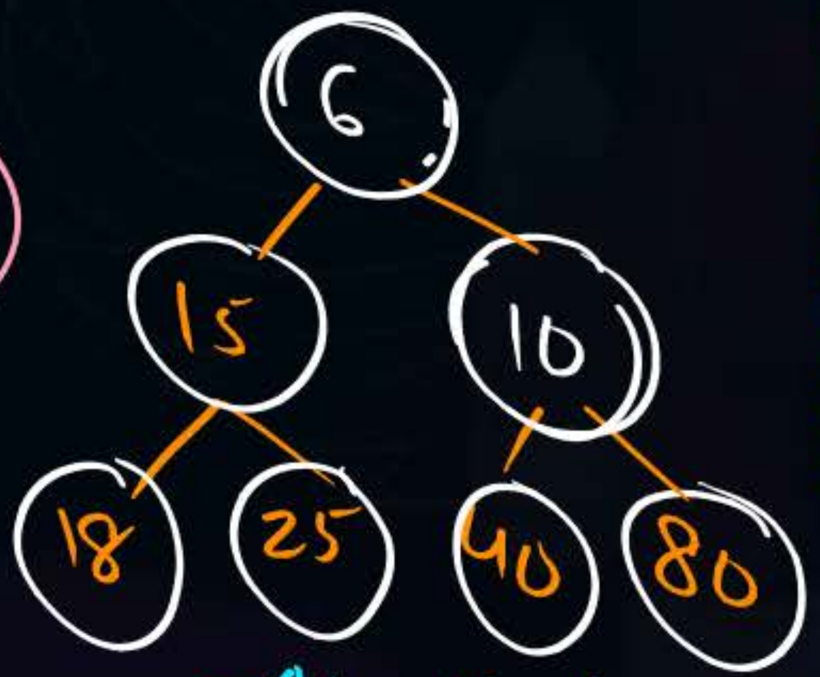
1) $a_K > |L, R|$: Max-Heap

2) $a_K < |L, R|$: Min-Heap

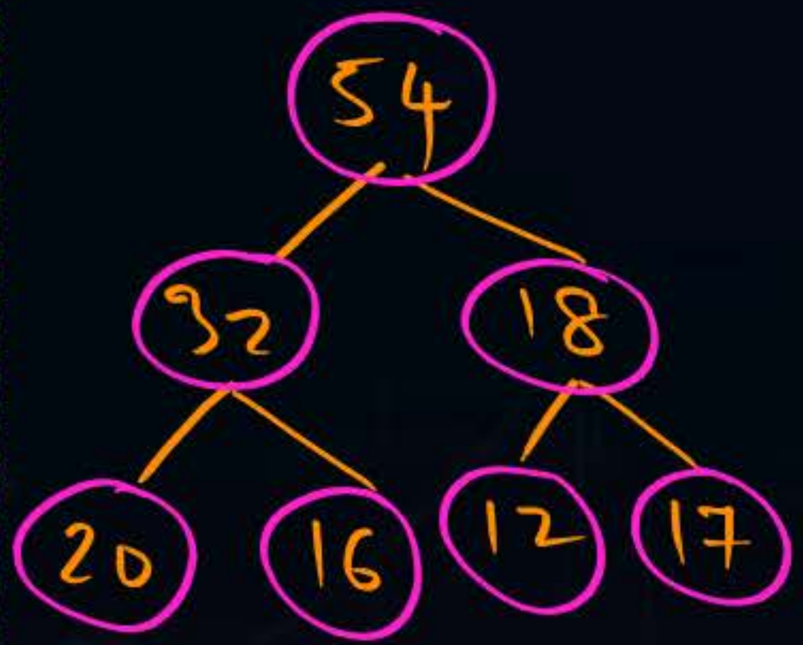
3) $|L| < a_K < |R|$

Need Not be: B.S.T

Complete



Min-Heap



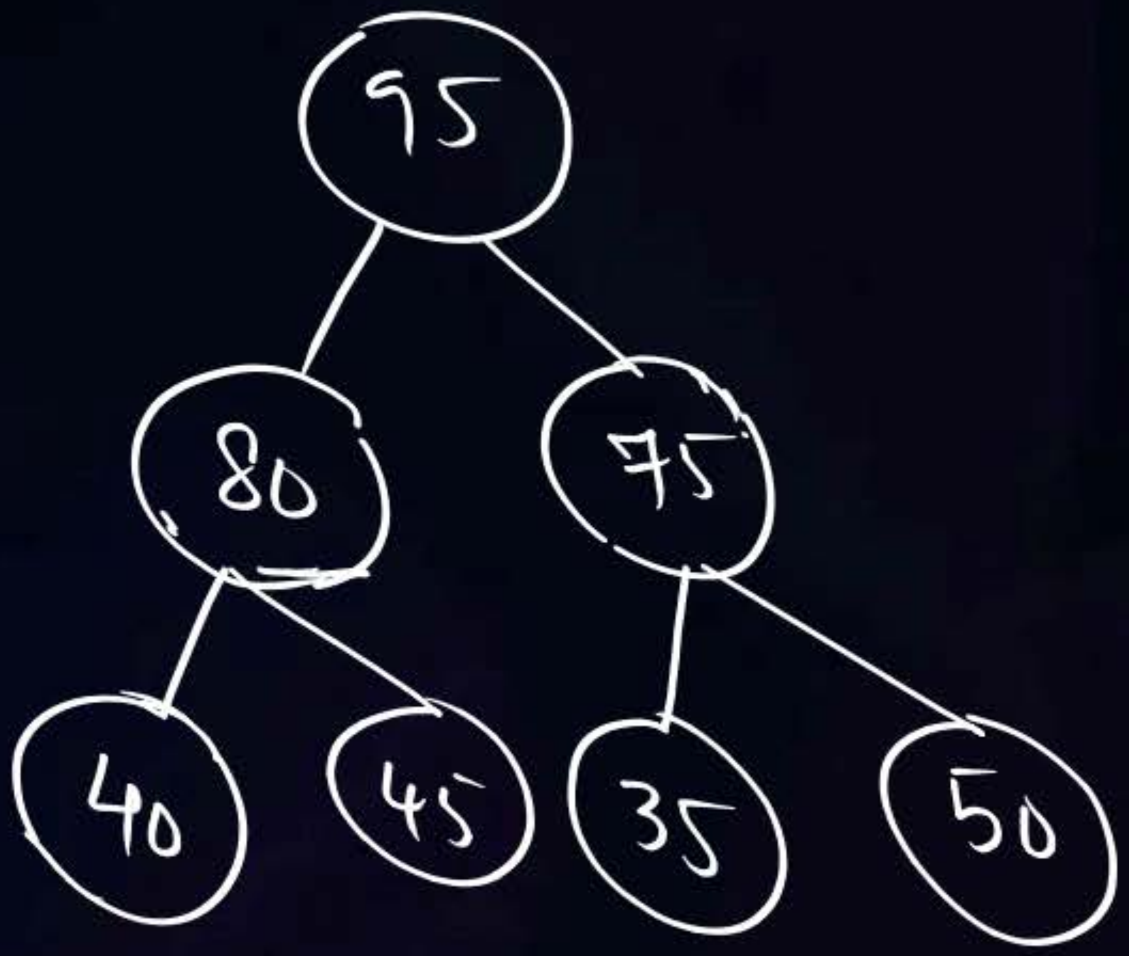
Max-Heap

Heap Construction:

1) **Insertion Method:** → Insert one element @ a time Starting from an Empty Tree;

$\langle 40; 80; 35; 95; 45; 50; 75 \rangle$

Max-Heap



1) Best Case: $\langle$ Decreasing order $\rangle$

$O(n)$

2) Worst Case: $\langle$ Inc. order $\rangle$

$\boxed{O(n \cdot \log n)}$

```
procedure INSERT(A, n)
integer i, j, n,;
j — n; i ← ⌊n/2⌋; item ← A(n)
while(i > 0 and A(i) < item)do
    A(j) ← A(i) //move the parent down//
    j ← i; i ← ⌊i/2⌋
repeat
A(j) ← item     //a place for A(n) is found//
end INSERT
```

: Inserting an Element

```
for i ← 2 to n do
        call INSERT(A, i)
repeat
```

II: Insert - op n :

Given a Heap with n-elements, the Time Complexity to Insert an element into it is $O(\log n)$

Time-Complexity - Worst Case :

→ Max No. of Nodes @ level 'i' of a Binary Tree : 2^{i-1}

→ The No. of level Comp's (movements) for a Node getting inserted @ level 'i' : $(i-1)$

→ Total No. of level Comp's for all (Max) Nodes @ level 'i' : $(i-1)\cdot 2^{i-1}$

