



SNS COLLEGE OF ENGINEERING

Kurumbapalayam (Po), Coimbatore - 641 107

An Autonomous Institution

Accredited by NBA - AICTE and Accredited by NAAC - UGC with 'A' Grade
Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING-IOT Including CS&BCT

COURSE NAME : 19CS307- DATA STRUCTURES

II YEAR / III SEMESTER

Unit III- NON LINEAR DATA STRUCTURES - Tree

Topic 8 : Heaps



Problem



- Construct an Binary Heap having the following elements
44,33, 77,11,55,88,66



Binary Heap



- A **binary heap** is a data structure, which looks similar to a complete binary tree. Heap data structure obeys ordering properties discussed below. Generally, a Heap is represented by an array. In this chapter, we are representing a heap by *H*.
- Every level is filled except the leaf nodes (nodes without children are called leaves).
- Every node has a maximum of 2 children.
- All the nodes are as far left as possible, this means that every child is to the left of his parent.



Binary Heap –Cont..

As the elements of a heap is stored in an array,
considering the starting index as **1**,
the position of the parent node of **i^{th}** element can be found at
 $\lfloor i/2 \rfloor$.

Left child and right child of **i^{th}** node is at position **$2i$** and **$2i + 1$** .



Binary Heap–Cont..



A binary heap can be classified further as either a *max-heap* or a *min-heap* based on the ordering property.

Max-Heap

In this heap, the key value of a node is greater than or equal to the key value of the highest child.

Hence, $H[\text{Parent}(i)] \geq H[i]$

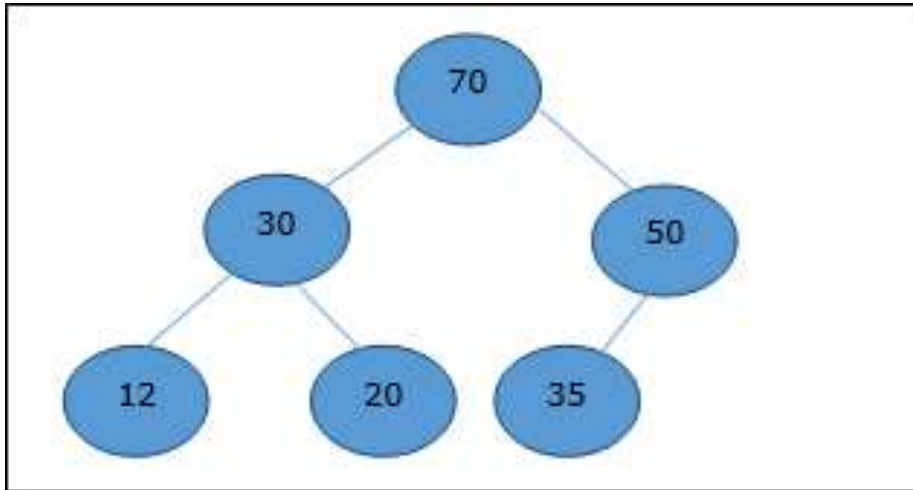
Min-Heap

In min-heap, the key value of a node is lesser than or equal to the key value of the lowest child.

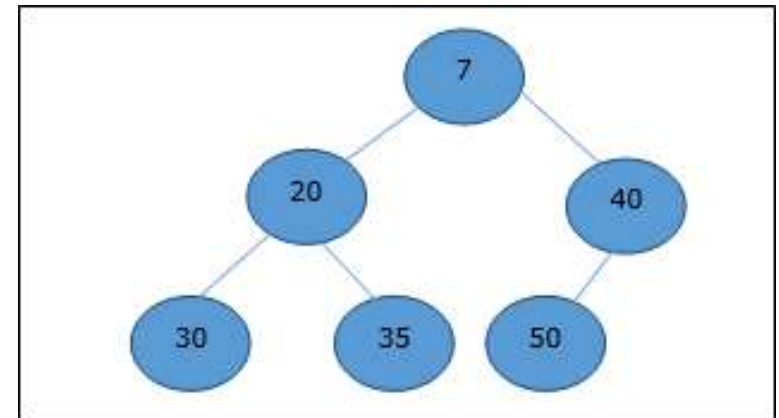
Hence, $H[\text{Parent}(i)] \leq H[i]$



Binary Heap-Cont..



Max-Heap



Min-Heap



BINARY HEAP –Cont..



