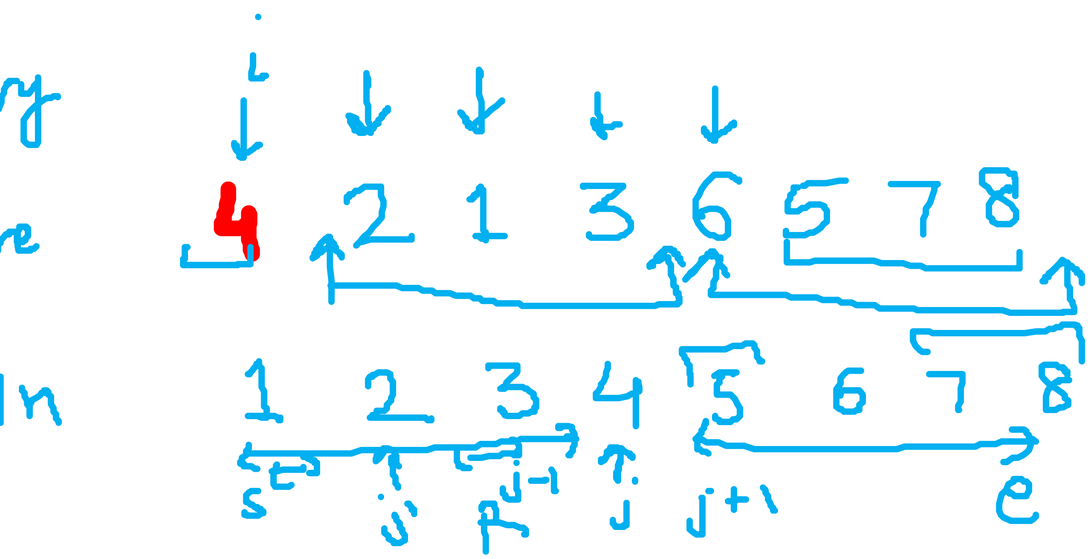
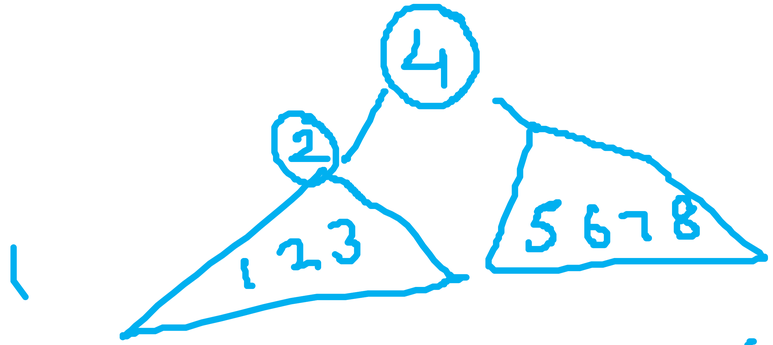
Binary