→ Time = $T(n)$ = No. of Comps/Mov's for all Nodes @ all levels (1..K) $= \sum_{i=1}^{K}(i-1)\cdot 2^{i-1}$

$$\sum_{i=1}^{K}(i-1)\cdot 2^{i-1}$$

$$\frac{1}{2}\left[\sum_{i=1}^{K} i\cdot 2^{i} - \sum_{i=1}^{K} 2^{i}\right]$$

$$= \frac{1}{2}\left[(K-1)\cdot 2^{K+1} + 2 - \left(2^{K+1} - 2\right)\right]$$

$$= \frac{1}{2}\left[K\cdot 2^{K+1} - 2^{K+1} + 2 - 2^{K+1} + 2\right]$$

$$= K\cdot 2^{K} - 2\cdot 2^{K} + 2$$

Let $n = 2^{K}$

$K = \log n$

$$T(n) = \boxed{n\cdot\log n - 2n + 2}$$

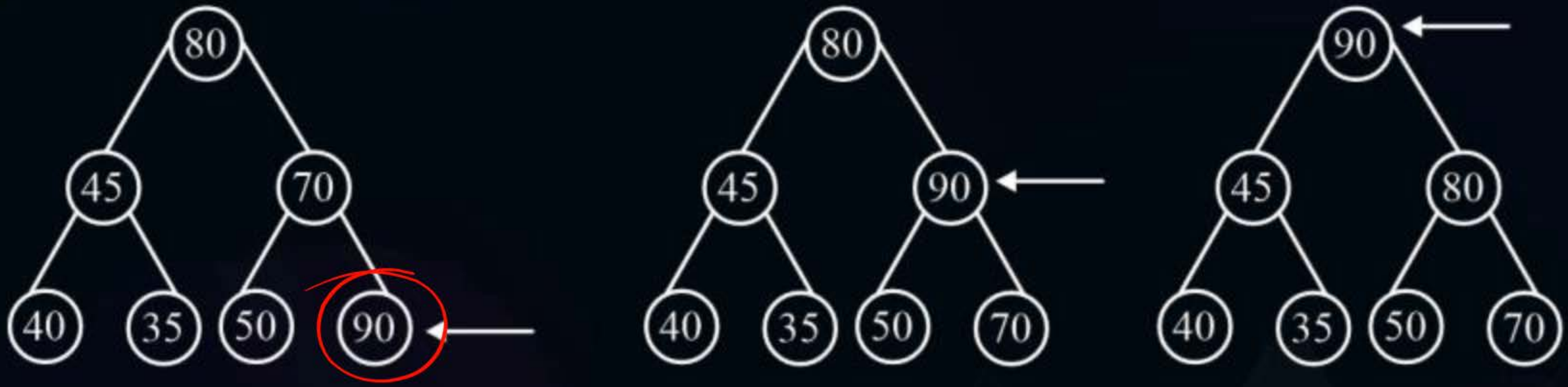
$$= O(n\cdot\log n)$$

$$\sum_{i=1}^{n} i \cdot 2^{i} = \boxed{(n-1)\cdot 2^{n+1} + 2}$$

$$\sum_{i=1}^{n} 2^{i} = \frac{2(2^{n}-1)}{2-1}$$

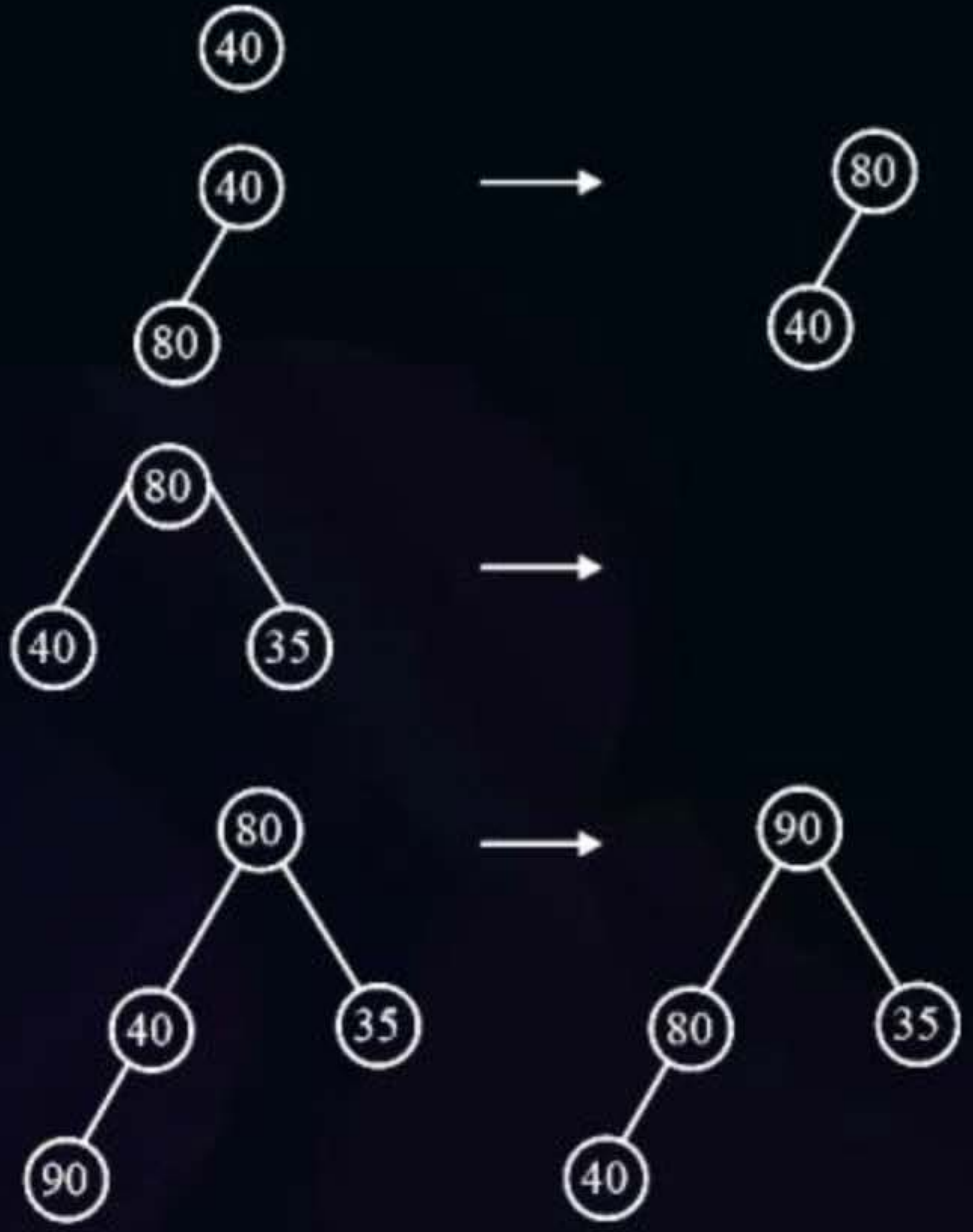
$$= \boxed{2^{n+1} - 2}$$

Topic : Algorithms

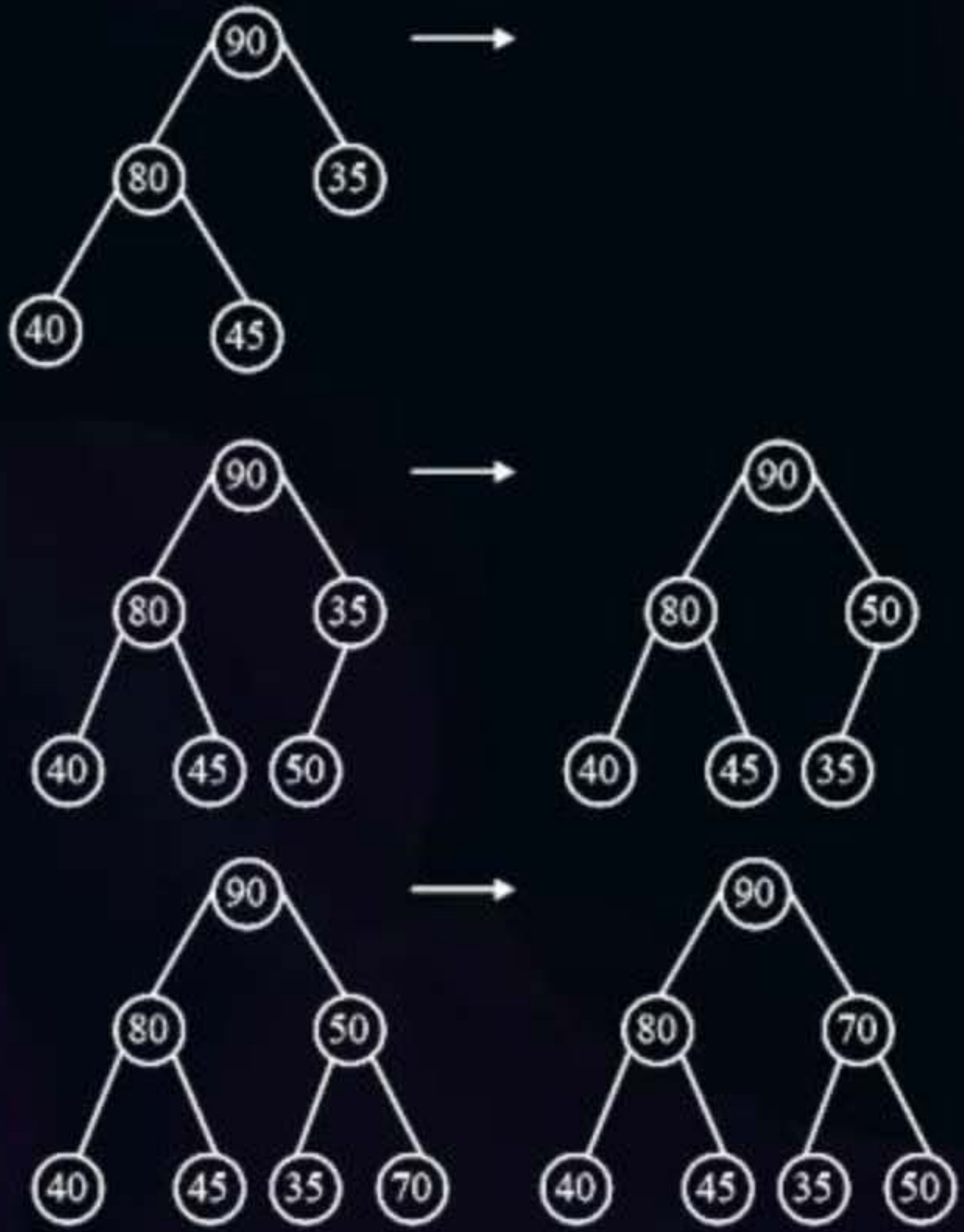


Action of INSERT inserting 90 as the seventh item into an existing heap

Topic : Algorithms



Topic : Algorithms



Slide 8

Forming a heap from the set (40,80,35,90.45.50,70)