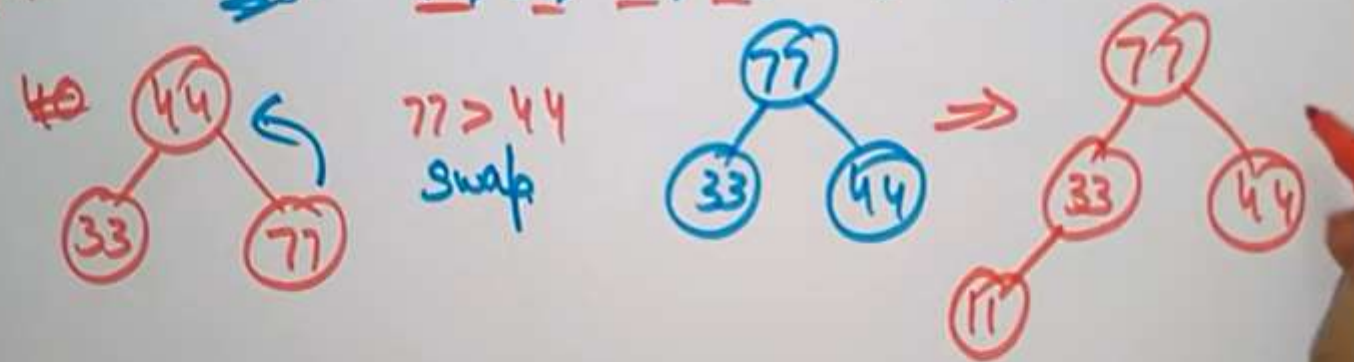
Essential Operations in Heaps

The following are the essential operations you might use when implementing a heap data structure:

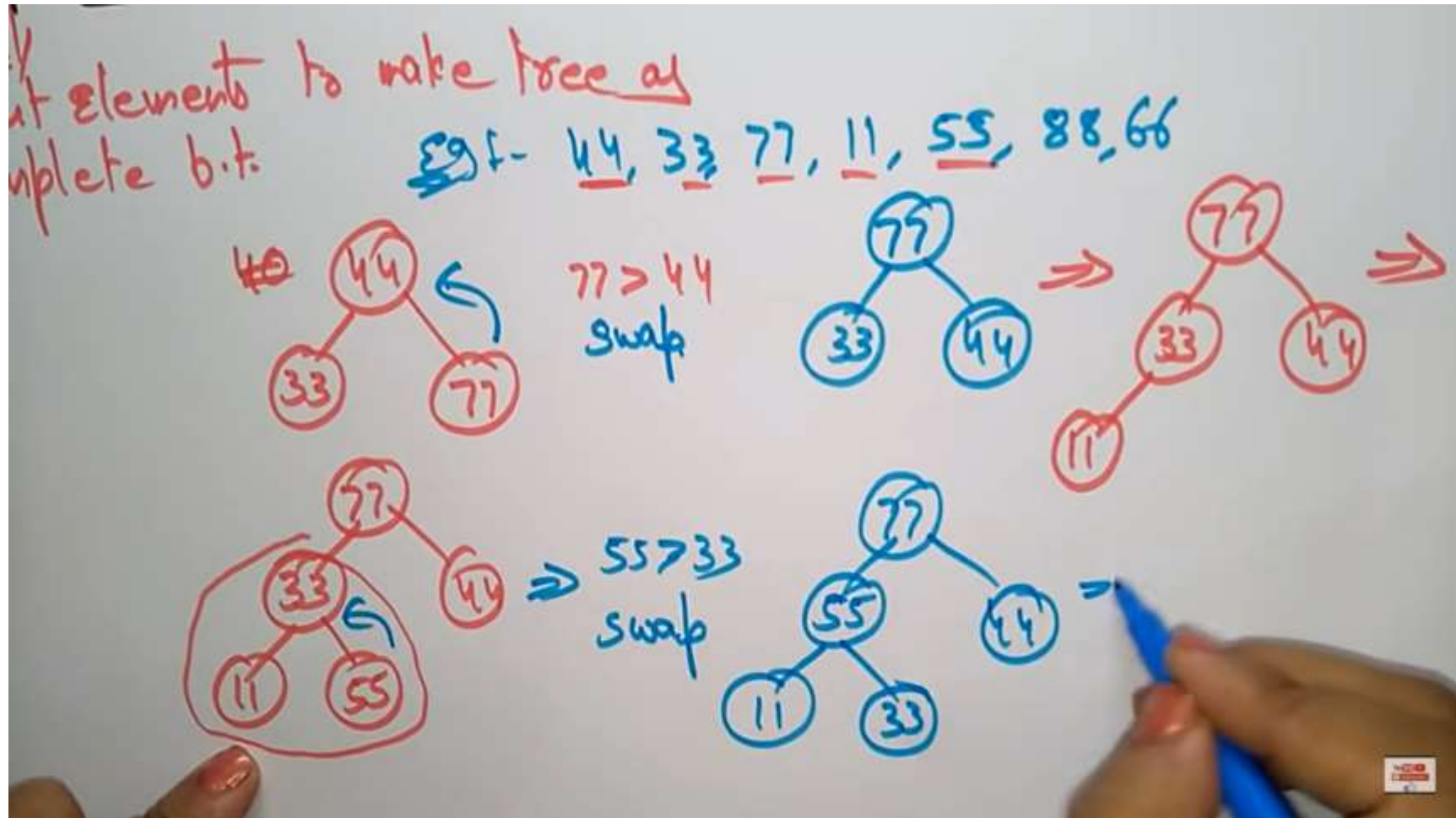
- **heapify**: rearranges the elements in the heap to maintain the heap property.
- **insert**: adds an item to a heap while maintaining its heap property.
- **delete**: removes an item in a heap.
- **extract**: returns the value of an item and then deletes it from the heap.
- **isEmpty**: boolean, returns true if boolean is empty and false if it has a node.
- **size**: returns the size of the heap.
- **getMax()**: returns the maximum value in a heap

