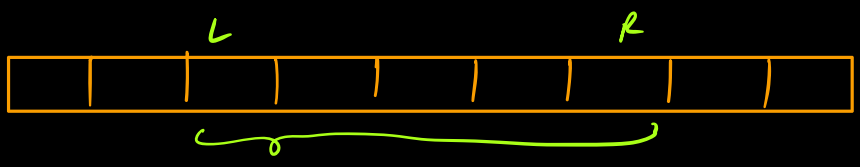
Suryansh Gupta 2021 grad from IIEST Shibpur

Research - SDE at Microsoft Research

Ex. SWE - III (L9) at Google

Intern at Directi & Microsoft IDC

Quiz 1 :- Prefix Sum



SUM (L - R) => Prefix[R] - Prefix[L-1]

0 to R 0 to L-1

if L == 0, Prefix[R]

Quiz 2 :- Kadanes Algo

2	5	-6	1	8	-10	-1	-5	20	-19	20

sum = 2 7 1 2 10 0 -1 0 -5 0 20 1 21

best = 2 7 10 20 21

0	1	2	3	4	5	6
2	5	-6	1	8	-11	20

Sum = 0 2 7 1 2 10 17 0 20

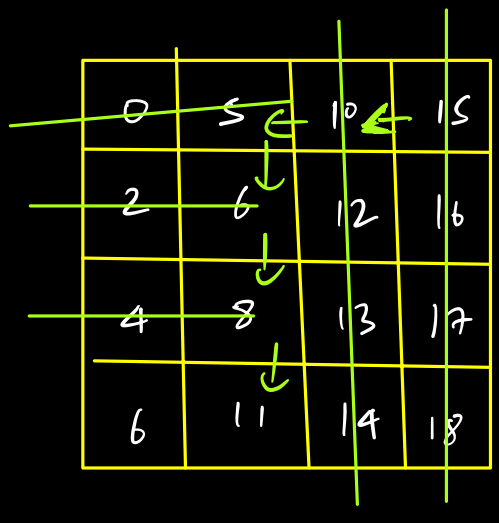
l = 0 6

but = 2 7 10 20

but & l = 0 0 0 6

but & K = 0 1 7 6

Quiz 3:-



Search for 9

TC : O(N+M)

Quiz 4:-

x = 39

x = x >> 2

x = x | 5

32 bit

```
    00000000000000000000000000100111  = 39
    00000000000000000000000000001001  = 9
    00000000000000000000000000000101
    --------------------------------
    00000000000000000000000000001101  = 13
```

Quiz 5:-

```
void fun(int x) {
1) if (x == 0)
       return
2) Print (x)
3) fun(x - 1)
}
```

5 4 3 2 1

fun(5)

~~1) 2) 3)~~

5 4

fun(4)

~~1) 2) 3)~~

fun(3)

⋮

Quiz 6:-

```
int fun(x) {
    if x == 0
        return 1
    int ans = 2 * fun(x - 1)
```

```
{ return and
```

```
print( fun(4) )
```

$$fun(x) = 2 * fun(x-1)$$

$$= 2 * 2 * fun(x-2)$$

$$= 2 * 2 * 2 + 2 * fun(x-3)$$

$$fun(x) = \underbrace{2 * 2 * 2 * \dots * 2}_{K \text{ times}} \; fun(\underbrace{x-K}_{0})$$

