•

2 5 2 6 3

cost to connect 2 ropes → sum of length of 2 ropes

Find minimum cost to connect all the ropes, connecting two at a time.

cost = 0

(1)

2 5 7 + = 7

2 7 9 + = 9

6 9 15 + = 15

15 3 18 + = 18

49

(2)

2 2 4 + = 4

4 3 7 + = 7

5 6 11 + = 11

11 7 18 + = 18

40

$x < y < z$

$x \,\&\, y$ z

$x+y+$ $\qquad x+z+$ $\qquad y+z+$

$x+y+z$ $\qquad x+z+y$ $\qquad x+z+y$

insertion sort

2 2 3 5 6 7

+=4

4

+=7

7

5 6 8 9 11 12 13 17 50

11

17

11

17

insertion sort — $O(n^2)$

DS → insert(x) — log N

→ extractmin() — log N

} N log N

1) Build your heap — $O(N)$

2) loop
```
{
    x = extractMin();        cost += (x+y);
    y = extractMin();
        insert(x+y);
}
```

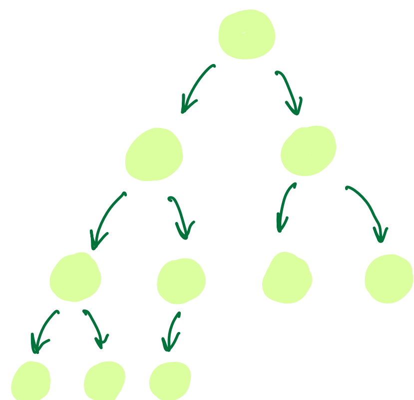
Heap (Binary Heap) — Binary Tree

Structure

CBT — complete BT

Balanced

every level completely child
last level should be filled from L to R

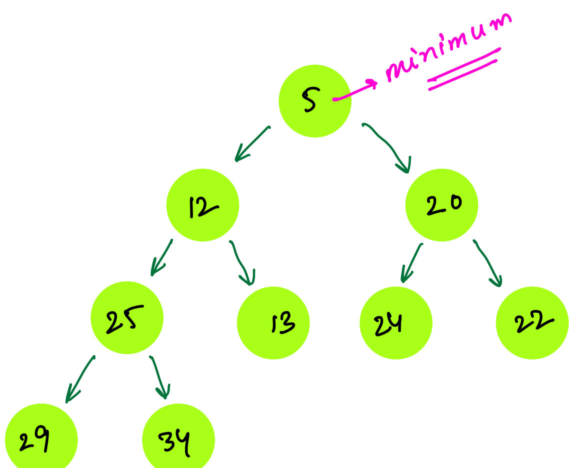


order of elements

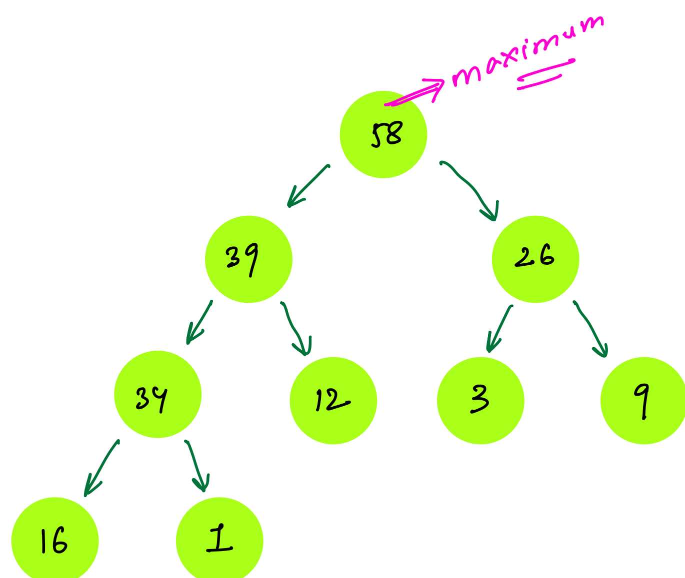
for every Node

max heap: 1) node's data greater or equal to its children

min heap: 2) node's data smaller or equal than it's childen



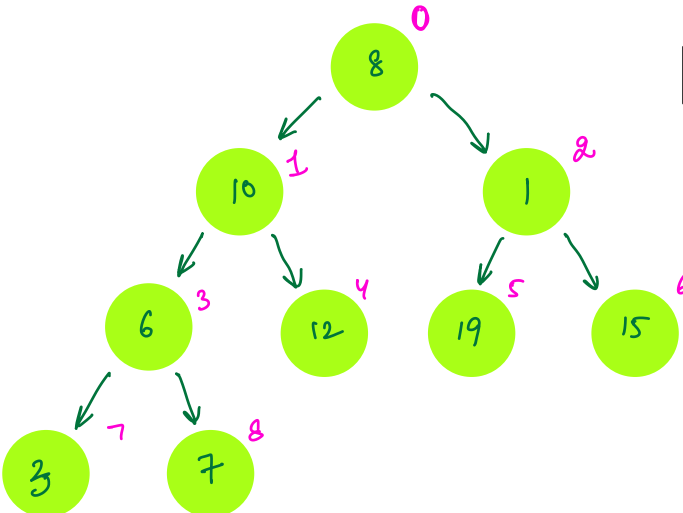
Min heap



max heap

No relationship b/w siblings.

Array implementation of trees



8	10	1	6	12	19	15	3	7

root = arr[0]