II : Build-Heap | Heapify :

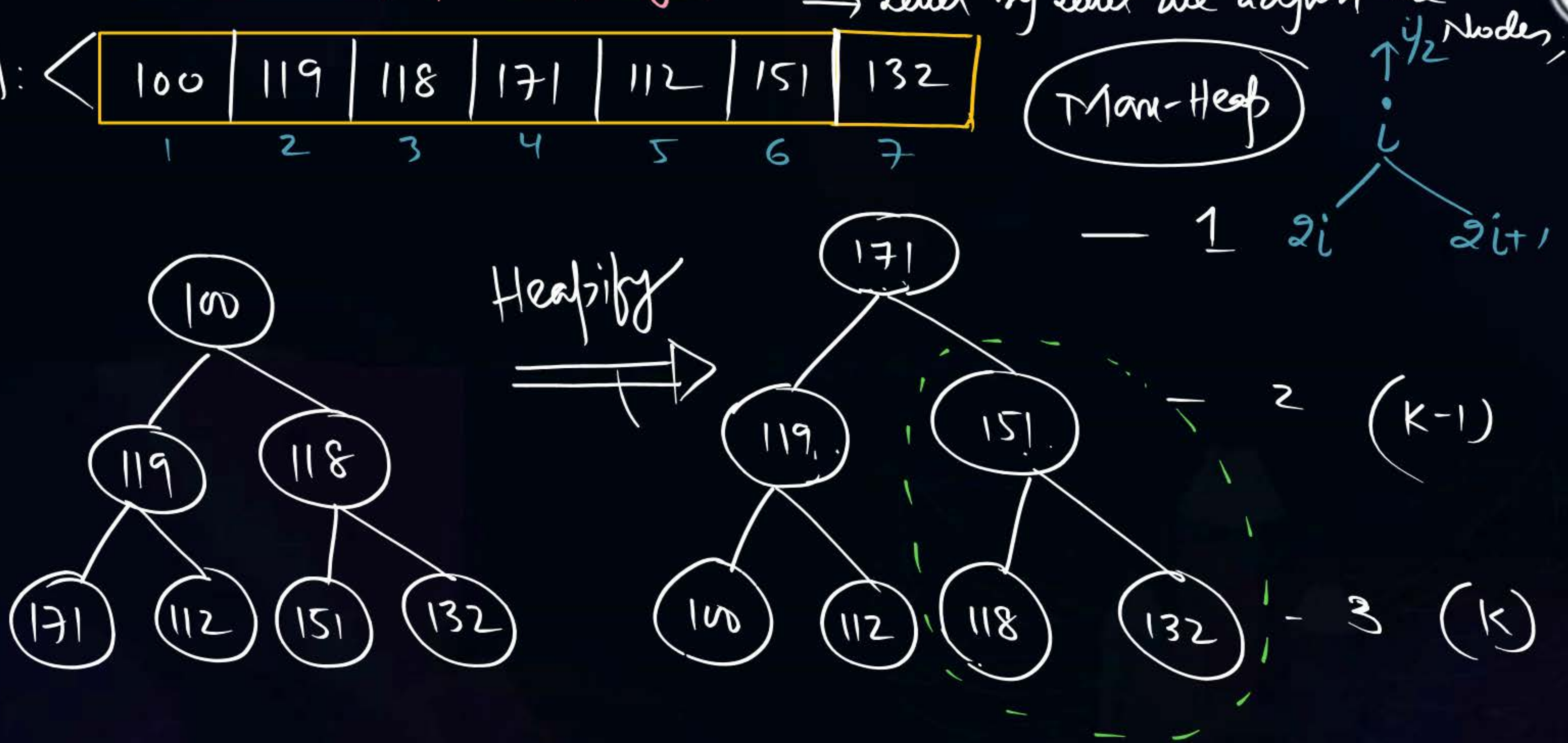
→ Tree already Exists.

→ Level by Level we adjust the ½ Nodes.

A: < 100 | 119 | 118 | 171 | 112 | 151 | 132 >

1	2	3	4	5	6	7
100	119	118	171	112	151	132

Max-Heap



Topic : Algorithms

```
procedure ADJUST(A, i, n)
integer i,j, n;
j ← 2 * i; item ← A(i)
while j ≤ n do
    if  (j ≤ n and A(j) < A(j + 1) ) then
      j ← j + 1 //j points to the larger child//
    end if
    if (item ≥ A(j) )   then
      exit // a position for item is found //
    else
      A (⌊j/2⌋) ← A(j)// move the larger child up a level//
    J ←2*j
    end if
repeat
A (⌊j/2⌋) ← item
end ADJUST
```

Topic : Algorithms

procedure **HEAPIFY** (A ,n)

//Readjust the elements in A(1 : n) to form a heap//

integer n ,i

for i ← $\lfloor n/2 \rfloor$ to 1 by –1 do

call ADJUST (A, i, n)

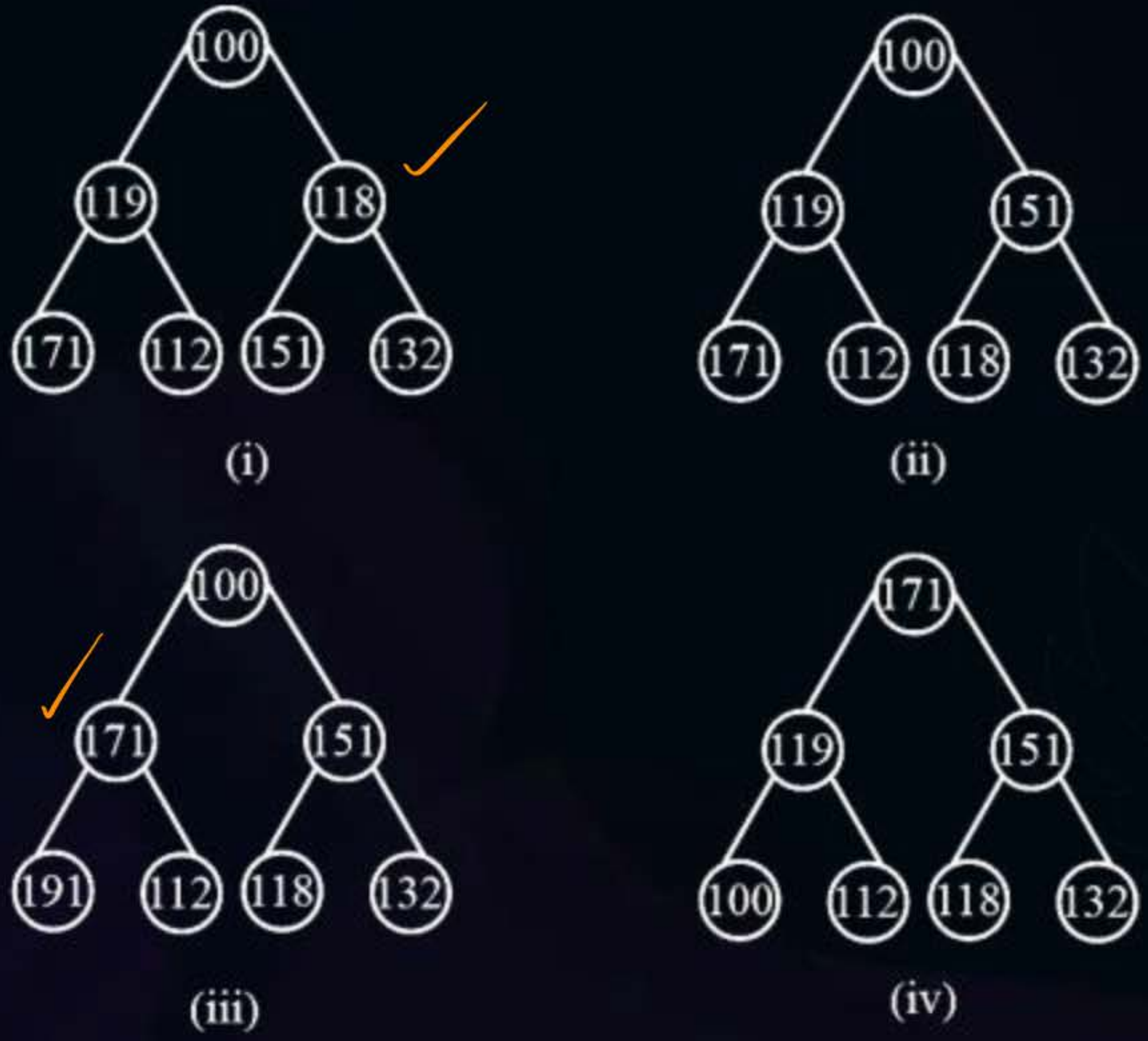
repeat

end HEAP1FY

$n = 7/2$

$= 3$

Topic : Algorithms



Action of HEAP1FYG4, 7) on the data of (100, 119, 118. 171. 112, 151, 132)

