adv data structs

Segment Tree + BIT (Fenwick Tree)

Pre-requisite → Recursion

Basic Constructs

structs / class

What we will cover → Segment tree, lazy prop, BIT, 2D segment tree, merge sort tree, problem solving

Motivation Problem :→ Given a list of integers. You will also get Q queries. In each query you will get 2 no. L and R denoting indices of array. For each query find the sum of all the number whose index lie in the range [L, R]

$$\begin{array}{cccccccc} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ \end{array}$$

[2, 3, ~~5~~ 10, 2, 6, 4, 5, 2]

arr[x] = y

$n \le 10^6$

$Q \le 10^6$

[2, 4] → 11

[3, 6] → 17

2 − 10

Brute force → foreach query linearly process the indexes from L to R.

worst case
L = 0 , R = n-1

$O(q \times n)$ → TLE

Efficient Sol^n

$f(x) \rightarrow$ returns the sum of elements in the range $[0, x]$.

prefix sum func^n

precompute

$sum(l, r) = f(r) - f(l-1)$

$= f(r) - f(l) + arr[l]$

$O(1)$

Say, we have one more type of query in the same problem. In this query we will get two values X and Y. We need to update the X^{th} index of array with a value Y.

if we just update the index with a new value, the previously computed prefix sum is invalid.

So we need to update the prefix sum array in the update query only.

$$q_1 + q_2 n \approx O(qn)$$

(Range Query Problems)

Segment Trees

→ It is a hybrid data structure.

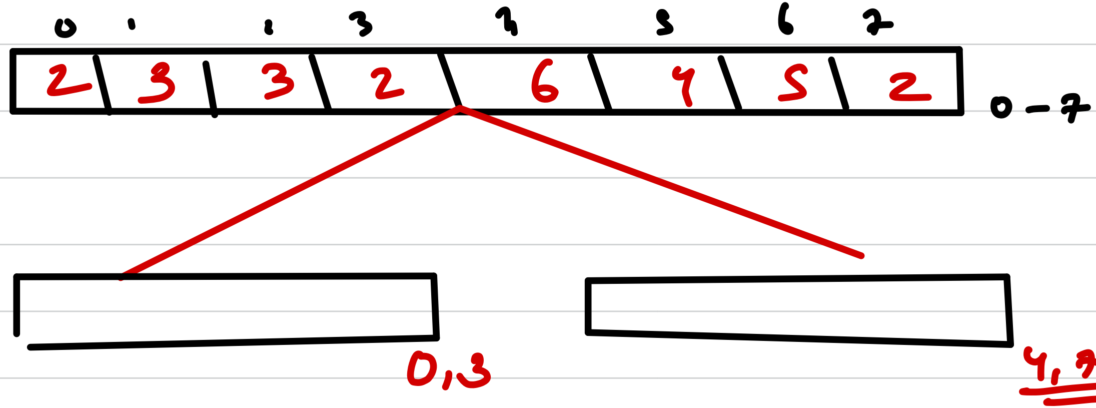
1. It is represented as a tree
2. But stored in an array.

→ It will help us to update the complexity of our prev algo to $O(\log n)$

The Seg tree divides your problem into multiple segments & prune out those which are not useful for you.

* It is a Binary tree.

2, 3, 3, 2, 6, 4, 5, 2



Binary tree

$$h \approx O(\log n)$$

2, 3, 3, 2, ~~6~~, 4, 5, 2
10

4 - 10

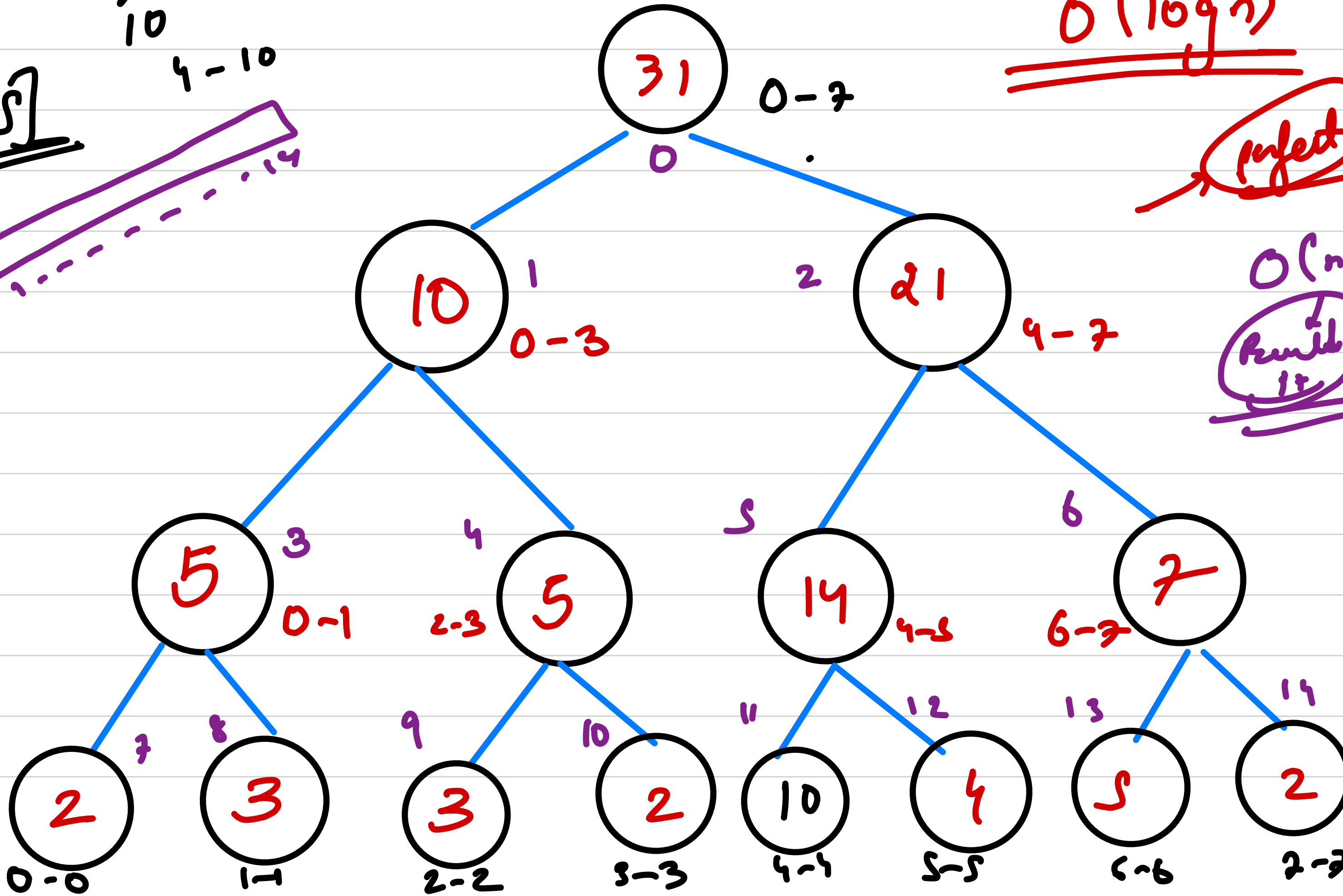
[2 5]