Pre: 4 2 1 3 6 5 7 8

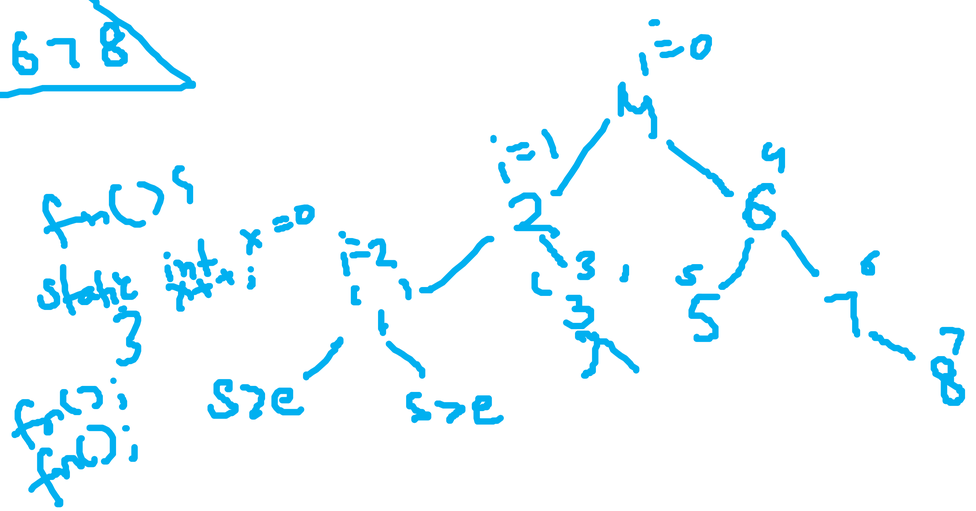
In: 1 2 3 4 5 6 7 8



Uniq
↳ Pre + In
↳ In + Post



Root Left Right



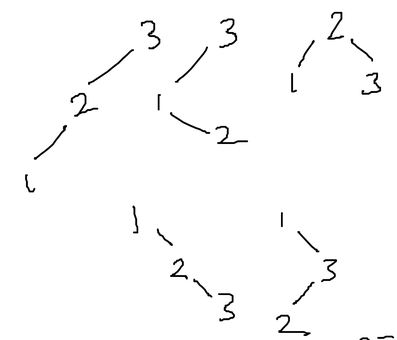
Next Q?

Find No of

- ✓ BST
- – BT's
- ✓ Unlabelled BT's

Using N Nodes. (1 .... N)

N=3



5 BST's

BT = $\frac{2n_{C_n}}{(n+1)!}$

N elements

2n edges

N-1 edges choose

$2n C_{N-1} * (n-1)!$

n-1 edges

$(n-1)!$

$2N(2N-1)(2N-2)\cdots(2N-N+1)$

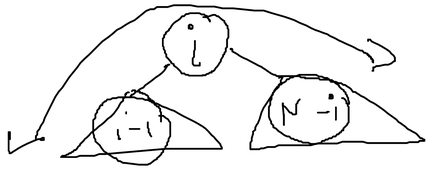
$= \frac{2N(\cdots)(N+1)N!}{N!}$

$2n C_N$

N elements

N choices

1, 2, 3, 4, 5, ... i ... N



$f(N)$ = No of BST's using N Nodes

$x = f(i)$ = ✓ i Nodes

$y = f(N-1)$ = ✓ N-i Nodes

$f(2) = \sum_{i=1}^{2} f(i) * f(N-i)$

$= 1*1 + 1*1 = \boxed{2}$

X ways and y ways

$x_{C_1} \cdot y_{C_1}$

$= xy$

$f(N) = \sum_{i=1}^{N} f(i-1) * f(N-i)$

Recursion

[optimize: DP]

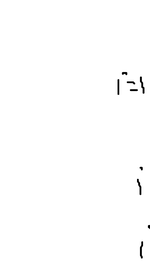
N=3

i=1 $f(0) * f(2)$ $1 * 2 =$ 2

i=2 $f(1) * f(1)$ $= 1$

i=3 $f(2) * f(0)$ $2 * 1 = 2$

+ + = 5 ways



$f(0) = 1$

$f(1) = 1$

$f(2) = 2$

$f(3) = 5$

⋮

14

42,

Catalan Series

$T_n = \frac{2n C_n}{n+1}$

|| # BST's

Catalan No $= \frac{2N C_N}{(N+1)} = \frac{6C_3}{4} = \frac{6.5.4}{1.2.3.4} = \boxed{5}$

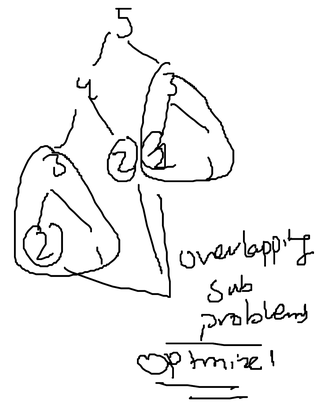
Labelled Binary Tree - $\frac{2N C_N}{N+1} * N!$

Unlabelled Tree

Applications of Catalan Number

```
int f(n){
    n==0
      return 1;

int ans=0;
for(i=1; i<=n; i++)
  { ans += f(n-i)*f(i-1)}
return ans;
}
```



a[3] = -1

Heaps — Max Heap / Min Heap

(Priority Queue)

- → Ins
- → Del
- → Search

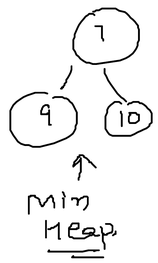
↳ Insert O(Log N)

↳ Get Min / Max O(1)

↳ Remove Min / Max O(Log N)

N	N Log N	To Sort
K	K Log N	

Heap ?

