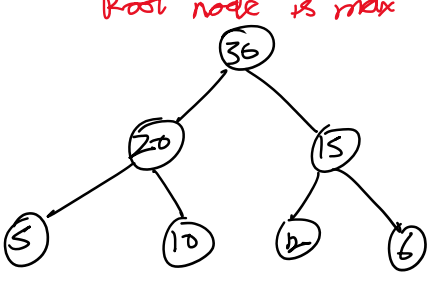
DSAA Part-9

Monday, December 21, 2020 2:56 PM

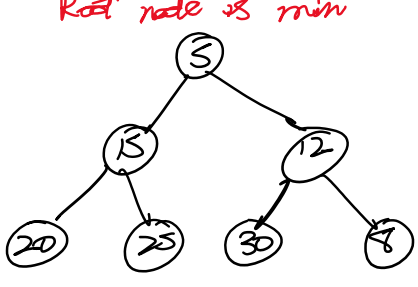
Heap.

"Complete binary tree".

Max Heap
Root node is max



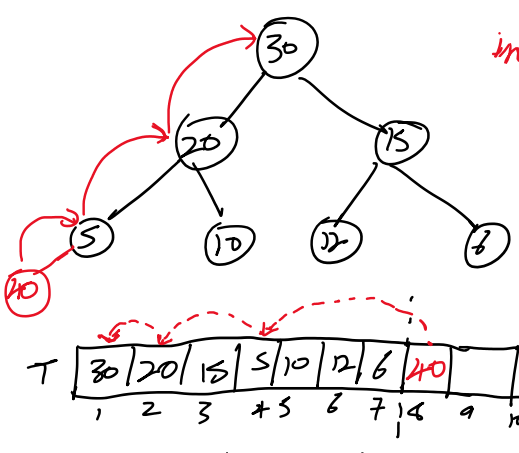
Min Heap
Root node is min



T

30	20	15	5	10	12	6

Inserting in a Heap -



→ Insert at leaf
→ Compare with parent
→ If greater then swap with parent.
→ Continue until stable.

T

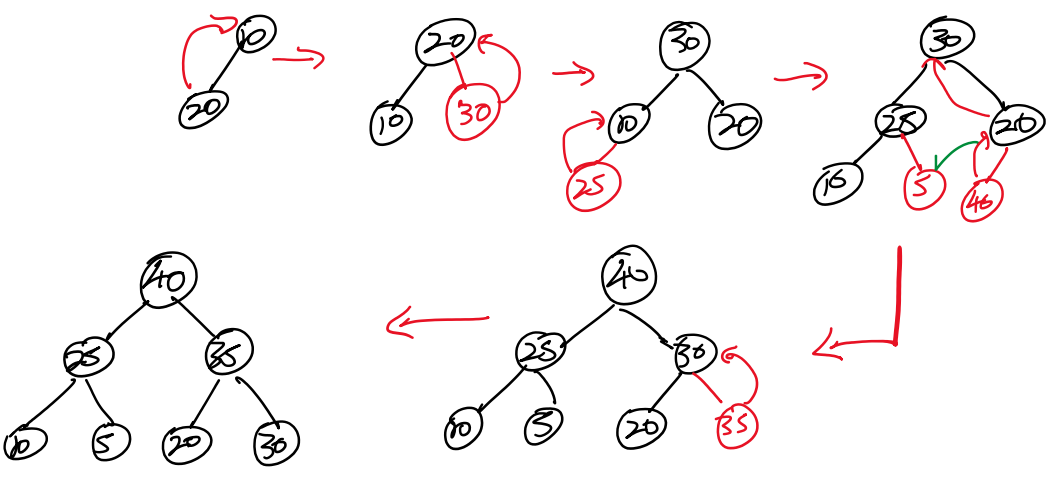
30	20	15	5	10	12	6	40		
1	2	3	4	5	6	7	8	9	10

```
void Insert(int A[], int n)
{
    int temp, i=n;
    temp = A[n];
    while(i>1 && temp > A[i/2])
    {
        A[i] = A[i/2];
        i = i/2;
    }
    A[i] = temp;
}
```

O(log n)

Creating a Heap -

keys - 10, 20, 30, 25, 5, 40, 35.