0 (log n)

perfect

0 (n)

Build it



Let N_L and N_R be the range denoted by nodes

let L & R be the query

Base case

if ($R < N_L$ or $L > N_R$)
complete outside //return 0;

if ($L \leq N_L$ and $R \geq N_R$)
complete inside.

{ Partial one $\longrightarrow$ left and Right

height of segment tree

→ what kind of Binary tree is segment tree.

Balanced tree → full Binary tree

$P_i = x$

$L \Rightarrow 2x+1$

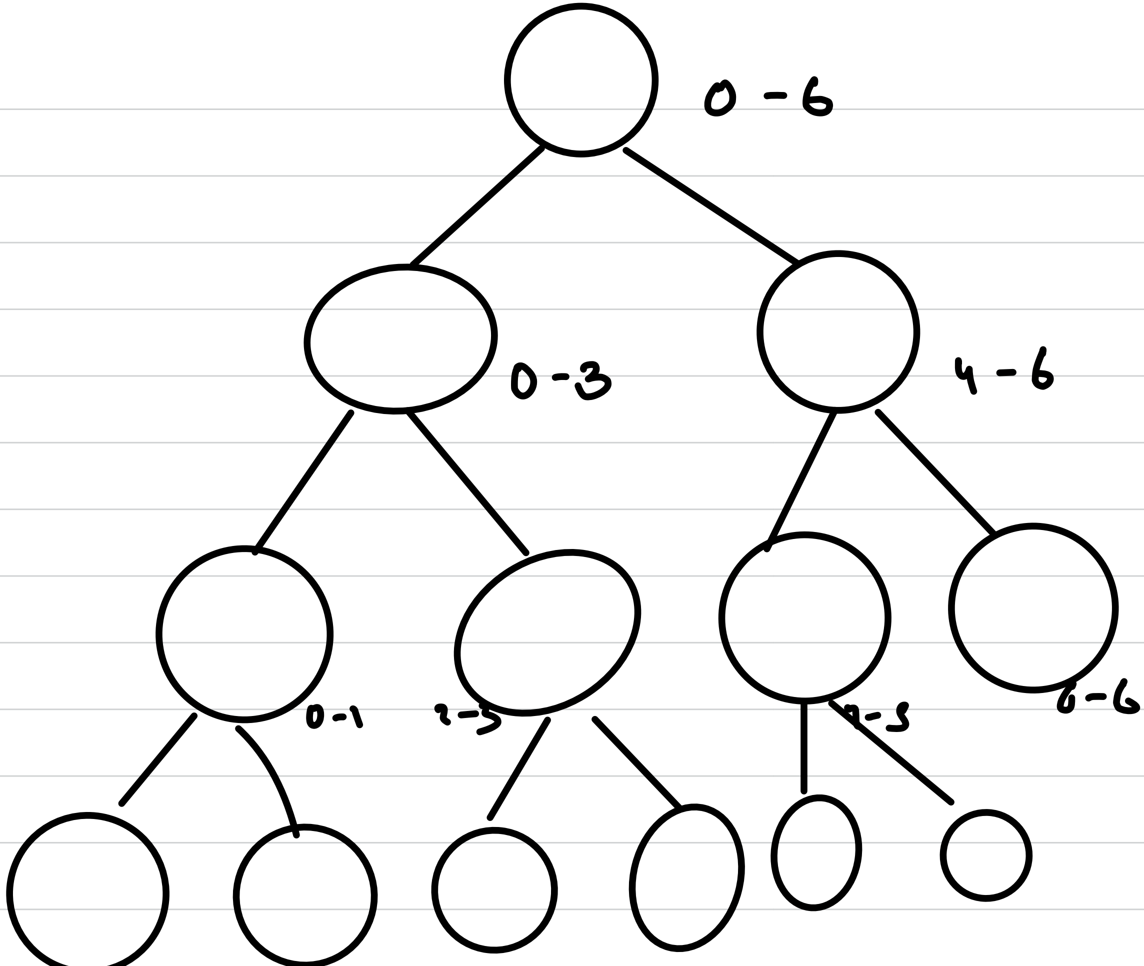
$R \Rightarrow 2x+2$

} if we store root node at index 0.

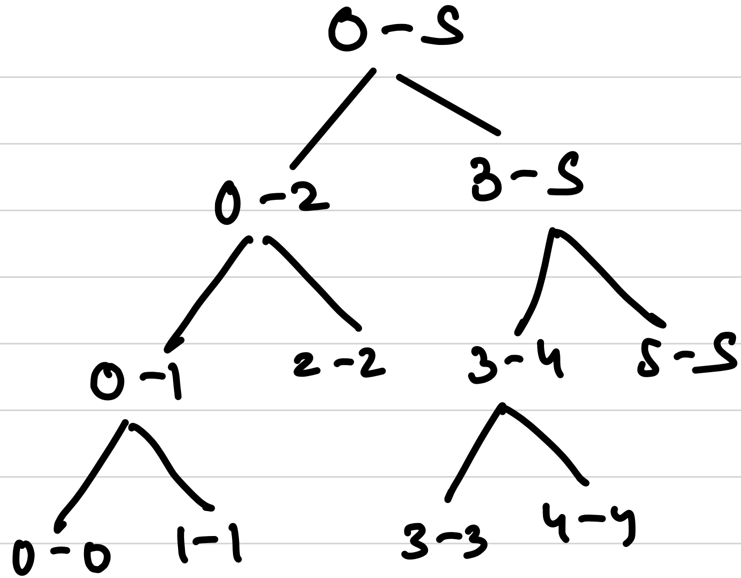
$L = 2x$

$R = 2x+1$

} if we start root node from index 1


0 - 6
0 - 3
4 - 6
0-1
2-3
6-6

$\lceil \log n \rceil$



not a complete binary tree

But a **full BT**

$\rightarrow$ $\underbrace{2^0 + 2^1 + 2^2 \ldots\ldots\ldots 2^{\log n}}_{gp}$ = Total nodes