$0 \xrightarrow{2*0+1} 1$

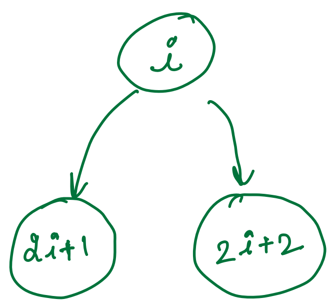
$0 \xrightarrow{2*0+2} 2$

$1 \xrightarrow{2*1+1} 3$

$1 \xrightarrow{2*1+2} 4$

$2 \xrightarrow{2*2+1} 5$

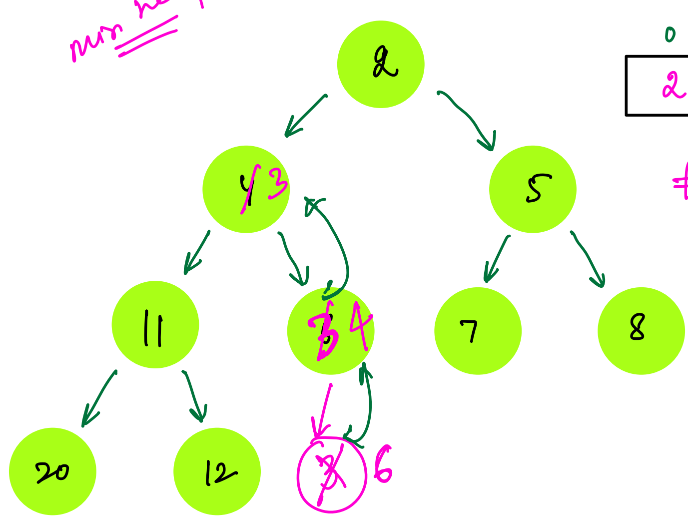
$2 \xrightarrow{2*2+2} 6$



$i \xrightarrow{\text{parent}} \frac{(i-1)}{2}$

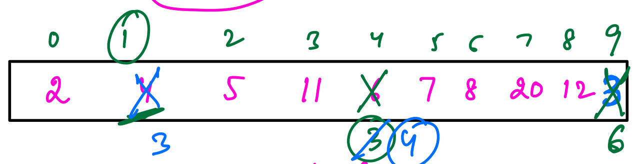
Insert in Heap

min heap



T.C: $\log N$
S.C: $O(1)$

Insert 3

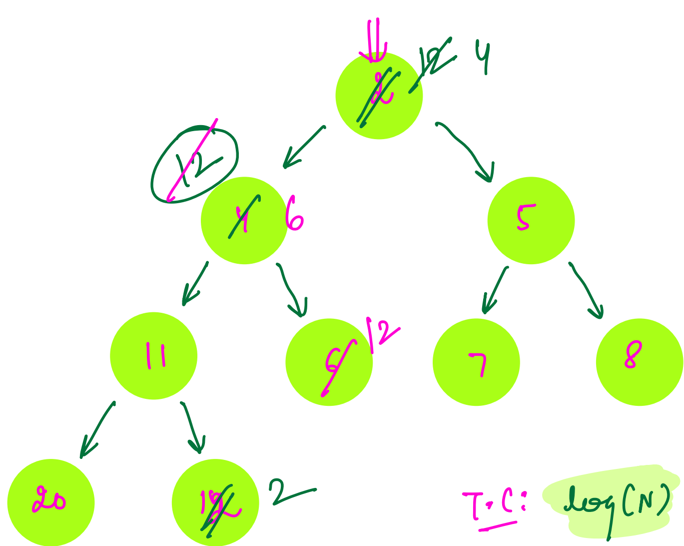


maintain the struct

heap.insert(3);

