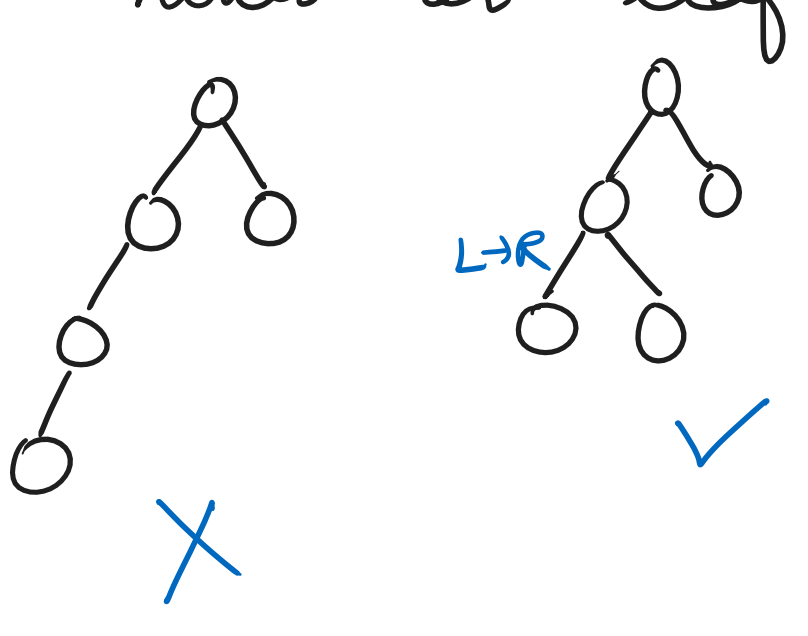
Introduction to Heaps :→ → Array ✓ Tree

An heap is a Complete Binary Tree where insertions always happen from left to right at each level & the lowest level can have incomplete nodes or leaf nodes.



If we have a requirement where we need to extract the min / max value in a collection in constant time O(1) we use heaps.

Height of a Complete Binary Tree / Balanced Binary Tree (AVL / Red Black)

Let's build the tree level by level :→

Level	Nodes at this level	Total nodes so far:
0	1	1
1	2	3
2	4	7
3	8	15
4	16	31 = 2^5 - 1
h	2^h	2^(h+1) - 1