$\boxed{n \rightarrow \text{length of array}}$

$$O\left(\frac{1 \times (2^{\log n + 1} - 1)}{2 - 1}\right)$$

$$O\left(\boxed{2 \times 2^{\log n} - 1} \right) \rightarrow \boxed{2n - 1} \leftarrow \approx O(n)$$

$S(n)$ ← tightest upper bound for $S(n)$

size of segment tree

height of full tree

$$S(n) \leq 2^{\lceil \log_2 n \rceil + 1} - 1$$

$$S(n) \le 2^{\lceil \log n \rceil + 1} - 1$$

$$\boxed{4n}$$

$$S(n) < 2 \times 2^{\lceil \log n \rceil}$$

$$S(n) < 4 \times 2^{\lceil \log n \rceil - 1}$$

$$x^{\log_x n}$$

$$n$$

$$S(n) \le 4 \times 2^{\lfloor \log n \rfloor}$$

$$\boxed{S(n) \le 4n}$$

$$\boxed{c \times g(n)}$$

tight upper bound

Bob and array queries → initially array is of all zeros

1 → x → $2a[x]+1$ — increment no. of 1 by 1

2 → x → $\left\lfloor \frac{a[x]}{2} \right\rfloor$ — decrement no. of 1 by 1

3 → L R → ~~# of 1's~~

$\frac{0}{2} \rightarrow 0$

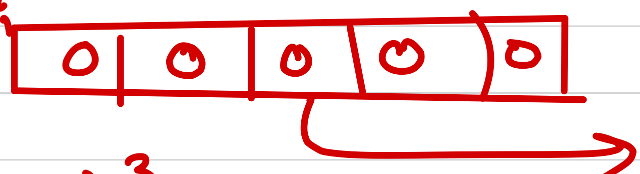
$2 \times 0 + 1 \rightarrow$

$0 \rightarrow 1 \rightarrow 3 \rightarrow 7$

15

$0 \rightarrow 1 \rightarrow 3 \rightarrow 7 \rightarrow 15$

0   01   11   111   1111



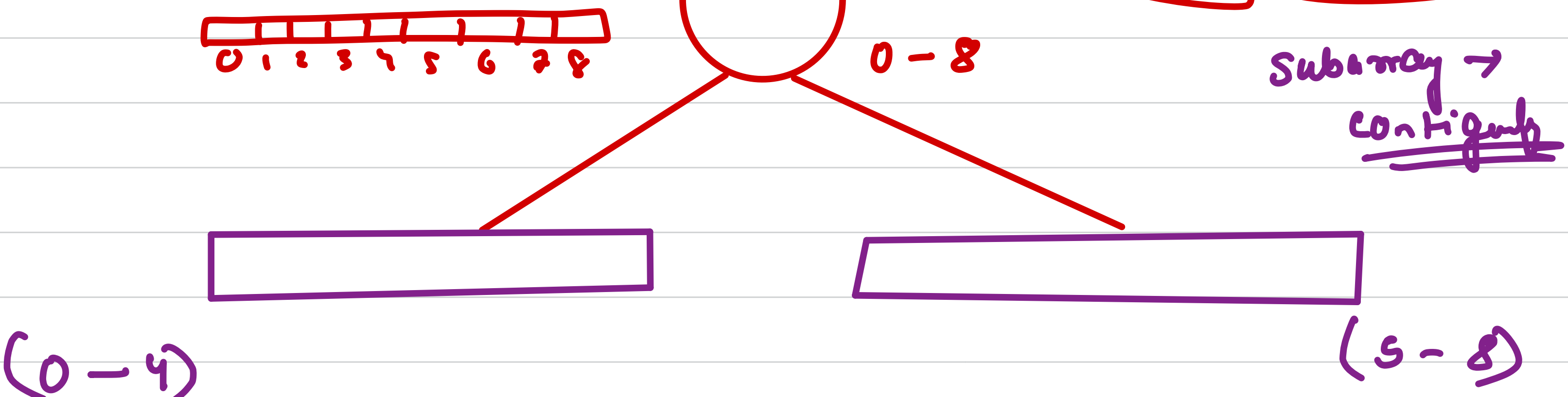
$\left\lfloor \frac{15}{2} \right\rfloor \rightarrow \left\lfloor \frac{7}{2} \right\rfloor \rightarrow 3$

1111 → 111   11

what data we should store on a node.

and how to compute this data from children.

Q ⇒ GSS1 – Spoj



→ {
MaxSum
leftSum + best right prefix Sum
rightSum + best left suffix Sum
left best suffix + right best prefix

Q. 2v3 - hackerearth

$$(101110)_2 = 1\times 2^5 + 0\times 2^4 + 1\times 2^3 + 1\times 2^2 + 1\times 2^1 + 0\times 2^0$$