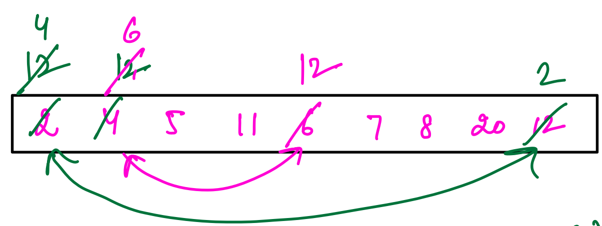
i	parent idx	
9	4	ar[4] > ar[9]
4	$(4-1)/2 = 1$	ar[1] > ar[4] swap
1	$(1-1)/2 = 0$	ar[0] < ar[1]

extract min



T.C: $\log(N)$

getMin() { return arr[0]; }



swap(heap[0], heap[size-1]);

i	children	
0	1 2	4 5 min(4,5) = 4 swap(12,4)
1	3 4	11 6 min(11,6)
4		

update/ search/ delete - $O(N)$

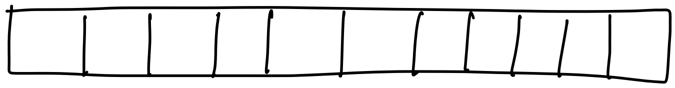
Build Heap

↓

using insert

↓ N elements

$N \log N$

input: 

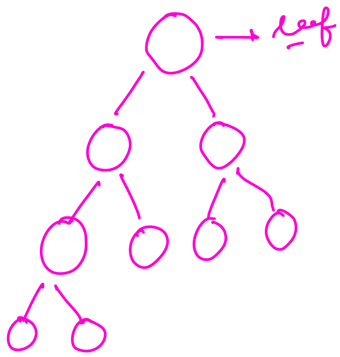
heap: 