$$x - K = 0$$

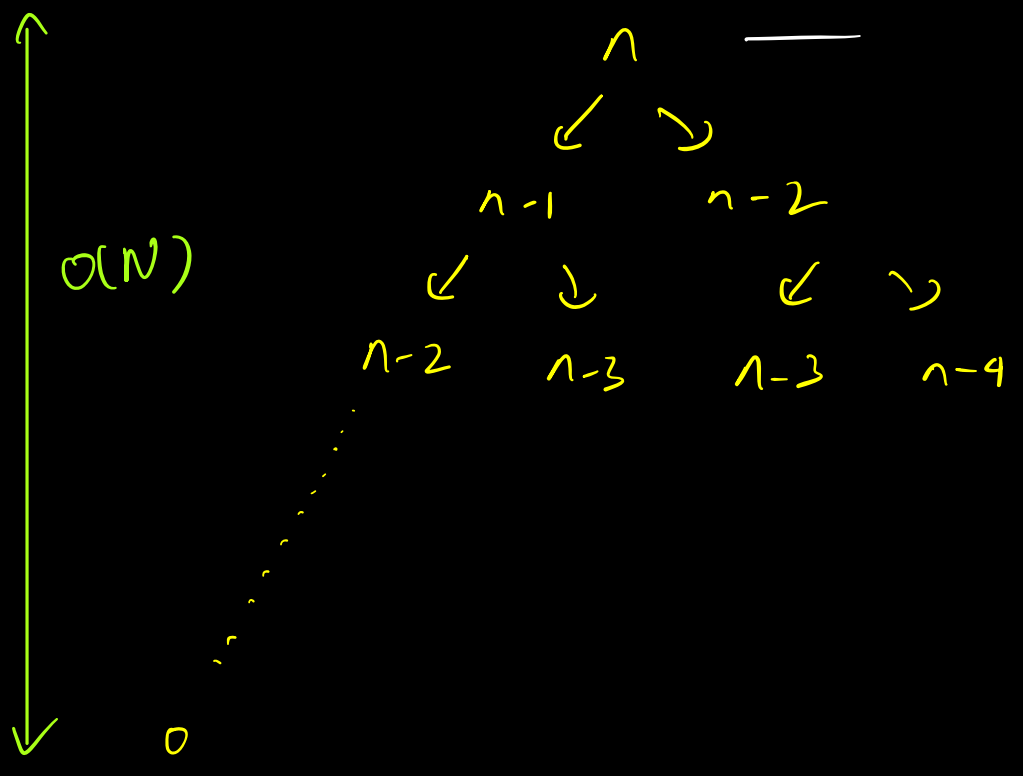
$$K = x$$

$$fun(x) = \underbrace{2 * 2 * 2 \dots * 2}_{x \text{ times}} * 1$$

$$= 2^x$$

$$= 2^4 = 16$$

Quiz 7:- Fibonacci Recursive Code



0 => 1
1 => 3
2 => 7
3 => 15
x => $2^{x+1}-1$

TC: O(No. of nodes / fun calls * Time take in each function call)

O(2^N) * O(1)

TC: O(2^N)

Ques 8:- GCD (10, 55)

gcd(a, b) = gcd(b , a%b)

$$gcd(10, 55) = gcd(55, 10\%55)$$

$$= gcd(55, 10)$$

$$= gcd(10, 55\%10)$$

$$= gcd(10, 5)$$

$$= gcd(5, 10\%5)$$

$$= gcd(5, 0) = 5$$

$$gcd(a, 0) = a$$

$$gcd(a, b) = gcd(b, a\%b)$$

Quiz 1: Check if a single no. is prime or not?

Basic: Divisors lie in pairs

Go from 2 to $\sqrt{N}$ and check if any no. divides the no. N then it's not prime

TC : $O(\sqrt{N})$

Fancy Way : Sieve ⇒ Gives us all the primes from 1 to N

TC : $O(N \log \log N)$

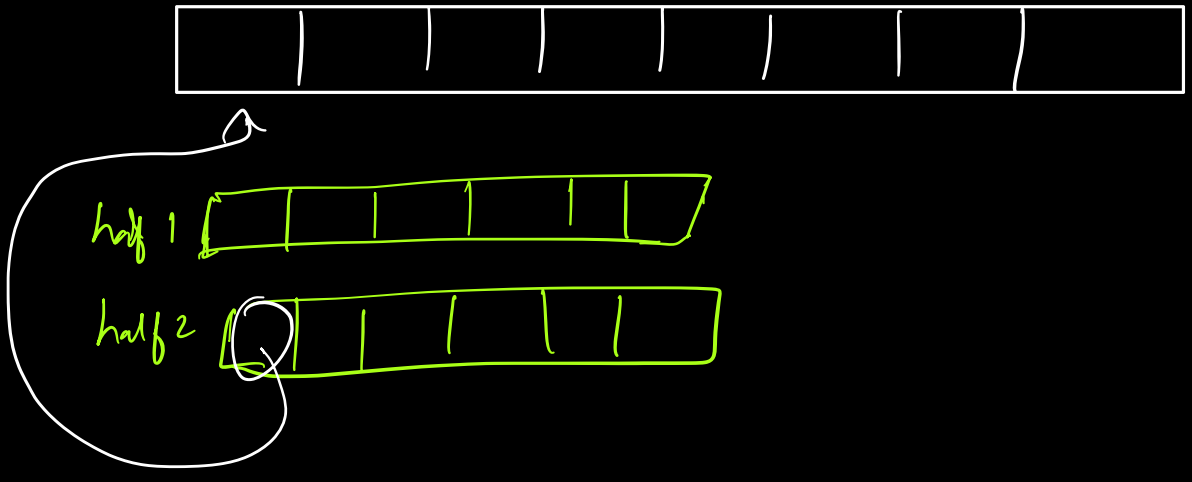
Ques 10 :- arr = ['a' 'c' 'a' 'l' 'e' 's']