$$(101)(110) \qquad = 32 + 8 + 4 + 2$$

$$= \underline{\underline{46}}$$

$$(101)_2 \rightarrow 5 \rightarrow (1\times 2^2 + 0\times 2^1 + 1\times 2^0)$$

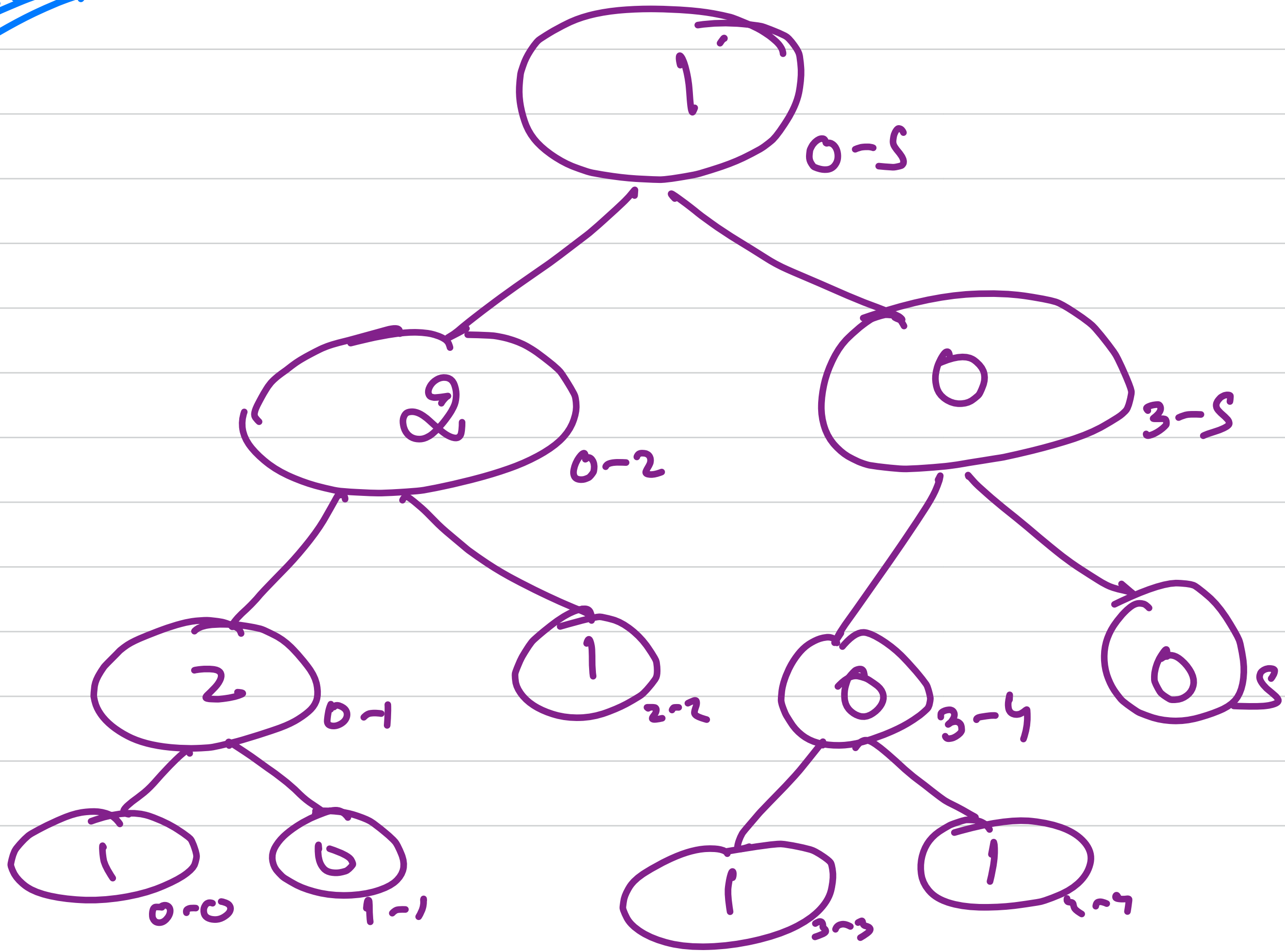
$$(110)_2 \rightarrow 6 \rightarrow (1\times 2^2 + 1\times 2^1 + 0\times 2^0)$$

$$2^3 \times (1\times 2^2 + 0\times 2^1 + 1\times 2^0) + 1\times 2^2 + 1\times 2^1 + 0\times 2^0$$

$$2^3 \times 5 + 6 \Rightarrow \underline{46}$$

10^5

p.3



$$(a+b) \% c = (a \% c + b \% c) \% c$$

$$(a * b) \% c = \underline{\underline{(a \% c * b \% c) \% c}}$$

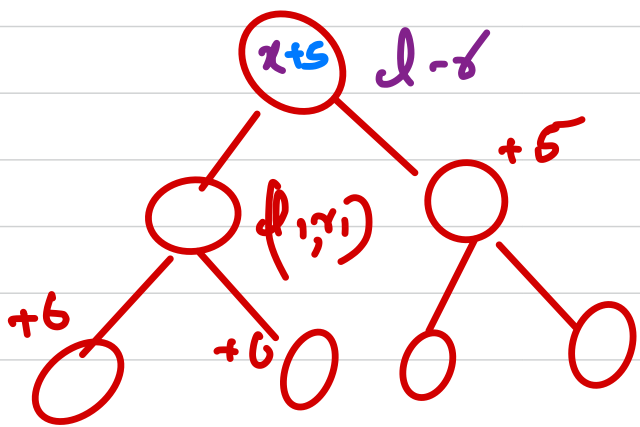
→ We have till now discussed only point update

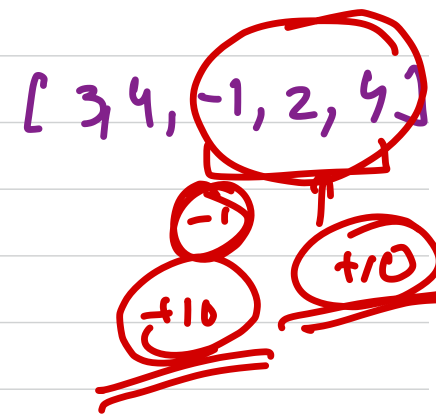
How about update on a range ??