→ $n = 2^{(h+1)} - 1$

$\log_2 n = \log h+1$

$h = \log(n) + 1$

$= \log n$

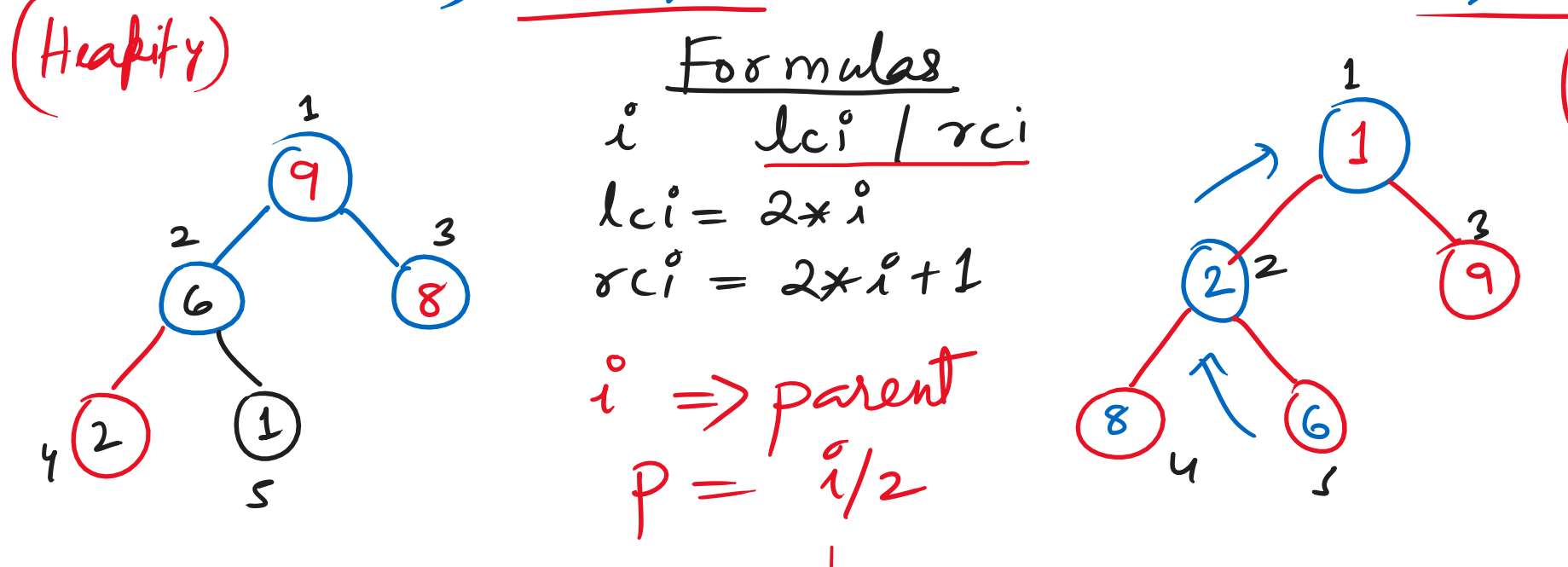
Optimal height is $h = O(\log(n))$

Heaps → (Max Heap) [8, 2, 9, 6, 1] (Min Heap)
0 1 2 3 4

→ L<N>R L>N<R

(Heapify) (log n)

Formulas
i lci / rci
lci = 2*i
rci = 2*i+1
i ⇒ parent
P = i/2



-1	9	6	8	2	1
0	1	2	3	4	5

Mx H

Cardinal / Sentinel value

Mn H

-1	1	2	9	8	6
0	1	2	3	4	5

Heaps Interview Questions :→ (k largest elements) (kth smallest element)

① Find the k largest elements in an array :→

arr = [19, 21, 4, 9, 3, 7] k = 3 (n)

Sort (n log n) ✓ optimised

n log(n-k) ✓ 1000

MinHeap [~~3~~, ~~4~~, ~~7~~, 9, 19, 21] ans

if size > k remove element

n (k) log k (n-k)

Find the kth (Smallest) Element in an array :→

arr = {7, 10, 4, 3, 20, 15} k = 3

n-elements

log(n-k) insertion or deletion

Max Heap: [20 ×, 15 ×, 10 ×, 7, 4, 3]

mh.peek()

Size
1
2
3
4 → 3
5 → 3
6 → 3

Dijkstra's
Prim's (MST)

* Minimum Cost of Connecting Ropes :→ L>N<R

[4, 3, 2, 6] ⇒ (min Heap) 29

→ Sort (n logn) [2, 3, 4, 6]
[(5), 4, 6]

nlogn Sort [4, 5, 6]
[(9), 6]

nlogn sort [6, 9]
(very large)! (15)