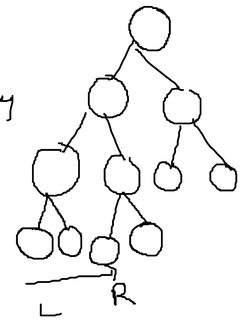


Min Heap

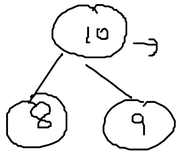
Parent ↳ Data Structure

Children ↳ Complete Binary Tree for all nodes

All levels are completely filled except last level which may be partially filled but must be in Left to Right order.

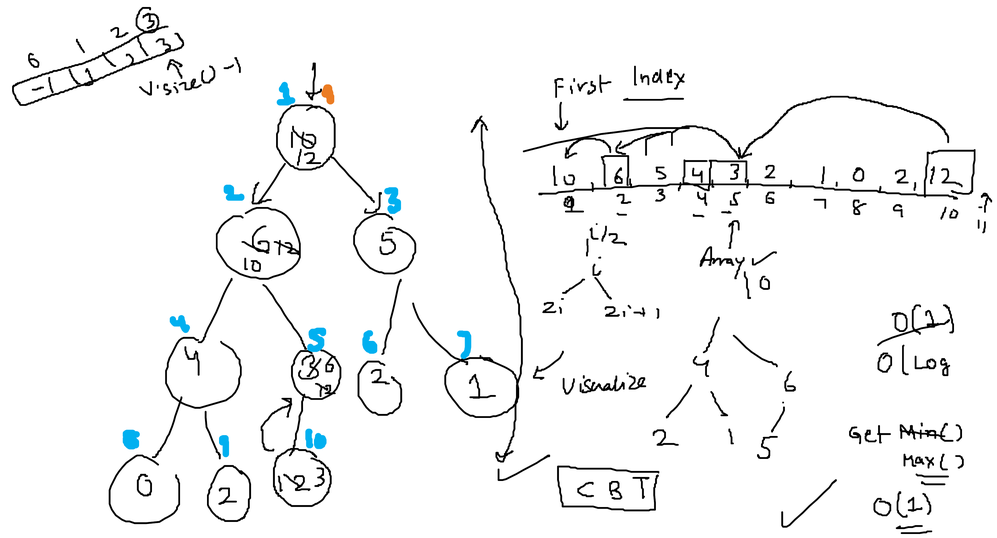


Maxheap



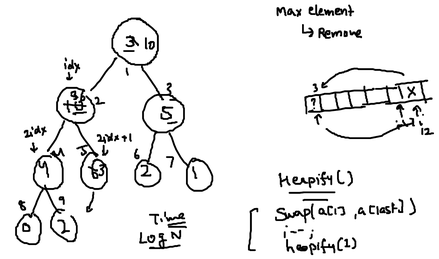
→ Parent ? Children

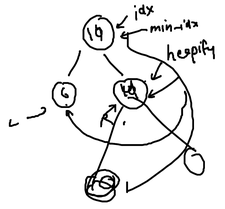
↳ Heap Order Property



Next

Remove Min()/Max()



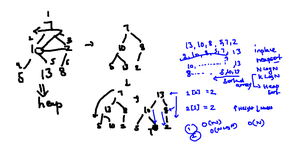


→ N Log N + K Log N
build

ARRAY → heap (build) = O(N) + K Log N

How to Build a heap from Array in O(N) time.

7, 10, 2, 5, 13, 8



[for (i = n/2; i >= 1; i--)
↳ heapify (i);]