$\underline{\underline{N}}$

Lazy Prop

→ Delay the updates to the descendants of a parent until the descendants need it themselves





RMD

Update → 0-2 +10

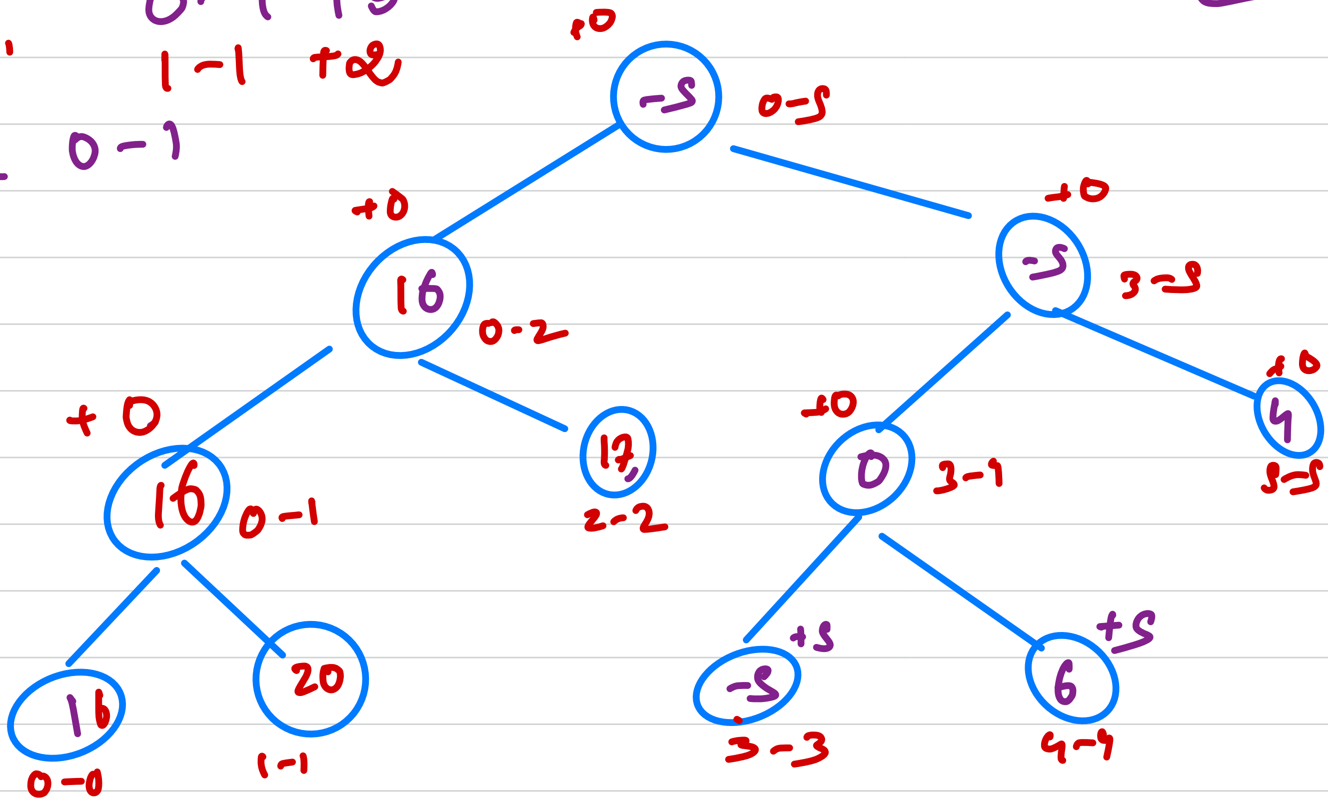
" 0-4 +5

" 1-1 +2

Query 0-1

logn

$\boxed{\begin{matrix}\text{RU}\\ \text{RQ}\end{matrix}}$



Q. Given an array of length n, $(n \le 10^6)$.

find the length of **LIS**. (Longest Inc Subse)

$[2, 4, 1, 6]$ 3

Ex 7, 3, 5, 3, 6, 2, 9, 8

lis(i) → 1 1 2 1

$$f(i) = \begin{cases} 1 + \max(f(j)) & \forall\ j < i \ \&\& \ a_j < a_i \\ \end{cases}$$

lis ending at i

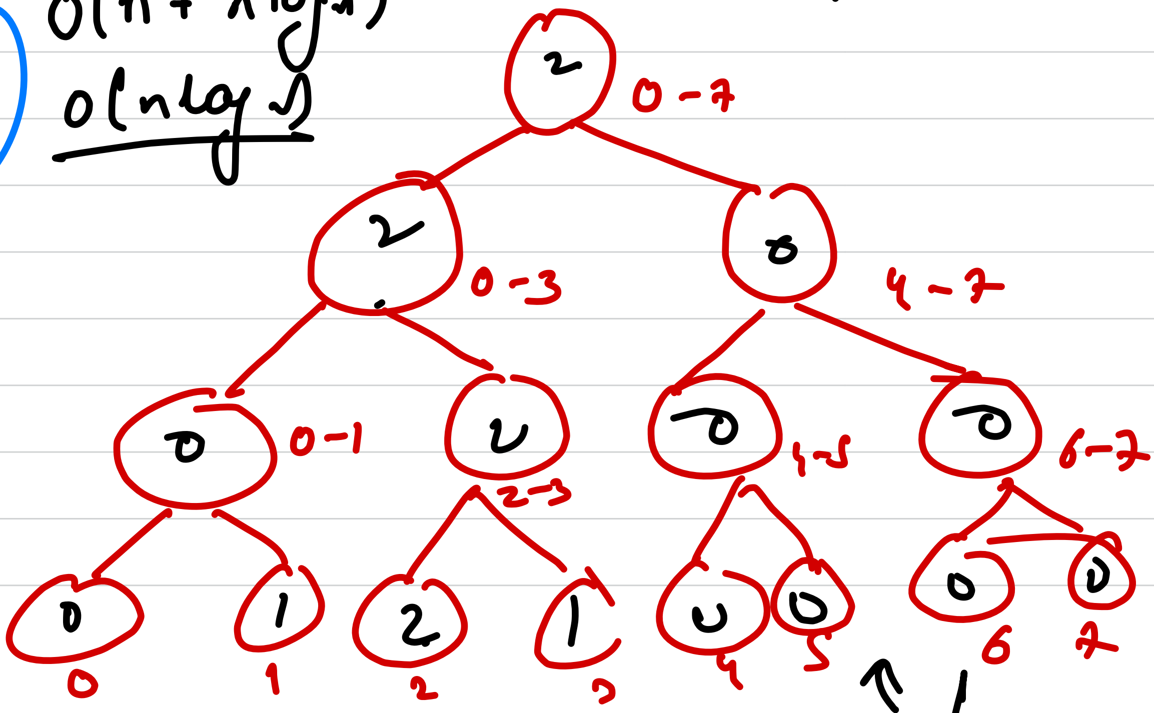
for any element a_i, all the elements $\geq a_i$ are irrelevant to compute CIS. So they can be resolved later.