{2, 3, 4, 6} O(1) O(1)
2+3 (5) {4, 5, 6} O(1) O(1)
4+5 (9) {6, 9}

5 + 9 + 15 = (29) Cost

Trie Data Structure :→

Dictionary <Abhijeeth, true>
<Abhishek, false>
<Abhimanyu, true>
String, Boolean

} Too much space Memory is less efficient

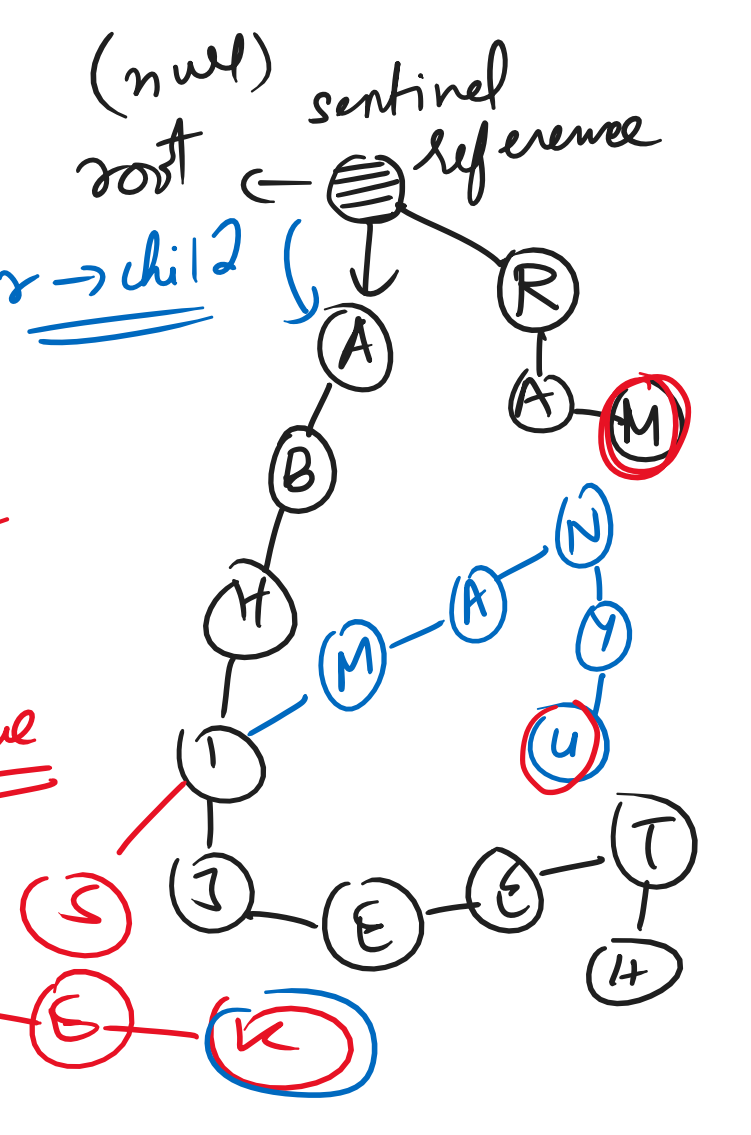
LCP

C++
map → unordered Hashmap (RAM)

bool isEndOfWord = true

Bhijeet

26 [ ]



code
coding
coder
codingninja
coders arcade
codeflow

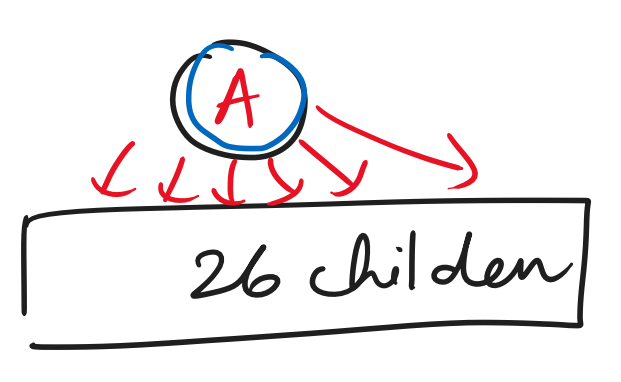
Longest Common Prefix

LCP = cod ⇒ TRIE

flow
fly
flower
flight
flown

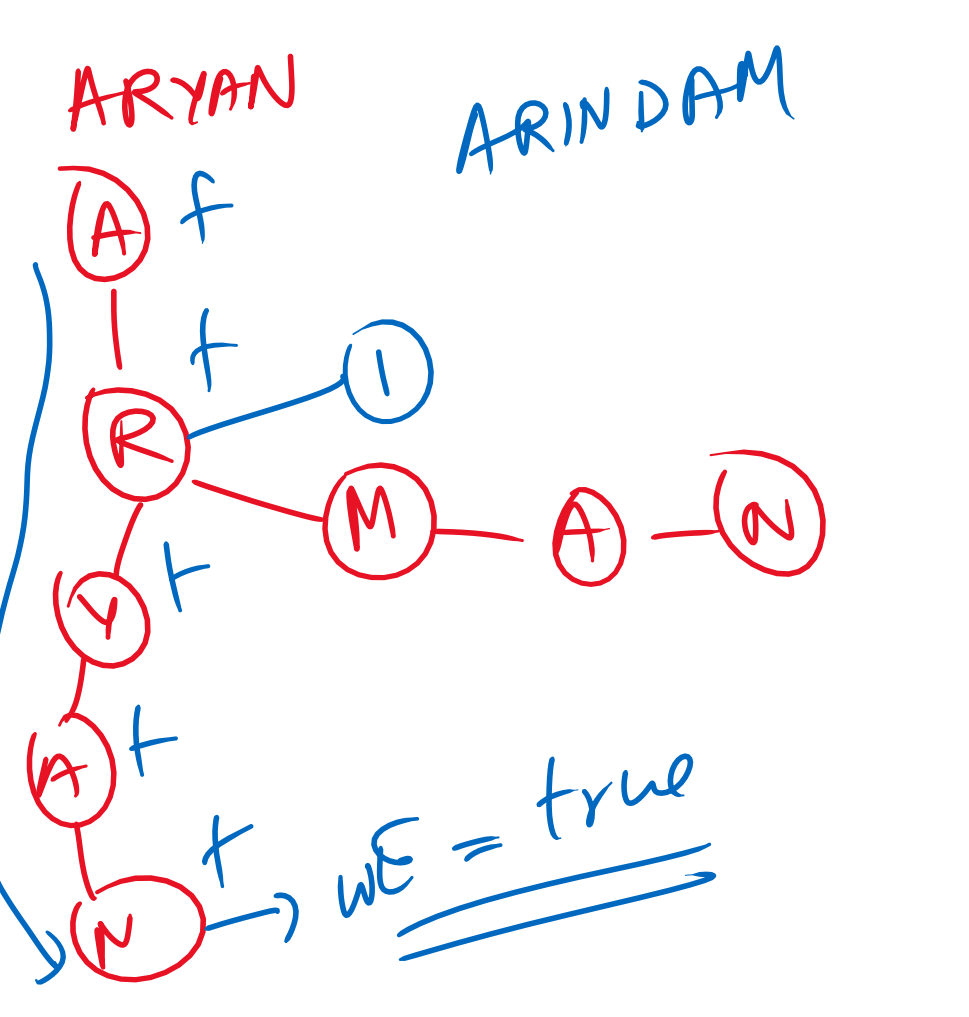
fl

{ insert
search
delete() }



AB
AC
Ad

Note :→
① If the node with the character exists we don't create the node again.
[0, 1, 2, 3, 4 ....., 25]
② If it doesn't exist we create a new node of that character.



