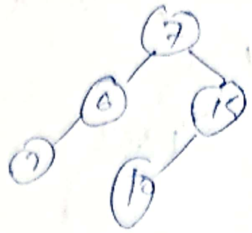
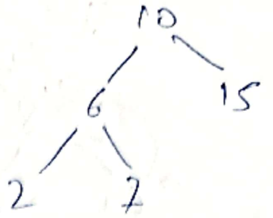
In order = 2 6 7 8 10 15
↑ predec.. ↑ successor.





min don't have predecessor
max " " successor.

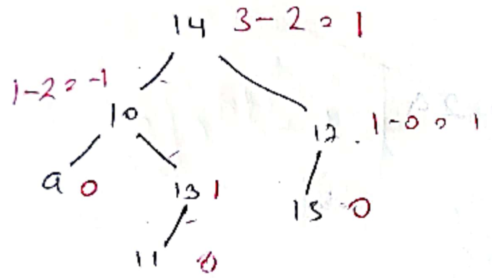
AVL tree

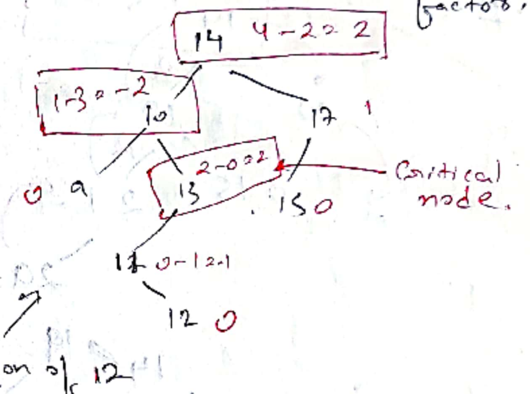
BST → Insert Delete min max search, — Hight dependent worst case O(n) Linear.

AVL : Height Balanced Tree

A BST is an AVL tree if difference of height of left subtree and right subtree is 0, 1 or -1 (Balancing factor).

AVL.





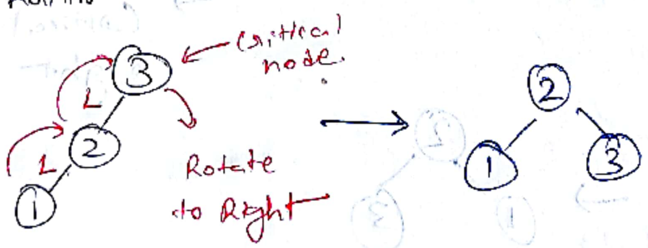
After insertion the first node Balancing factor violated, is Critical node → in leaf to root path.

∴ Rotation needed.

- Single rotation: LL rotation, RR rotation
- Double rotation: LR, RL

Single LL Rotation.

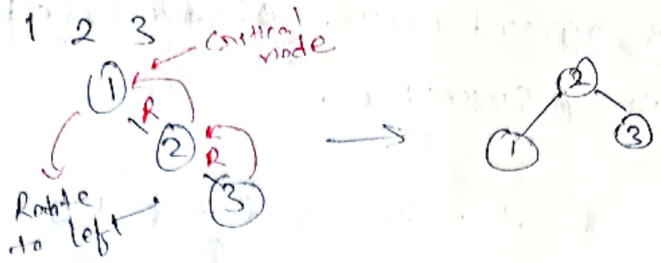
3, 2, 1



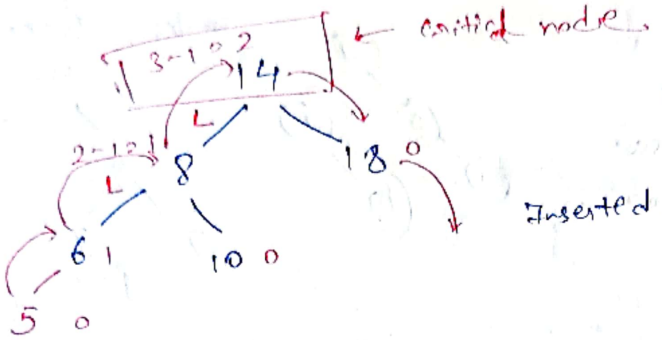
LL
↳ Left subtree of Critical node
↳ Left side of child node of Critical node.