	7,	3',	5,	3'',	6,	2,	9,	8
l.s	0	1	2	1	3	1	4	4
idx	0	1	2	3	4	5	6	7

$O(n + n\log n)$

$\underline{O(n\log n)}$

assume seg tree

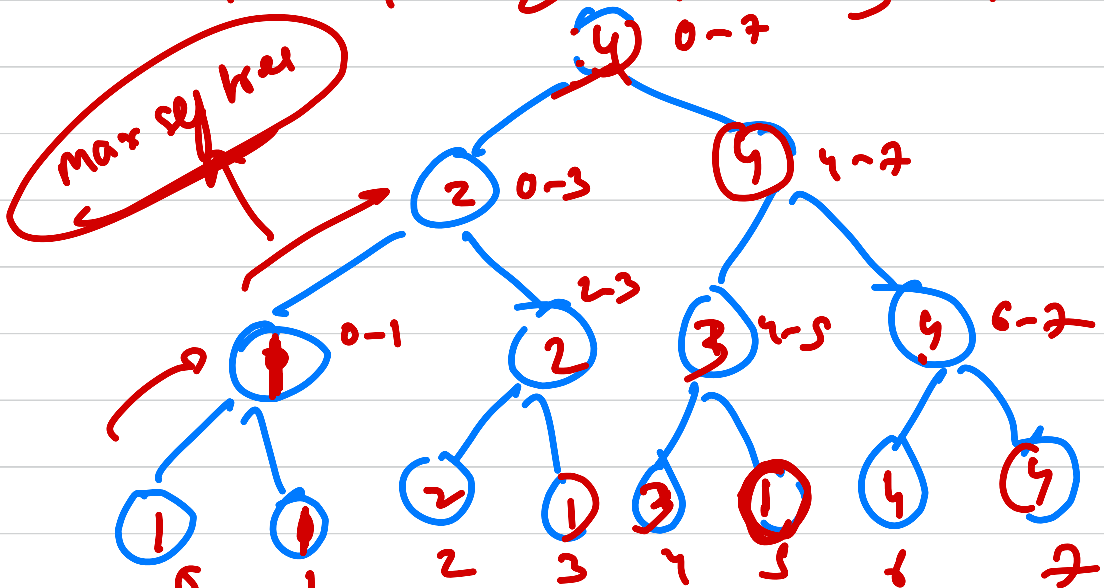


Sort

2
3''
3'
5
6
8
9

7	3'	5	3''	6	2	9	8
0	1	2	3	4	5	6	7
1	1	2	1	3	1	4	4

→ max



ele	idx
2	5
3''	3
3'	1
5	2
6	4
7	0
8	7
9	6

sorted

any node denotes lis

$a_1 \quad \underbrace{a_2 \quad a_3 \; \cdots}\; \cdots \quad a_n$

(L, R) $\quad (a_m, a_n)$