Sort (arr) ⇒ 'a' 'c' 'e' 'l' 's' 'a'

Subarray ⇒ 2 Point Problem

Subarray Sum [L-R] ⇒ Prefix [R] - Prefix [L-1]

	TC	SC
Quick Sort	$O(N \log N)$	$O(\log N)$ } Recursion
Bubble Sort	$O(N^2)$	$O(1)$
Merge Sort	$O(N \log N)$	$O(N)$
Count Sort	$O(N)$	$O(\text{Range})$



Ques :- Walls P1 = 10 days

P2 = 15 days

P3 = 8 days

Minimize the maximum

Quiz 21 :- 1 5 8 11 16 18 25 K = 9

l r

A[r] - A[l] = 25 - 18 = 7

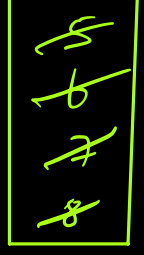
= 25 - 16 = 9

Monotonic : Increasing or Decreasing

Next Greater on right :

0	1	2	3	4	5	6	7	8
1	10	5	15	6	0	9	3	8
↓	↓	↓	↓	↓	↓	↓	↓	↓
1	3	3	9	6	6	9	8	9

0
1
~~2~~
3
~~4~~



Ques 27: Arr = [3 1 7 2 6]

Ans = [3 7 7 6]

Approach 1:- Max Heap with index

TC: O(N log N)

SC: O(N)

1 2 3 4 5 6 7 9

Approach 2:- Dequeue / Monotonic queue

Arr = [3 1 7 2 6]

Max ↳ ~~3~~ ~~1~~ ~~7~~ 2 6 ↳ Push

TC: O(N)

SC: O(K)

Ans: [3 7 7 6]