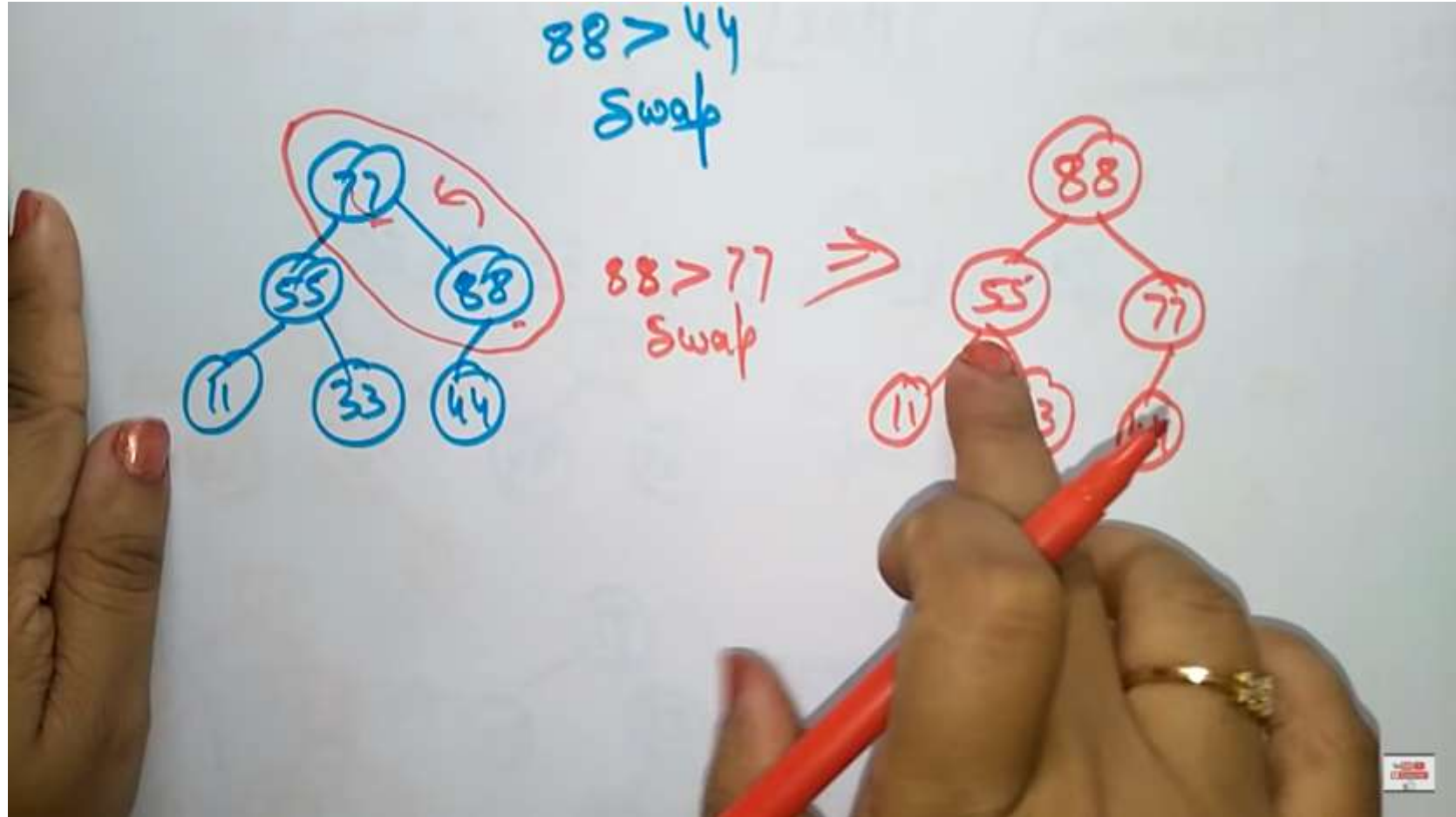
Tree (insertion) all children.