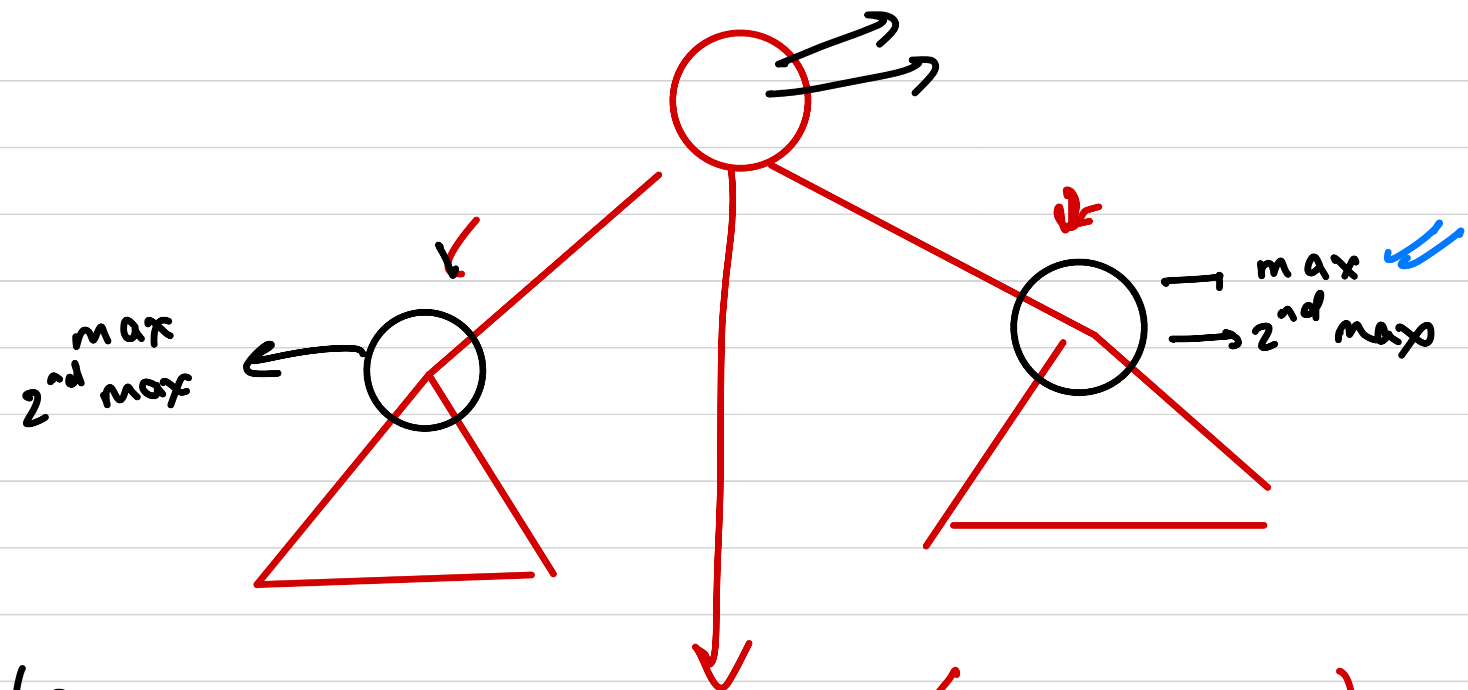
max sum pair in range

$a_m + a_n \to \max$

max seg tree

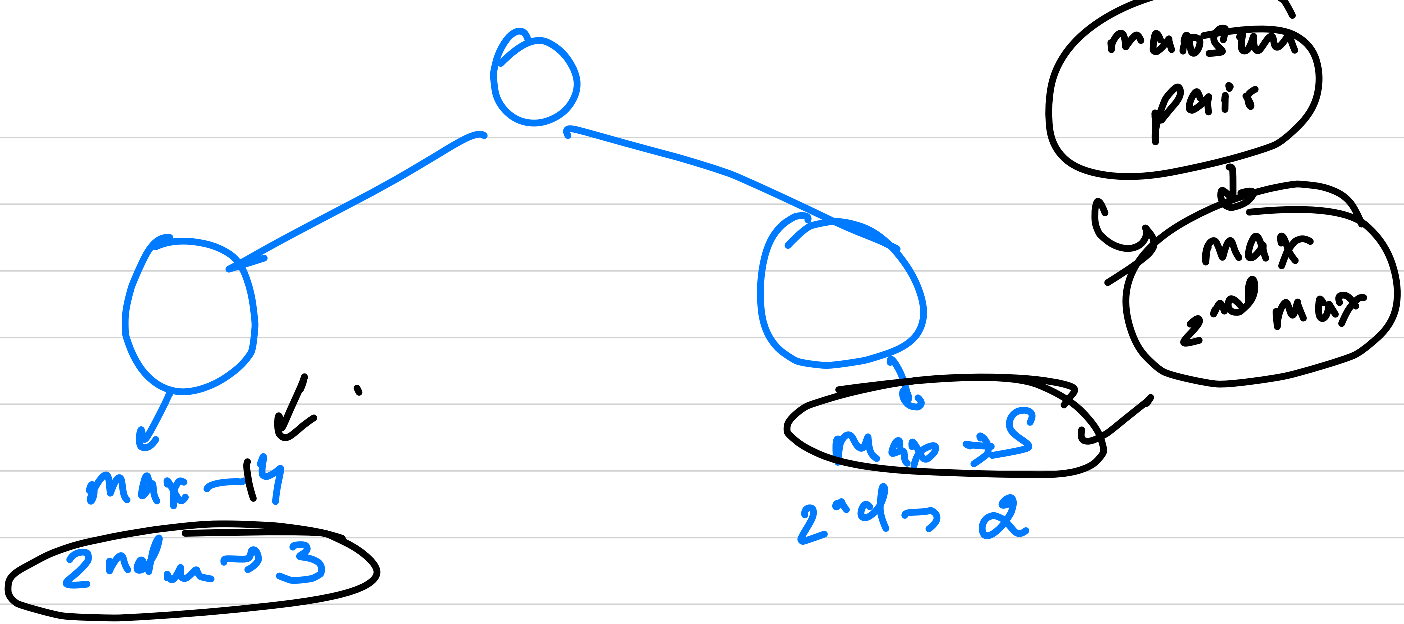
2, 1, 6, 4, 3

max, second max



$$\max = \max(L_{max}, R_{max})$$

$$2^{nd}\max = \min\begin{pmatrix} \max(L_{2max}, R_{max}), \\ \max(R_{2max}, L_{max}) \end{pmatrix}$$



$$\max = \max(L_{max}, R_{max}) = 14$$

$$2nd\max = \min\begin{pmatrix} \max(L_{2nd\max}, R_{max}) \\ \max(R_{2nd\max}, L_{max}) \end{pmatrix}$$