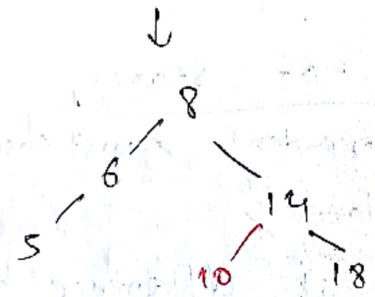
RR Rotation

1 2 3



#

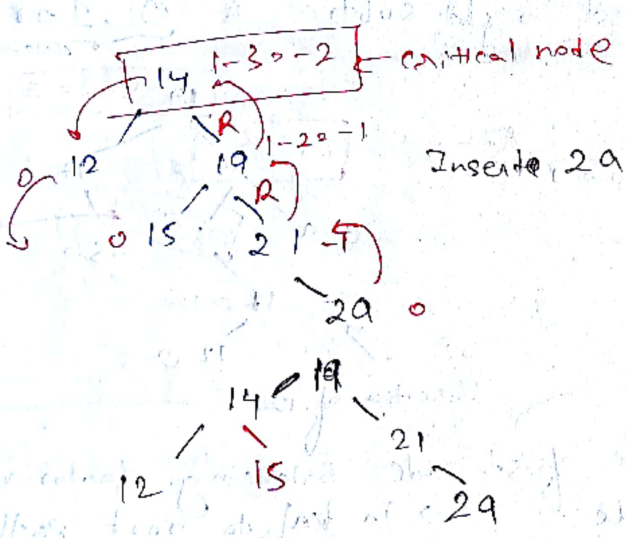




10 ?

As 10 was at the left subtree of 14, ∴ it will be placed at the left of 14

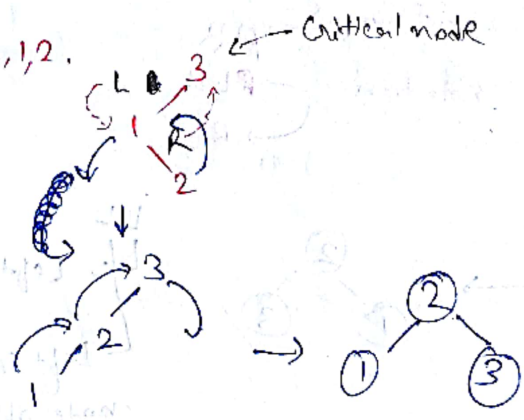
#



15 was at the right of 14. ∴ placed at right child of 14.

Double Rotation
LR rotation.

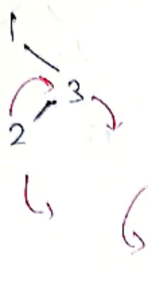
Insert 3, 1, 2.



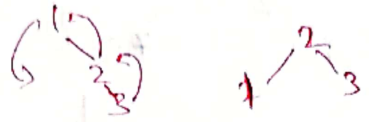
L R
- Left child of critical node is rotated left
- Critical node rotated right

RL rotation:

Insert 1, 3, 2



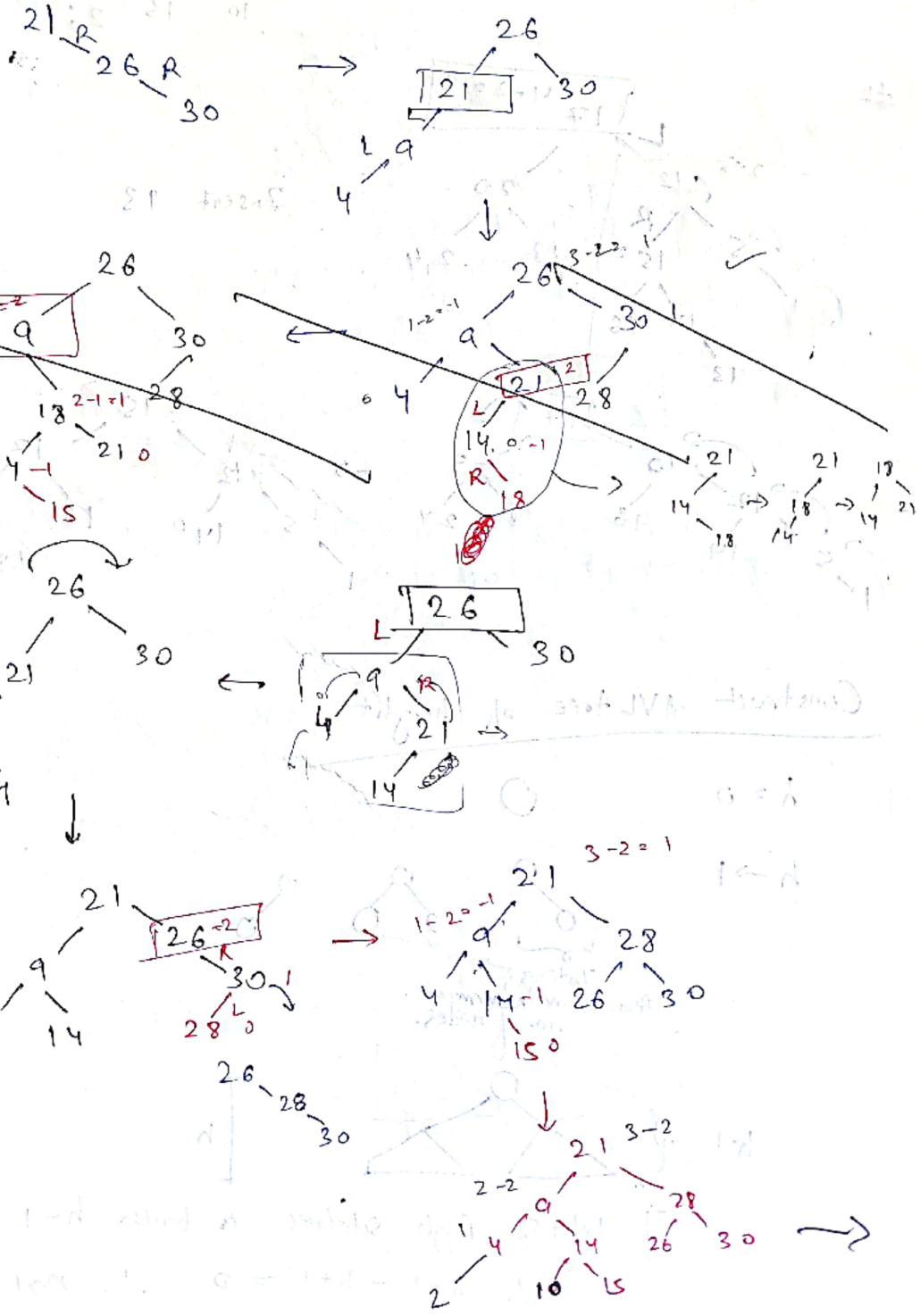
R L
- Right child of C.N. is rotated Right
- Critical node rotated left