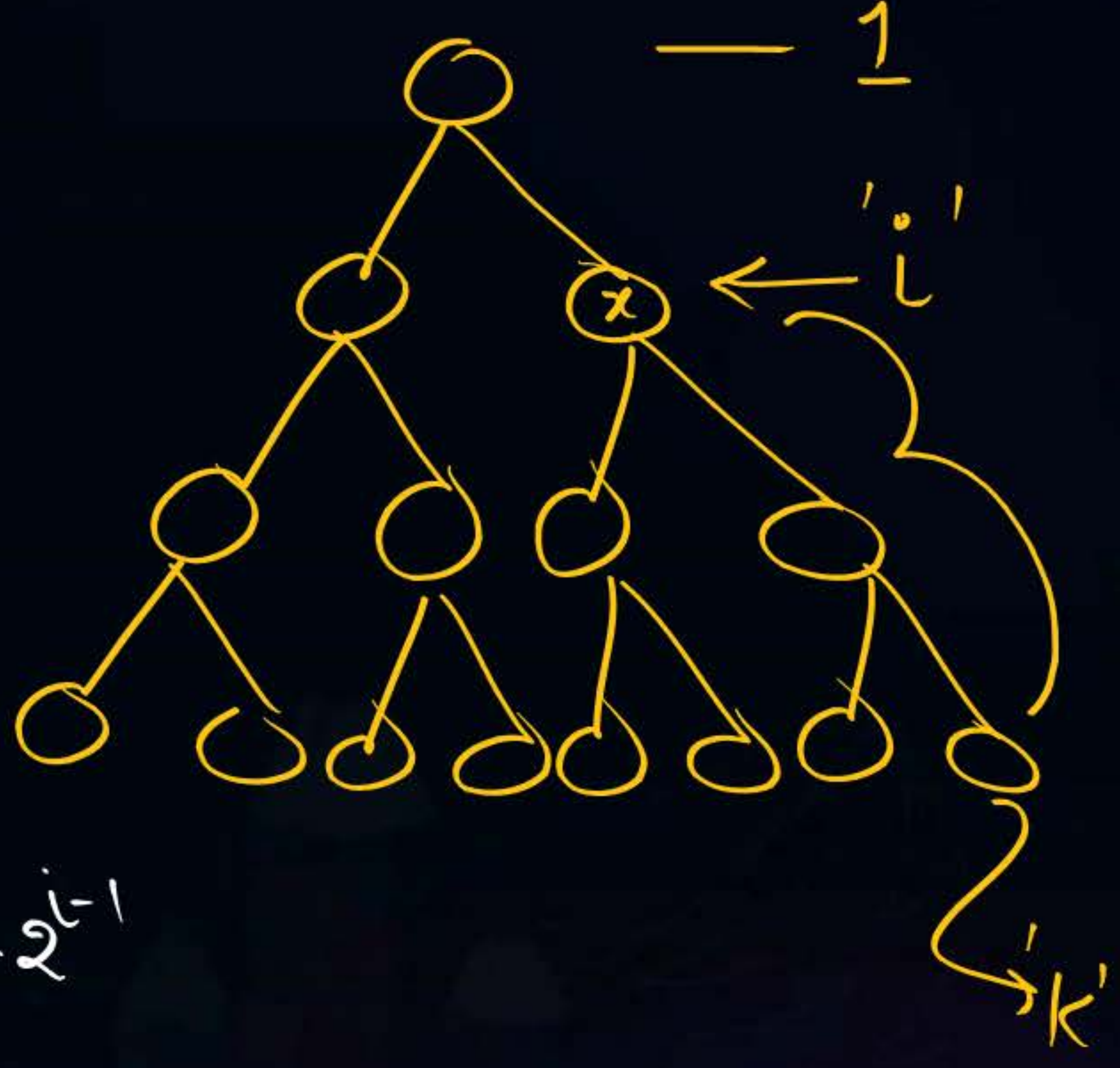
Time Complexity of Build-Heap (Heapify):

→ Max. # of Nodes @ level 'i' of a B.T $= 2^{i-1}$

→ Max # of level Comp's/mov's for a Node getting adjusted @ any level 'i' $= (k-i)$

→ For all nodes (max) the No. of level Comp's $= (k-i) \cdot 2^{i-1}$

→ Total Time = Sum of all level comp's for all Nodes @ all levels (1 .. k) $= T(n) = \sum_{i=1}^{k} (k-i) \cdot 2^{i-1}$



$$T(n) = \sum_{i=1}^{K} (K-i)\cdot 2^{i-1}$$

$$= \frac{1}{2}\left[\sum_{i=1}^{K} K\cdot 2^{i} - \sum_{i=1}^{K} i\cdot 2^{i}\right]$$

$$= \frac{1}{2}\left[K\left(2^{K+1}-2\right) - \left((K-1)\cdot 2^{K+1}+2\right)\right]$$

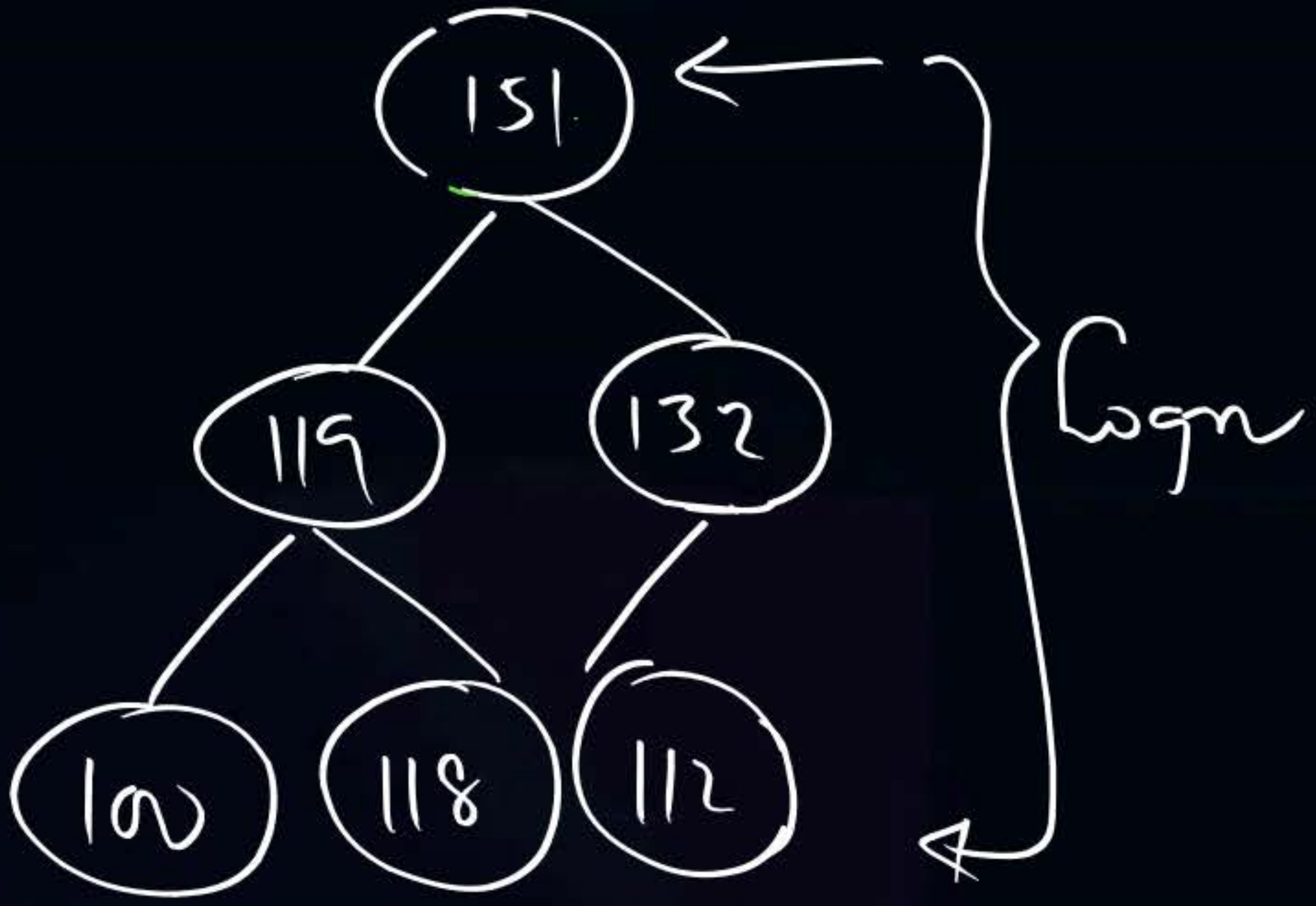
$$= \frac{1}{2}\left[\cancel{K\cdot 2^{K+1}} - 2K - \cancel{K\cdot 2^{K+1}} + 2^{K+1} - 2\right]$$

$$= 2^{K} - K - 1$$

$$T(n) = \boxed{n - \log n - 1} = O(n)$$

$n = 2^{K}$

Delete-opn in a Heap : Deleting the Root



1 (i) Swap Root (A[1], A[n])