Lazy prop

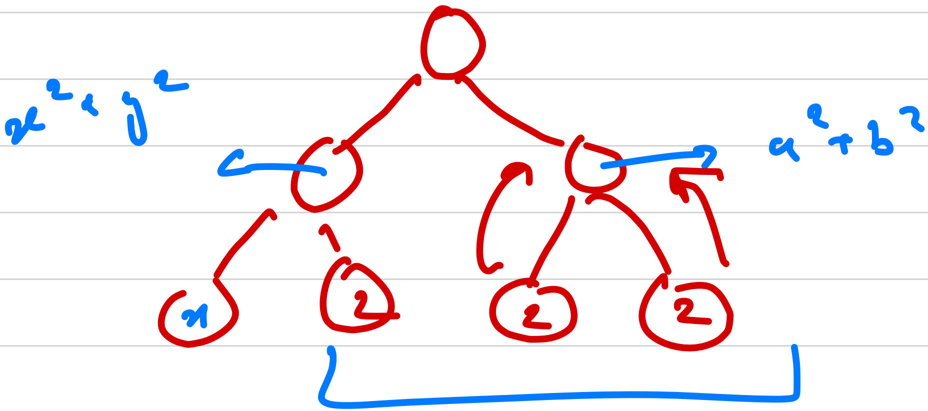
Spoj → SEGSQRSS

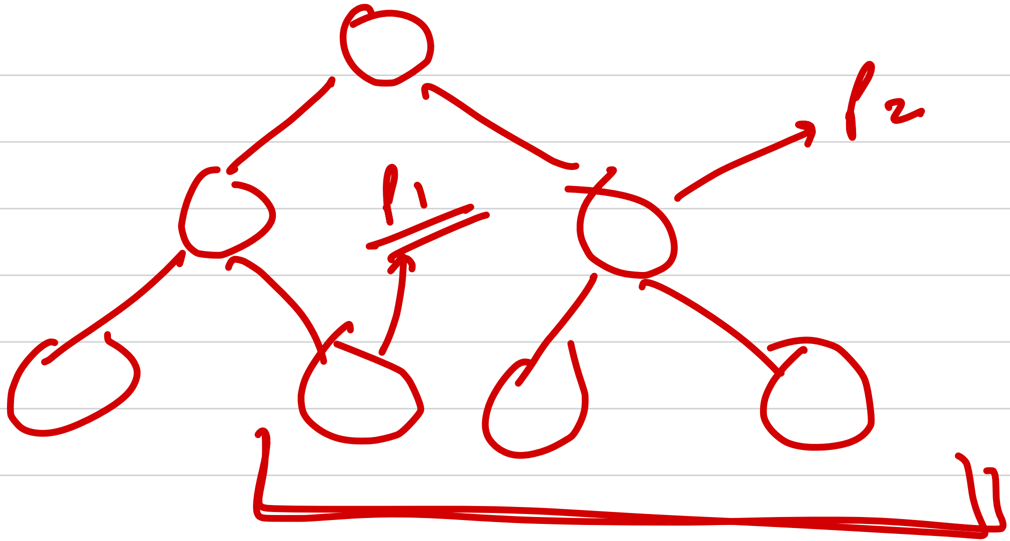
query → easy

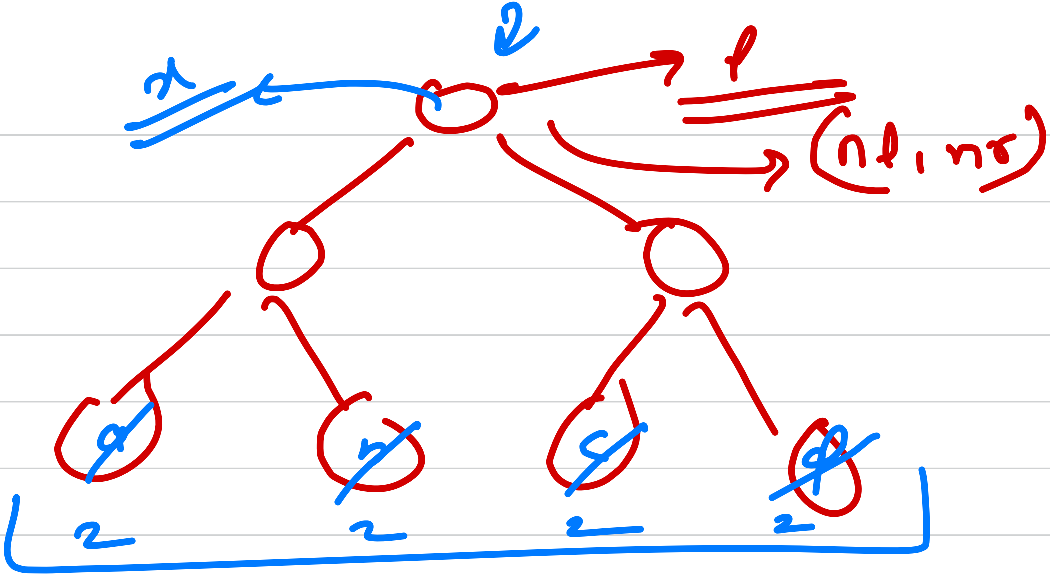
at each node store the sum of square of the range

update → increment
update → set

l - r set




p1
p2



$$\boxed{(nr - nl + 1) * 2 * 2}$$

l

current

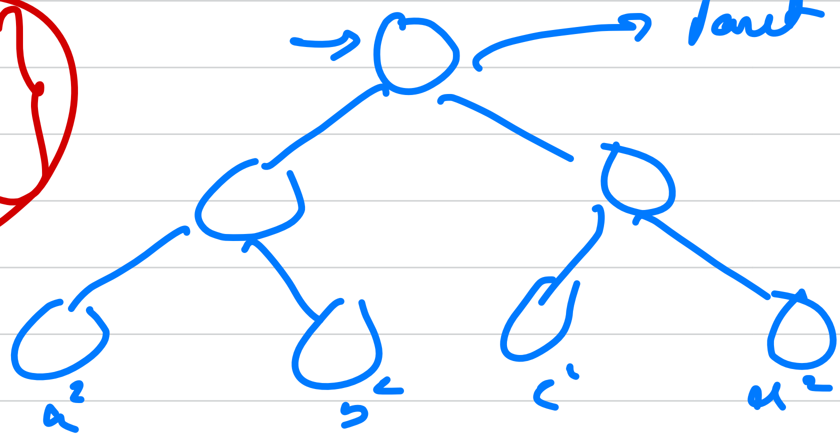
$a^2 \quad b^2 \quad c^2 \quad d^2 \rightarrow$ stored

$(a+x)^2 \quad (b+x)^2 \quad (c+x)^2 \; - - - -$

$\boxed{a^2 + x^2 + 2ax}$ ← point

sum sq

sum



lazy → $+= 2ax + 2bx + 2cx + 2dx + \underbrace{x^2 + x^2 + x^2 + x^2}$

$(r - l + 1)\,x^2 + 2 \times x \times (\text{sum})$

{sum of sq} {sum of sq ↓ sum} → set → x, inc → x

Range update ↓ lazy prop

initially

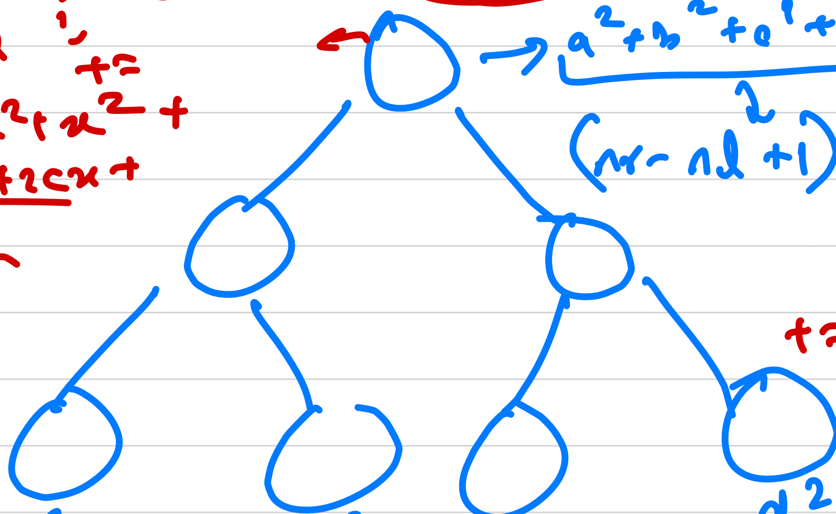
$a^2+b^2+c^2+d^2$

nd, nr

$a^2+b^2+c^2+d^2$ → $(nr-nd+1)x^2$

prev val +=

$x^2+x^2+x^2+x^2+$
$2ax+2bx+2cx+$
$2dx$



$+= (nr-nd+1)x^2 + 2x\underbrace{(a+b+c+d)}_{\text{sum on range}}$

$(a+x)^2$

a^2, x^2 → a^2+x^2+2ax

b^2, x^2 → b^2+x^2+2bx

c^2, x^2 → c^2+x^2+2cx

d^2, x^2 → d^2+x^2+2dx