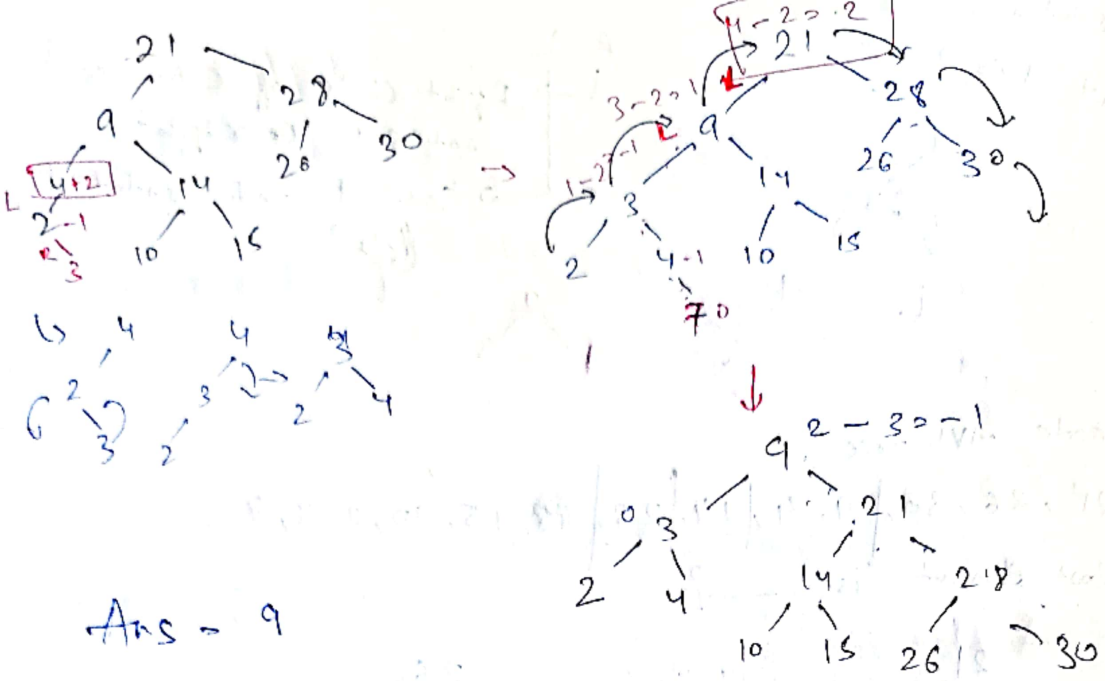


Create AVL tree ?.

21, 26, 30, 9, 4, 14, 28, 18, 15, 10, 2, 3, 7.

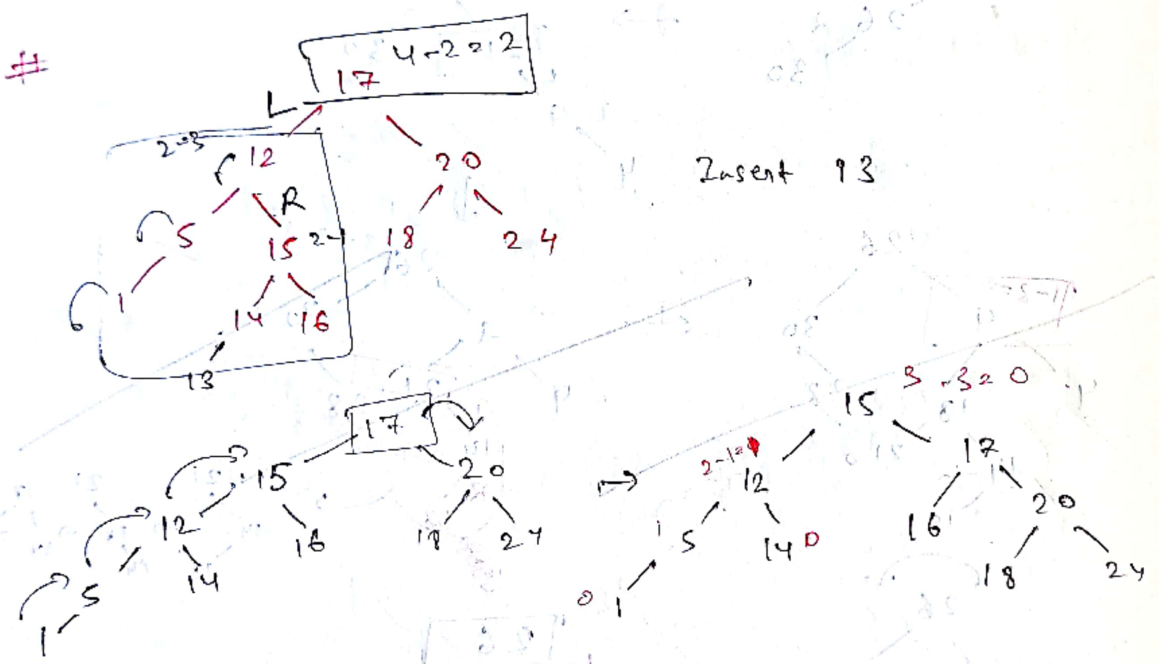
Root element is ____?.