Logn (ii) Adjust (A[1])

Delete : O(Logn) ✓

procedure **HEAPSORT** (A ,n) : $O(n \cdot \log n)$ ✓

//A(1 : n) contains n elements to be sorted.//

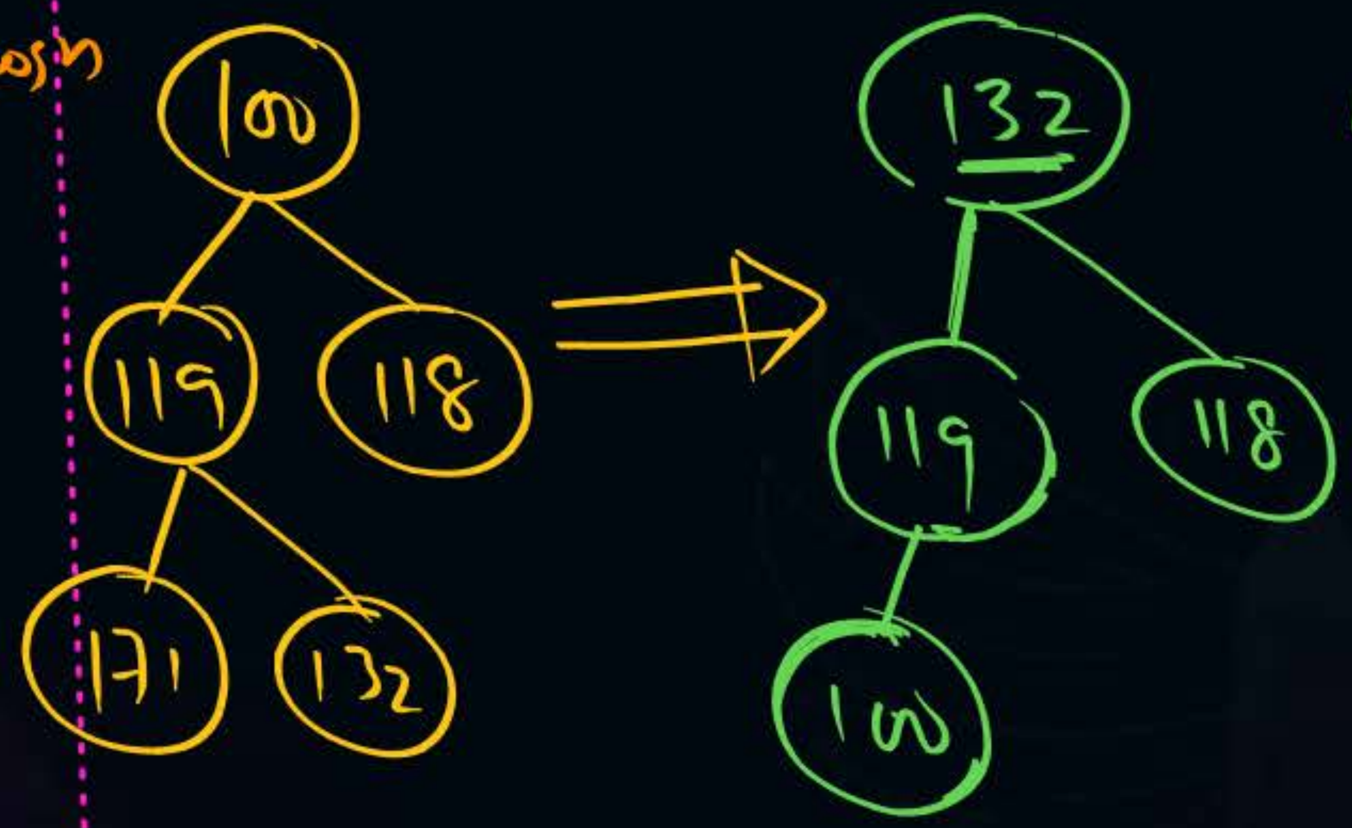
1. call HEAPIFY (A,n) : $O(n)$

2. for i ← n to 2 by - 1 do : $(n-1)$: $n \log n$

Delete / $\log n$ { t ← A(i); A(i) ← A(1); A(i) ← t
call ADJUST, (A, 1, i –1) }

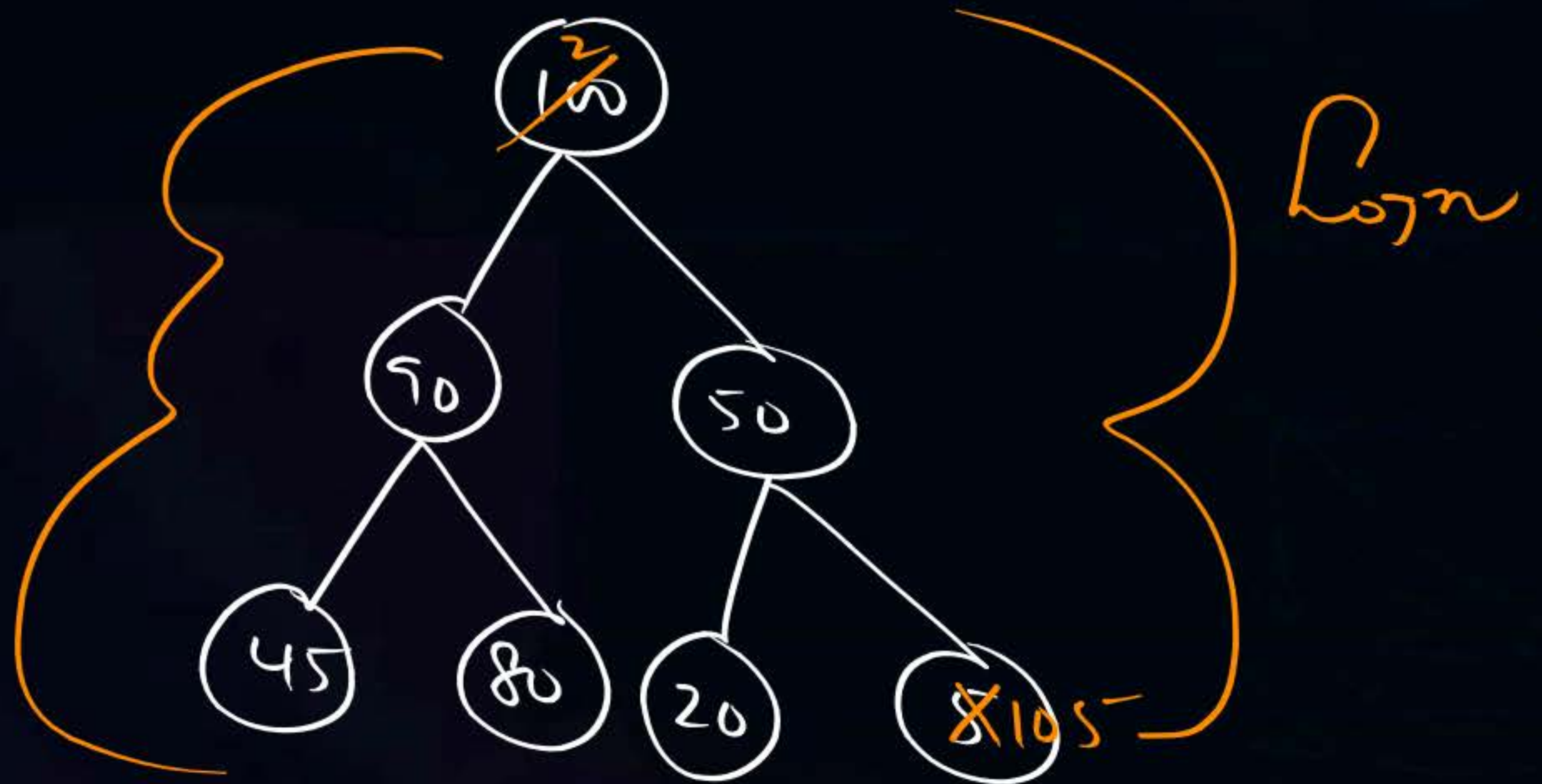
repeat

end HEAPSORT



	1	2	3	4	5
A	100	119	118	171	132
A	171	132	118	119	100
A	100	132	118	119	171
A	132	119	118	100	171
	100	119	118	132	171

Given a Heap with n-elements, The Time Complexity to Perf. the Opn of Inc/Dec. Key,