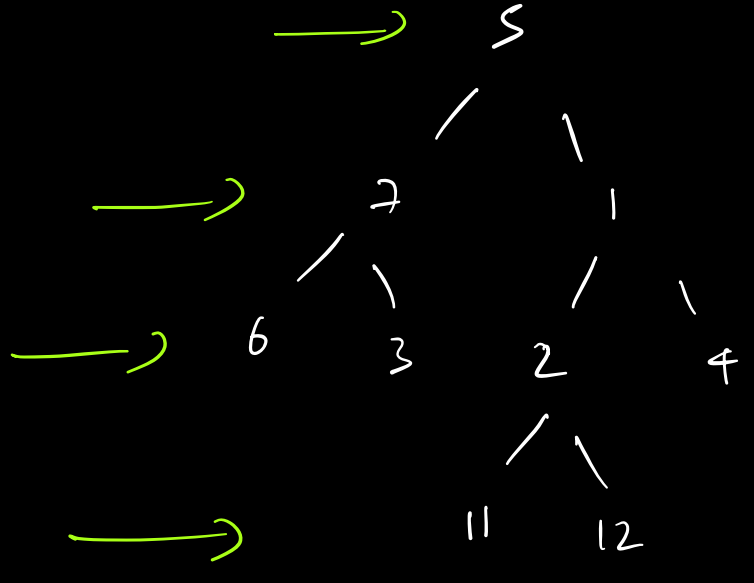
Trees

Inorder + Pre Order

Inorder + Post Order

Level Order => Left to Right

Top to Bottom



Level by Level

Each Level

=

Left to Right

Search in Binary Search Tree

TC: $O(H)$

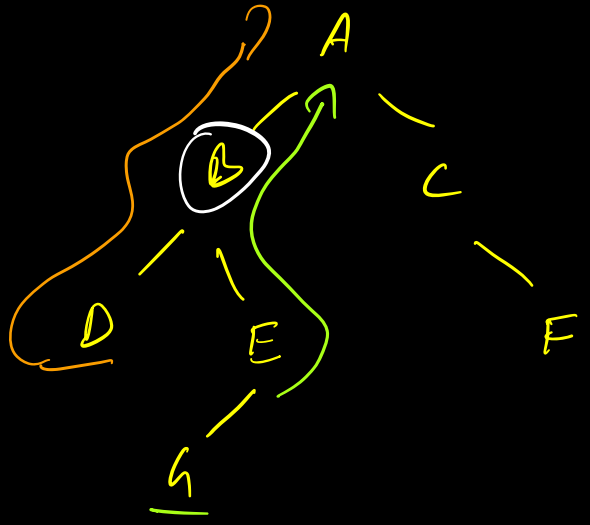
SC: $O(H)$

H: Height of the tree

In case of Balanced BST

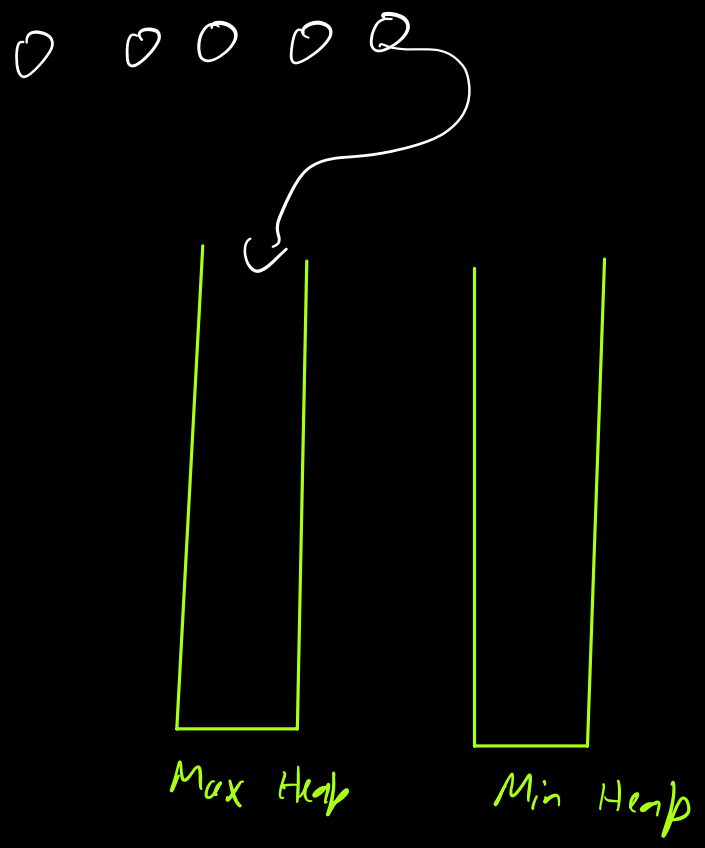
$H = \log(N)$ N: No. of nodes

Ques: LCA of D & G



G: E B A

D: D B A



Stair's Case :

$$ways(N) = \underbrace{ways(N-1)}_{\text{1 step move}} + \underbrace{ways(N-2)}_{\text{2 step move}}$$

0 - 1 Knapsack $\rightarrow$ DP[N][Sum]

Unbounded Knapsack $\rightarrow$ DP[Sum]

$$\frac{N^2 - N}{2} = \frac{N(N-1)}{2}$$

$^nC_2 = \frac{N(N-1)}{2}$

$$\frac{\text{All to All} - \text{Self Links}}{2 \text{ (For double counting)}}$$

Shortest Path in Graph

No weight $\Rightarrow$ BFS

0/x weight $\Rightarrow$ 0-1 BFS

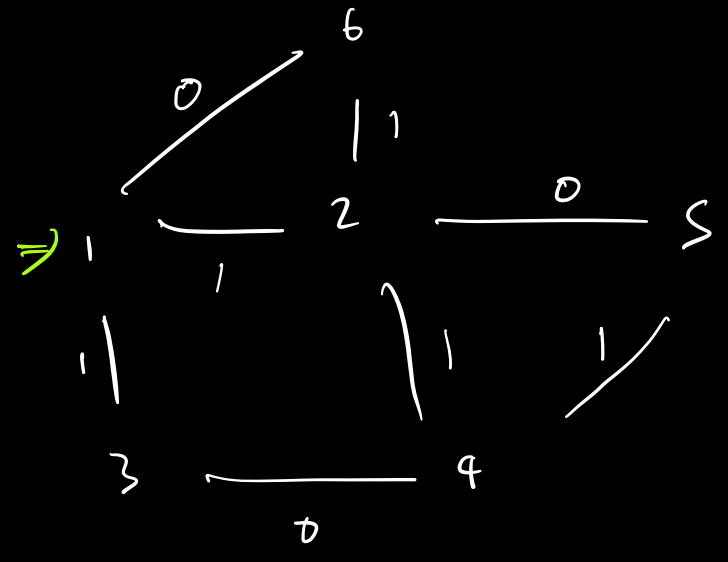
+ve weights $\Rightarrow$ Dijkstra (Single source to all) $O(V \log E)$

-ve weights $\Rightarrow$ Bellman Floyds $O(V^2)$

Floyd Warshall (all vertices to all vertices)

TC: $O(V^3)$

-ve cycle ⇒ Shortest Path does not exist



x 6 x 5 3 4

0, x

0 weight ⇒ Start of the queue

x weight ⇒ End of the queue