elements to make tree as
lete b.t. eg:- 44, 33, 77, 11, 55, 88, 66



```
graph TD
    A((44)) --- B((33))
    A --- C((77))
    C --- D((11))
    E((77)) --- F((33))
    E --- G((44))
```





BINARY HEAP -Cont..



```
1.int* buildmaxheap(int a[], int n)
2.{
3.int heapsize = n;
4.int j;
5.for (j = n/2; j >= 0; j--) {
6.maxheapify(a, j, heapsize);
7.} return a;
9.}
```

```
1.void maxheapify(int a[], int i, int heapsize)
2.{
3.int temp, largest, left, right, k;
4.left = (2*i+1);
5.right = ((2*i)+2);
6.if (left >= heapsize)
7.return;
8.else {
9.if (left < (heapsize) && a[left] > a[i])
10.largest = left;
11.else
12.largest = i;
13.if (right < (heapsize) && a[right] > a[largest])
14.largest = right;
15.if (largest != i) {
16.temp = a[i];
17.a[i] = a[largest];
18.a[largest] = temp;
19.maxheapify(a, largest, heapsize);}}}
```



Applications of the Heap Data Structure



1. Heap is used in the construction of **priority queues**. We can insert, delete, identify the highest priority element, or insert and extract with priority, among other things, in $O(\log N)$ time using a priority queue.

Although data structures such as BST, AVL trees, and the Red-Black tree can accomplish the same functionalities, they are more complex than heaps.

Priority queues themselves have more advanced uses, such as bandwidth control in an n/w router, prims and Dijkstra's algorithm, Huffman coding, and the BFS algorithm.

A real-Life example where a priority queue can be used:

This type of queue could be used when customers who take a short time to serve are given priority instead of those customers who arrive early. For example, customers with a small bill to pay may be given precedence at a licensing center.

2. The average waiting time for all clients in the queue is reduced due to this.
3. **Order statistics:** The Heap data structure can be used to find the kth smallest / largest element in an array quickly and effectively..
4. Heap Sort is used in systems concerned with security and embedded systems, such as the Linux Kernel.



Activity



MCQ

1. What is the location of a parent node for any arbitrary node i ?
 - a) $(i/2)$ position
 - b) $(i+1)/$ position
 - c) $\text{floor}(i/2)$ position
 - d) $\text{ceil}(i/2)$ position

2. Given an array of element 5, 7, 9, 1, 3, 10, 8, 4. Which of the following are the correct sequences of elements after inserting all the elements in a min-heap?
 - a) 1,3,4,5,7,8,9,10
 - b) 1,4,3,9,8,5,7,10
 - c) 1,3,4,5,8,7,9,10
 - d) 1,3,7,4,8,5,9,10



Advantages



- **Efficiency** – The time required to perform Heap sort increases logarithmically while other algorithms may grow exponentially slower as the number of items to sort increases. This sorting algorithm is very efficient.
- **Memory Usage** – Memory usage is minimal because apart from what is necessary to hold the initial list of items to be sorted, it needs no additional memory space to work
- **Simplicity** – It is simpler to understand than other equally efficient sorting algorithms because it does not use advanced computer science concepts such as recursion



Disadvantages



- Compared to stacks, heaps take more time to execute.
- Memory management is more complicated in heap memory, as it is used globally.
- Heaps generally take more time to compute.



Assessment 1



1. List out the advantages of binary heap

- a) _____
- b) _____
- c) _____
- d) _____

2. Identify the disadvantages of binary heap

- a) _____
- b) _____
- c) _____
- d) _____





REFERENCES



1. M. A. Weiss, “Data Structures and Algorithm Analysis in C”, Pearson Education, 8th Edition, 2007. [Unit I, II, III, IV,V]
2. A. V. Aho, J. E. Hopcroft and J. D. Ullman, “Data Structures and Algorithms”, Pearson Education, 2nd Edition, 2007 [Unit IV].
3. A.M.Tenenbaum, Y. Langsam and M. J. Augenstein, “Data Structures using C”, Pearson Education, 1st Edition, 2003.(UNIT I,II,V)
4. <https://www.youtube.com/watch?v=j6iP4lDTKyI>

THANK YOU