Ans - 9

#

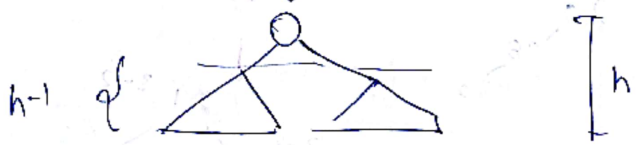


Construct AVL tree of height

$h = 0$

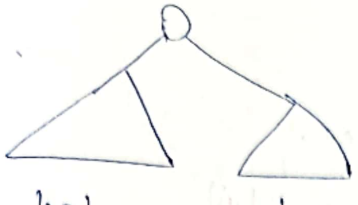
$h \to 1$

Interest
Because minimum no. of nodes.



If Left & Right Subtree is both $h-1$ height

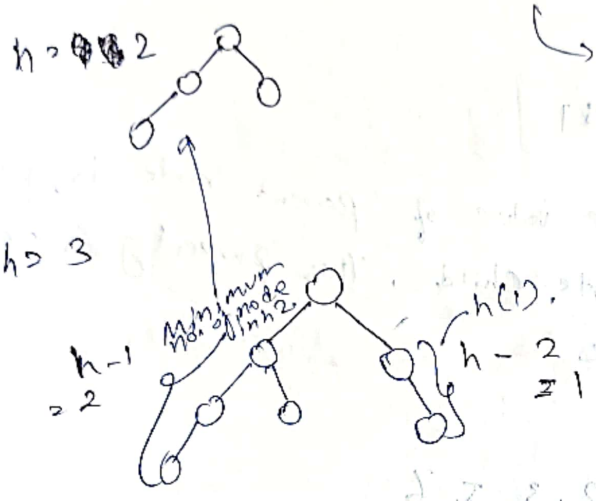
$\therefore h-1-h+1 = 0$ $\therefore$ not minimum node

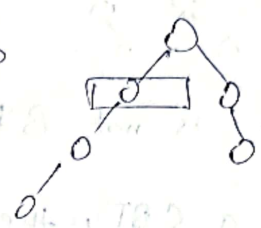


h

= 1
= -1

} Minimum node.





Not AVL Tree.

Minimum no. of node at height (h)

$N(h) = N(h-1) + N(h-2) + 1$

$N(0) = 1 \quad N(1) = 2$

for height 9.

N(0)	N(1)	(N2)	3	4	5	6	7	8	9
1.	2.	4.	7	12	20	33	54	88	143

+1 +1

AVL

Search
Insert
Delete
Min
Max

} Balanced BST $O(\log_2 n)$

Height of AVL tree with MINIMUM no. of Nod.