$O(\log n)$ ✓

Topic VI. Heaps

<7, 2, 10, 4>

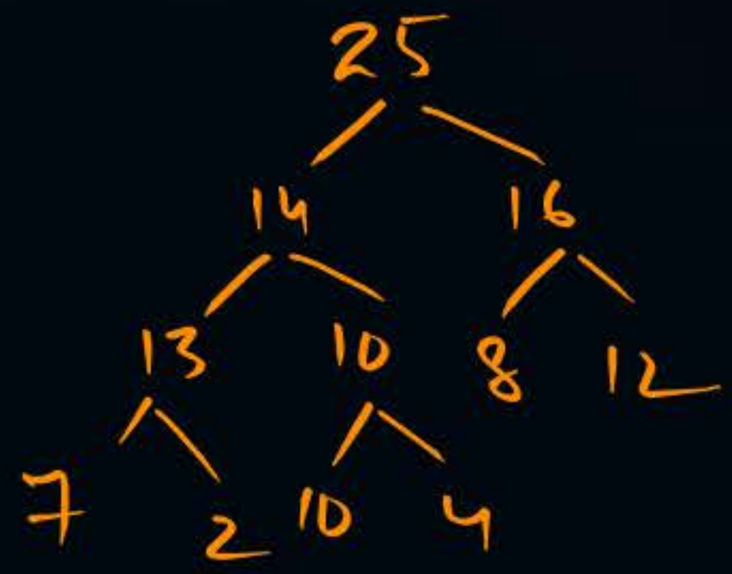
Q). Which Array Representation is a valid Binary Max-Heap

(a) <25, 12, 16, 13, 10, 8, 14>

(b) <25, 14, 16, 13, 10, 8, 12>

(c) <25, 14, 13, 16, 10, 8, 12>

(d) <25, 14, 12, 13, 10, 8, 16>



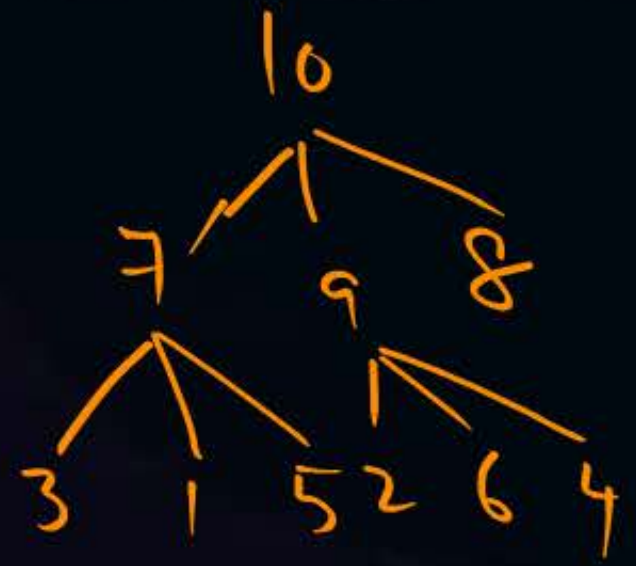
Q). Which one is valid 3-ary Maximum Heap Array representation

(a) <1, 3, 5, 6, 8, 9>

(b) <9, 6, 3, 1, 8, 5>

(c) <9, 3, 6, 8, 5, 1>

(d) <9, 5, 6, 8, 3, 1>

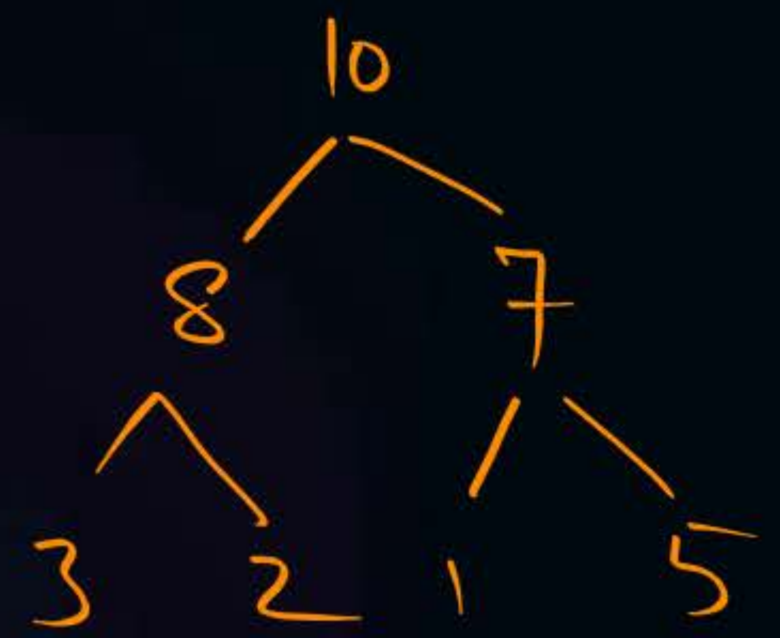


Topic VI. Heaps

Q). To the valid Heap of Previous Question insert elements < 7 2 10 4 >. Indicate the resultant Heap in Array.

Topic VI. Heaps

Q). Level order traversal of a binary max Heap generates: <10, 8, 5, 3, 2>. To This Heap Insert: <1 & 7> ; What is the resultant Level order Traversal



$\langle 10, 8, 7, 3, 2, 1, 5 \rangle$

Topic VI. Heaps

Q). In a Binary Max-Heap with n elements, the smallest element can be found in time of $O(n)$.

Convert Max-Heap to Min Heap [Heapify] : $O(n)$

Q). Given binary Heap with 'n' elements & it is required to insert 'n' more elements not necessarily one after another into this Heap. Total time required for this operation is:

(a) $O(n^2)$ (b) nlogn

(c) n ✓ (d) n^2logn

$\boxed{\text{'n'}} + \boxed{\text{'n'}}$

$(2n) \Rightarrow$ Heapify

Topic VI. Heaps

Min Heap

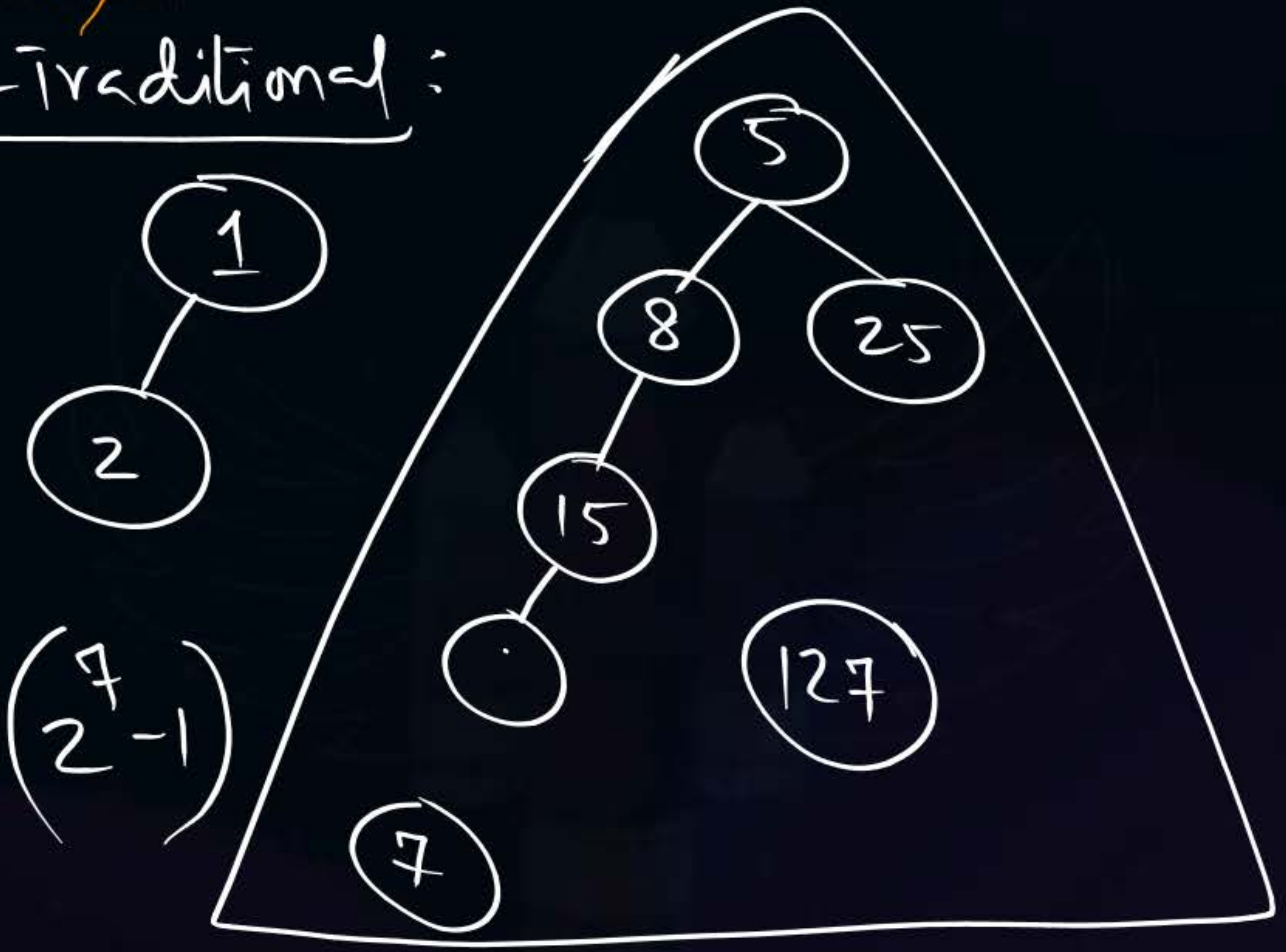
Q). Given Binary Heap in Array with the smallest at the root, the 7th smallest element can be found in time complexity of $\underline{O(1)}$.