Longest Common Prefix: (A)

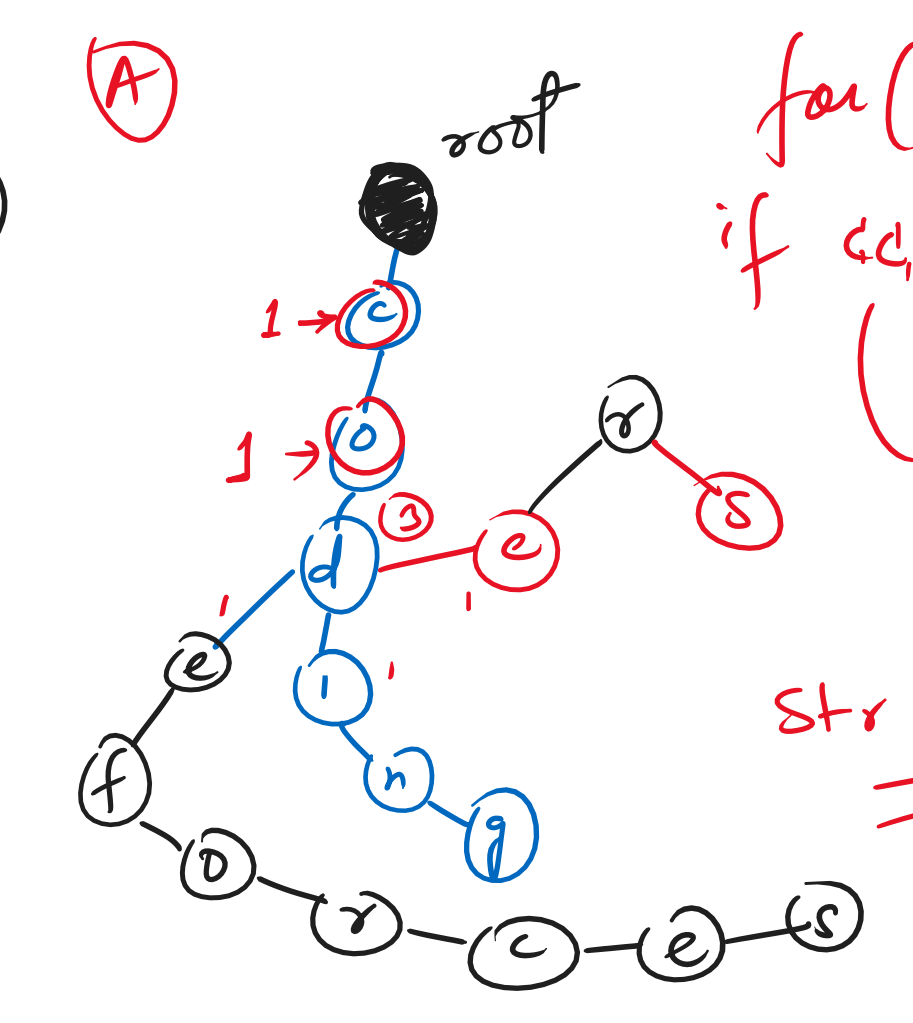
(pseudo-code)

s1 → coder
s2 → code
s3 → coding
s4 → coders
s5 → codeforces

null

len 0 word

children[26];
wordEnd = false;
childrenCount = 0;
default;



for (i=0, ...)
if cc > 1
(stop)
break!

str = return
(null)
(0)

Introduction to Greedy Algorithms :→ (min/max)

* Why greedy → Because we intentionally take the smallest or the largest route / path or value.

Important Problems for Placements :→

1. Minimum no. of coins.
2. Minimum no. of platforms.
3. Minimum arrows to burst all balloons.
4. Gas Station Problem.
5. Activity Selection Problem
6. Job Scheduling Problem
7. Fractional Knapsack
8. 0/1 knapsack
9. Huffman Encoding
10. Chocolate Distribution Problem
11. Policemen Catch Thieves
12. Lemonade Stand
13. Minimum cost of ropes
14. Nikunj & Donuts
15. Minimum no of steps
16. Assign Cookies