S.C: $O(N)$

Build already given the data - $O(N)$, $O(1)$

T.C S.C

N	leaves
1	1
2	1
3	2
4	2
5	3
6	3
7	4
8	4
9	5



CBT

N % $\frac{(N+1)}{2}$

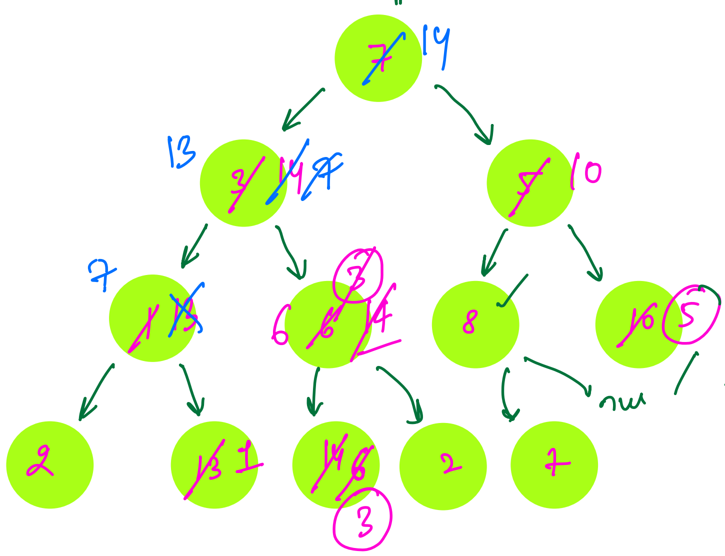
$N/2 - 1$ non-leaf

0	1	2	3	4	5	6	7	8	9	10	11
7	3	5	1	6	8	10	2	13	14	2	7

$N = 12$

leave = 6

maxheap



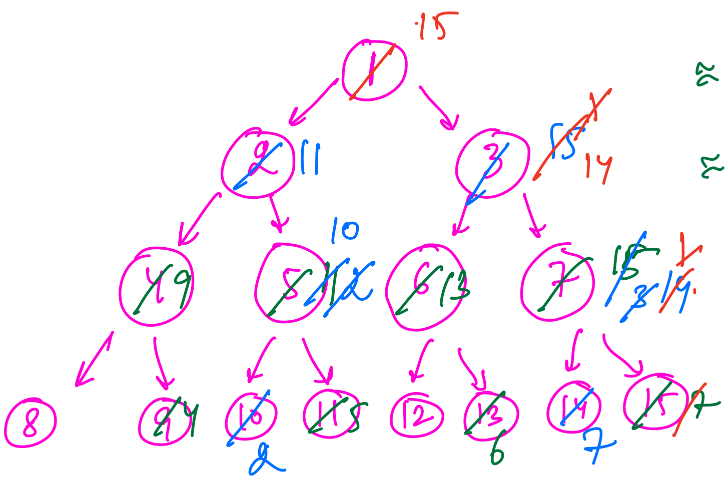
$i = 5$

$i = 4$ 6 ✗ swap with 14

$i = 3$ 1 ✗ swap with 13

$i = 2$

7 elements



for every node

≈ $N/16$ 3 swap

≈ $N/8$ 2 swaps

≈ $N/4$ 1 swap

≈ $N/2$ 0 swaps

T.C: $\frac{N}{2} * 0 + \frac{N}{4} * 1 + \frac{N}{8} * 2 + \frac{N}{16} * 3 \ldots$

$= \frac{N}{2}\left(\frac{1}{2} + \frac{2}{4} + \frac{3}{8} + \frac{4}{16} + \ldots\right) \to$ AGP

$\to 2$

T.C: $\frac{N}{2} \cancel{* 2}$

$S = \frac{1}{2} + \frac{2}{4} + \frac{3}{8} + \frac{4}{16} + \ldots$

$\frac{S}{2} = \frac{1}{4} + \frac{2}{8} + \frac{3}{16} + \frac{4}{32} \ldots$

$-$

$\frac{S}{2} = \frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \frac{1}{16} + \frac{1}{32} \ldots$

$S_\infty = \frac{a}{1-r}$

$\frac{S}{2} = 1$

$\frac{\frac{1}{2}}{1-\frac{1}{2}} = 1$

$S = 2$

```
Heapify( heap[], int i)
{
  while( 2i+1 < size)
  { int x = max( heap[i], heap[2i+1], heap[2i+2]);

      if( heap[i] == x)
      {   return; }
      else if( heap [2i+1] == x)
      {       swap( heap[i], heap[2i+1]);
                  i = 2i+1;   }
      else {  swap( heap[i]; heap[2i+2]); i=2i+2;
}   }
```

Q

N chocolate bags, each having A[i] chocolates.

Kid ⇒ select the bag with max no of chocolates & eats it.

Magician ⇒ Fill the bag again by A[i]/2 chocolates.

Find no of chocolates kid can eat in k steps.

K=4

max

A = [~~16~~ , 3 , 15 , ~~8~~ , 4]   k=5

5   ~~7~~ 3   4

maxheap

cout += 15
+= 10
+= 8
+= 7

40

1) Build heap
2) for k steps
   x = extract max
   ans += x
   insert (x/2).

Doubts

7	3	5	1	6	8	10	2	13	14	2	7

i=0

~~7~~	~~3~~	5									
3	7