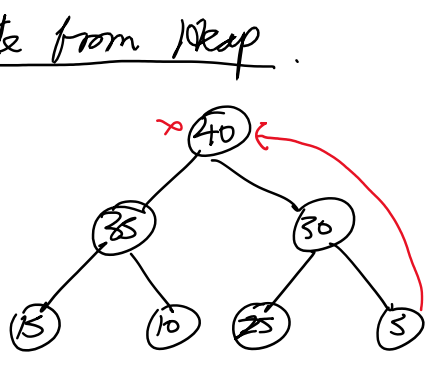

```
void create()
{
    int A[] = {0, 10, 20, 30, 25, 5, 40, 35};
    int i;

    for(i=2; i<=7; i++)
    {
        Insert(A, i);
    }
}
```

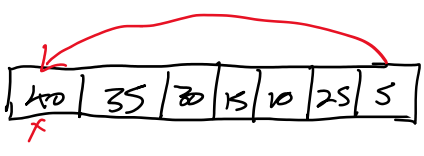
log n × n

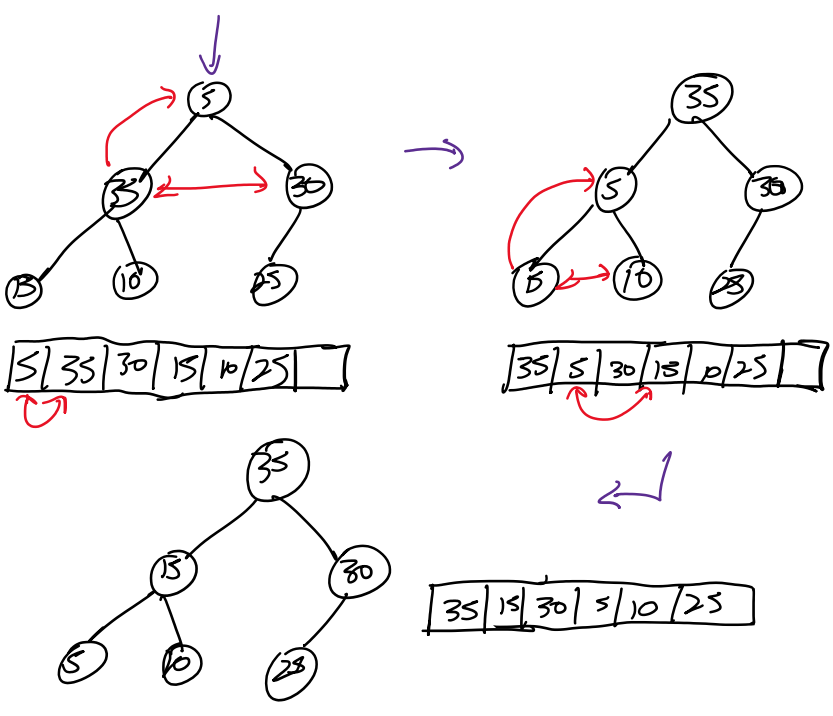
O(n log n)

Delete from Heap.



Can only delete root element.






```
void Delete(int A[], int n)
{
    int x, i, j;
    x = A[n];
    A[1] = A[n];
    i=1; j=2*i;
    while(j<n-1)
    {
        if(A[j+1] > A[j])
            j = j+1;
        if(A[i] < A[j])
        {
            swap(A[i], A[j]);
            i=j;
            j=2*j;
        }
        else
            break;
    }
    A[n]=x;
}
```

log n

Heap sort

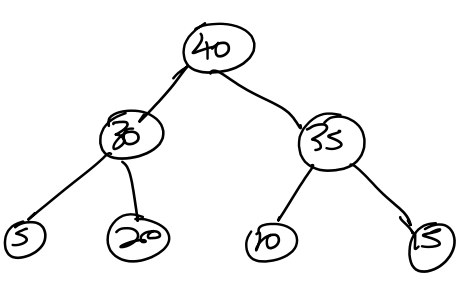
Heap Sort -
→ Create a heap.
→ Delete 'n' elements one-by-one. — n x log n
→ Insert deleted elements in vacant spaces.

Resulting array will be sorted.

O(n log n).

Heapify.

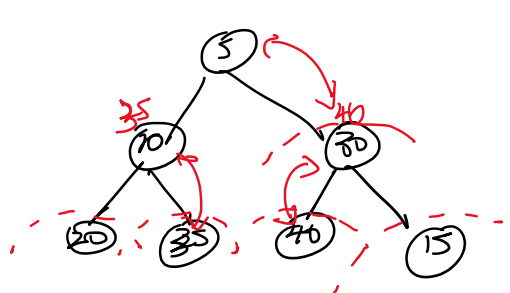
Create Heap.



5	10	30	20	35	40	15

O(n log n) time

Heapify.



→ Compare node with children.
→ Swap if smaller.

5	10	30	20	35	40	15

O(n) time

Faster!