Traditional Approach:

$\Longrightarrow$ 7 delete op'ns

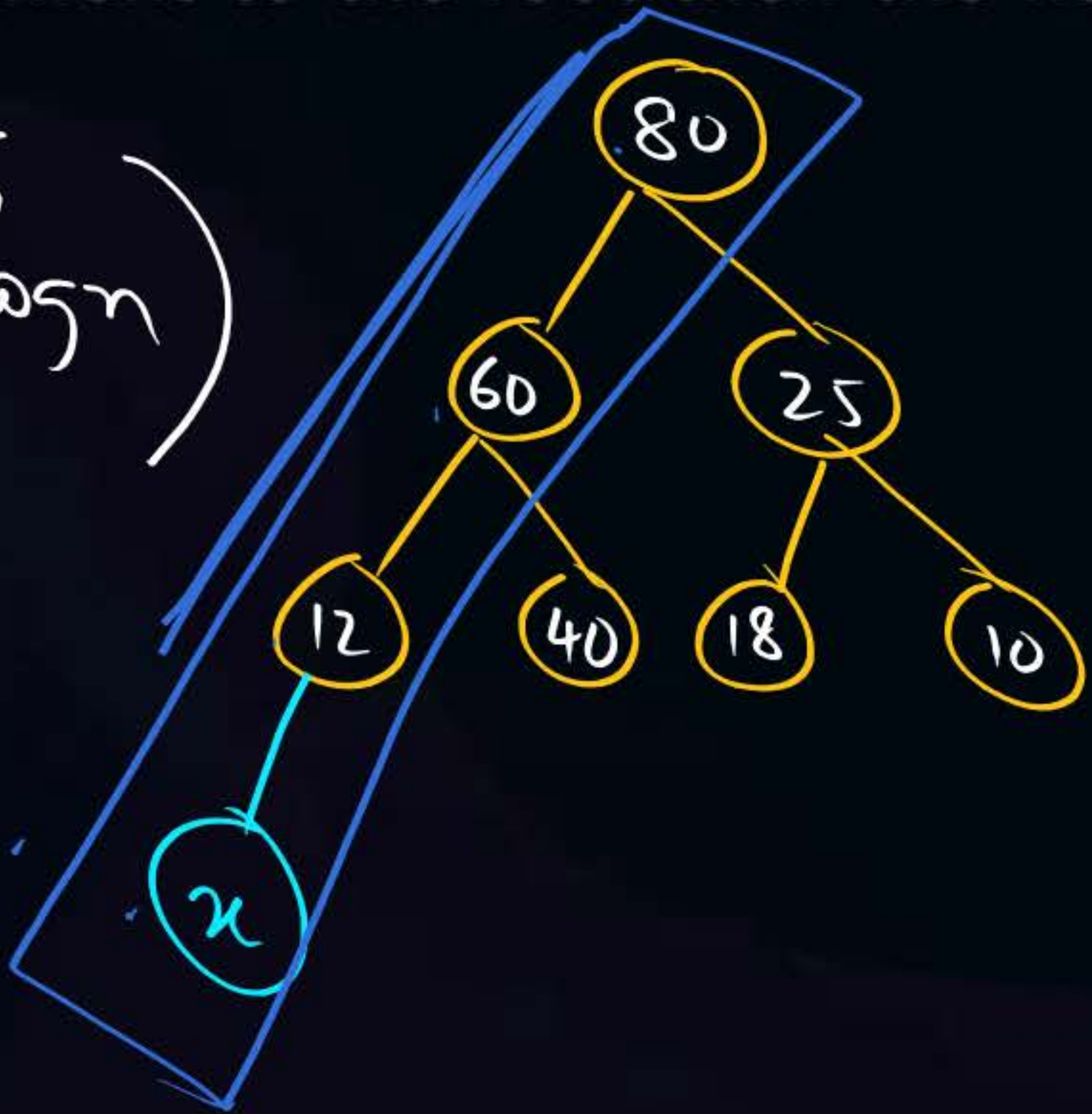
$O(\log n)$

Non-Traditional:



Q). Consider binary Heap in an Array with n elements. It is desired to insert an element into the Heap. If a binary search is performed along the path from newly inserted element to the root then the no. of comparisons made is order of Log Log n.

Log (Log n)



a) $O(n)$ ✗ b) $O(\log n)$ c) $O(1)$ ✗

d) $O\left(\frac{\log n}{\log \log n}\right)$ ✓

**

Q.) The approximate no. of elements that can be Sorted in O(logn) time using Heap Sort is ________.

Time ; 'n' elements $\Longrightarrow$ $n * \log n$

?? $\Longleftarrow$ $(\log n)$

No. of Elements will be $< n$

No. of Elements $= O(1)$

Time $= O(1)$

No. of elems $= \log n$

Time $= [\log n * \log \log n]$

$$n \Longrightarrow n * \log n$$

$$\frac{\log n}{\log\log n} \Longrightarrow \left[\frac{\log n}{\log\log n} * \log\left(\frac{\log n}{\log\log n}\right) \right]$$

$$\frac{\log n}{\log\log n}\left(\log\log n - \log\log\log n\right)$$

$$\boxed{\log n - \left(\frac{\log n}{\log\log n} * \log\log\log n\right)}$$

$$= O(\log n)$$

Topic VI. Heaps

H/W

Q.) Given $\lceil \log n \rceil$ Sorted lists each having $\lfloor n/\log n \rfloor$ elements. The time complexity to merge the given list into a single Sorted list, using Heap data structure is ______.

Algo - develop

Topic VI. Heaps

Q.) An operator delete(i) for a binary heap data structure is to be designed to delete the item in the i-th node. Assume that the heap is implemented in an array and i refers to the i-th index of the array. If the heap tree has depth d (number of edges on the path from the root to the farthest leaf), then what is the time complexity to re-fix the heap efficiently after the removal of the element?

(a) $O(1)$
(b) $O(d)$ but not $O(1)$
(c) $O(2^d)$ but not $O(d)$
(d) $O(d2^d)$ but not $O(2d)$

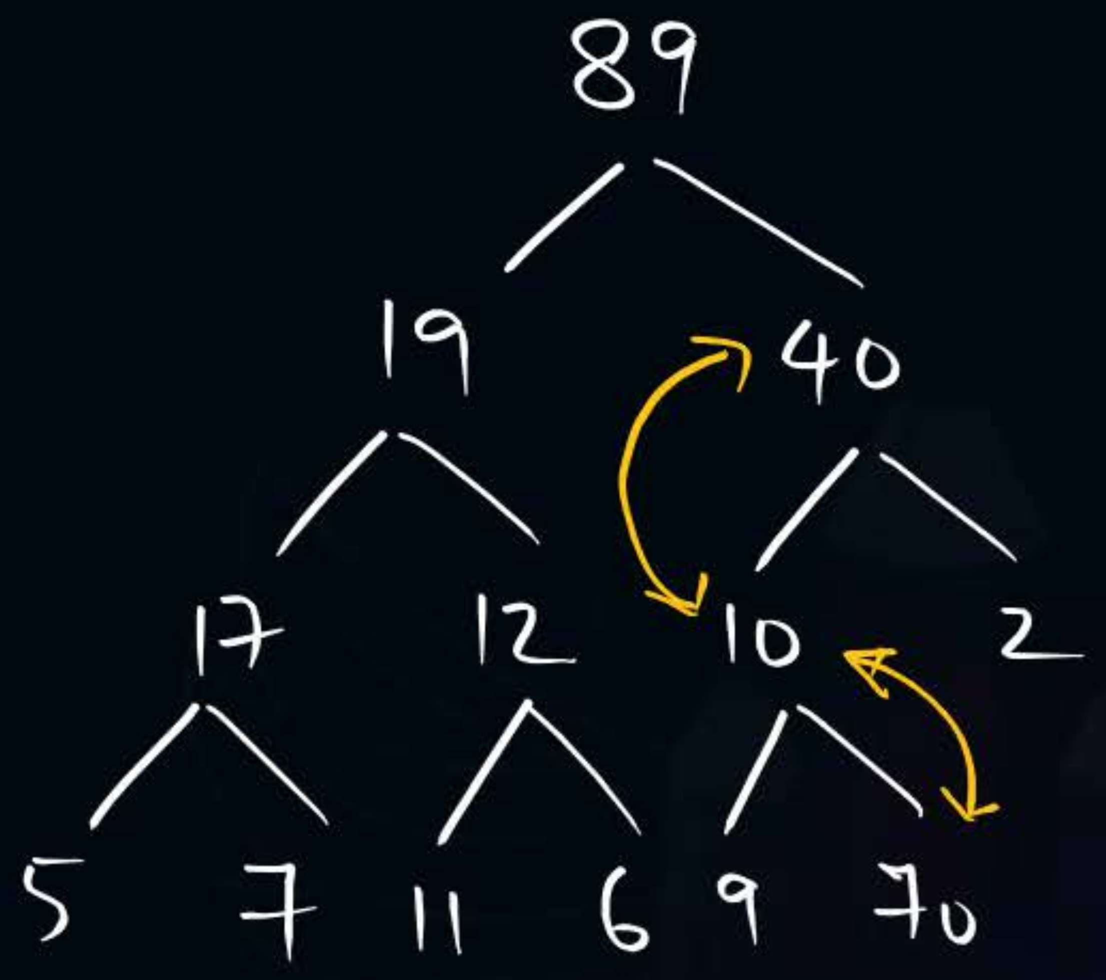
Topic VI. Heaps

Q.). The minimum number of interchanges needed to convert the array into a max-heap is

89, 19, 40, 17, 12, 10, 2, 5, 7, 11, 6, 9, 70

(a) 0 (b) 1

(c) 2 (d) 3



Heapify (Build-Heap)

Q). An array of integers of size n can be converted into a heap by adjusting the heaps rooted at each internal node of the complete binary tree starting at the node $\lfloor (n-1)/2 \rfloor$ and doing this adjustment up to the root node(root node is at index 0) in the order $\lfloor (n-1)/2 \rfloor$, $\lfloor (n-3)/2 \rfloor$,...., 0. The time required to construct a heap in this manner is

(a) O(log n)
(b) O(n)
(c) O(n log log n)
(d) O(n log n)

Topic VI. Heaps

Q). An array X of n distinct integers is interpreted as a complete binary tree. The index of the first element of the array is 0. If only the root node does not satisfy the heap property, the algorithm to convert the complete binary tree into a heap has the best asymptotic time complexity of

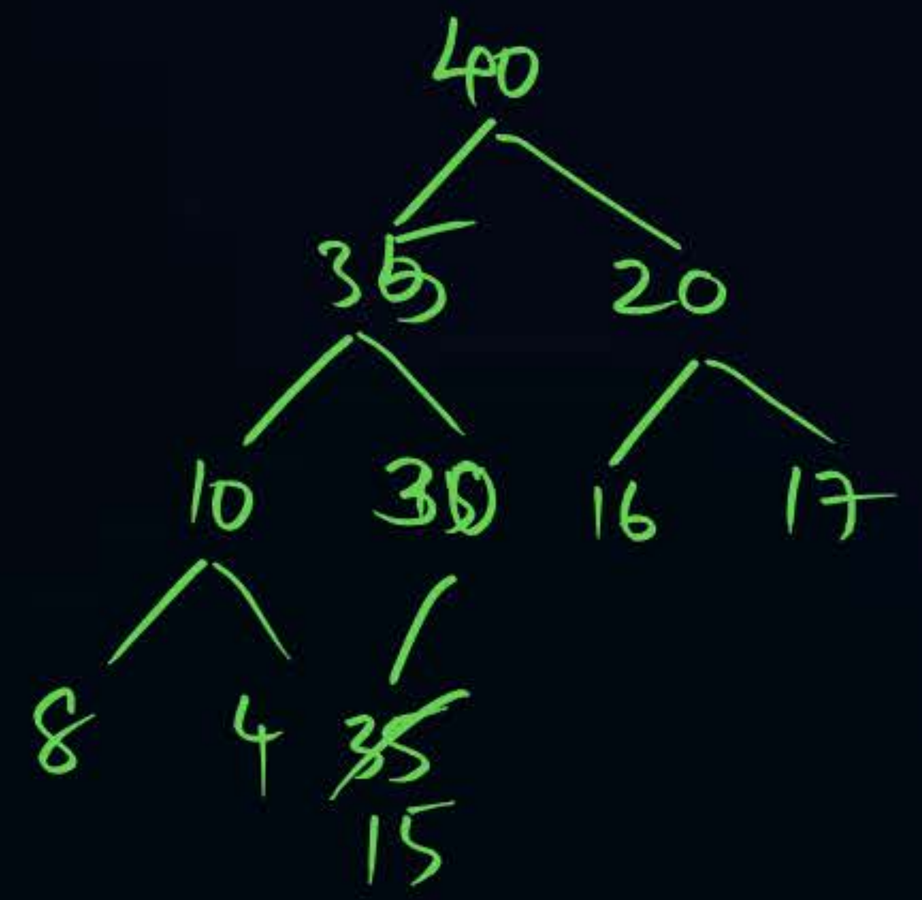
(a) O(n)
(b) O(log n)
(c) O(n log n)
(d) O(n log log n)

Topic VI. Heaps

Q) Consider a complete binary tree where the left and right subtrees of the root are max-heaps. The lower bound for the number of operations to convert the tree to a heap is

(a) $\Omega(\log n)$ (b) $\Omega(n)$
(c) $\Omega(n \log n)$ (d) $\Omega(n^2)$

Topic VI. Heaps



Q) Consider a max heap, represented by the array:
40, 30, 20, 10, 15, 16, 17, 8, 4.

Array index	1	2	3	4	5	6	7	8	9
Value	40	30	20	10	15	16	17	8	4

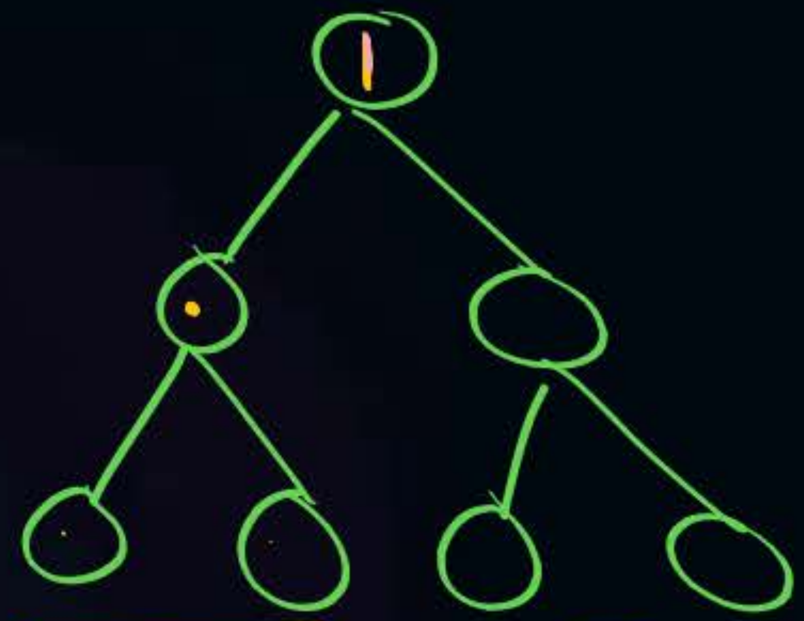
Now consider that a value 35 is inserted into this heap. After insertion, the new Heap ~~hope~~ is

(a) 40, 30, 20, 10, 15, 16, 17, 8, 4, 35
(b) 40, 35, 20, 10, 30, 16, 17, 8, 4, 15
(c) 40, 30, 20, 10, 35, 16, 17, 8, 4, 15
(d) 40, 35, 20, 10, 15, 16, 17, 8, 4, 30

Topic VI. Heaps

(D.S + Combinatorics)

Q) The number of possible min-heaps containing each value from {1, 2, 3, 4, 5, 6, 7} exactly once is 80



$$\boxed{({}^{6}C_{3})\cdot 2! * 2!} = 80$$

$$T(n) = {}^{(n-1)}C_{K} * T(K) * T(n-K-1)$$

K = No. of nodes in left subtree

$$T(7) = {}^{6}C_{3} \cdot T(3) \cdot T(3)$$

Topic VI. Heaps

Q). Consider the following statements:

I. The smallest element in a max-heap is always at a leaf node.

II. The second largest element in a max-heap is always a child of the root node.

III. A max-heap can be constructed from a binary search tree in (n) time.

IV. A binary search tree can be constructed from a max-heap in (n) time.

Which of the above statements are TRUE?

(a) I, III and IV

(b) II, III and IV

(c) I, II and III

(d) I, II and IV

Topic VI. Heaps

Q) Let H be a binary min-heap consisting of n elements implemented as an array. What is the worst case time complexity of an optimal algorithm to find the maximum element in H?

(a) Θ (log n)
(b) Θ (n log n)
(c) Θ (n)
(d) Θ(1)

THANK - YOU