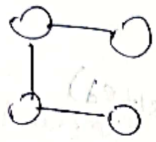
$\lceil \log_2 n \rceil$

$= \frac{5 \times 4 \times 3}{3 \times 2} \times 2 = 10$

$2 \qquad 20$

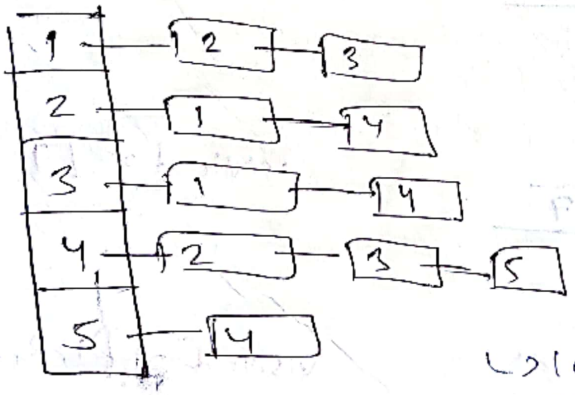
Graph $G(V, E)$

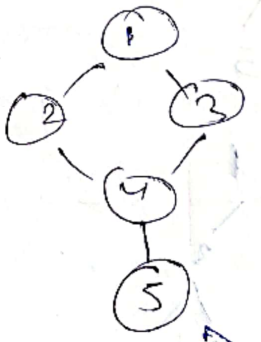
Simple graph! No self loop, no multiple edges between vertices.



1. Adjacency list (Sparse)
2. Adjacency Matrix (Dense)

1. Adjacency List





↳ length = 10

∴ sum of degree = 10

Space → (V + sum of degree)

(V + 2E)

2. Ad. Matrix.

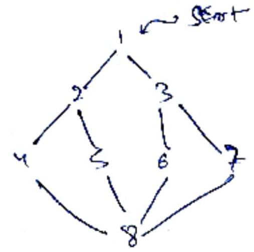
M[i][j] = 1 if $(i,j) \in E$

	1	2	3	4	5
1	0	1	1	0	0
2	1	0	0	1	0
3	1	0	0	1	0
4	0	1	1	0	1
5	0	0	0	1	0

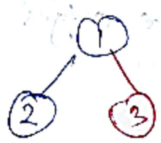
Space = (V^2)

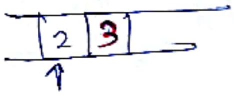
Graph Traversal

DFS Algo — BFS Algo



After visiting a node, adjacent node of vertex is visited. To maintain the order queue data structure is maintained.
when inserted in queue, insert in visited.

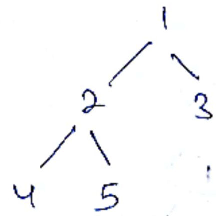


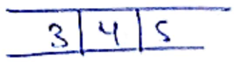


Visited = [1,2,3]

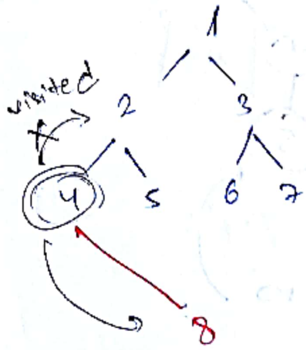
1 visit 2 and insert in queue. (If not visited) ~~visited = [1,2]~~
2 " 3 " " "

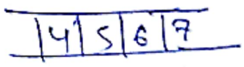
3. Delete element from queue. and visit adjacent node.



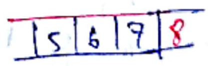


Visited = [1, 2, 3, 4, 5



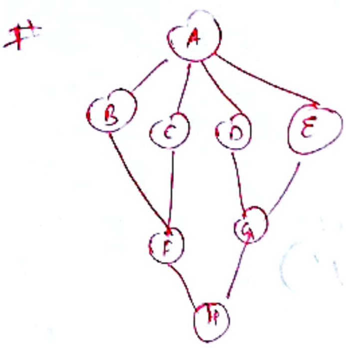


visited = [1, 2, 3, 4, 5, 6, 7

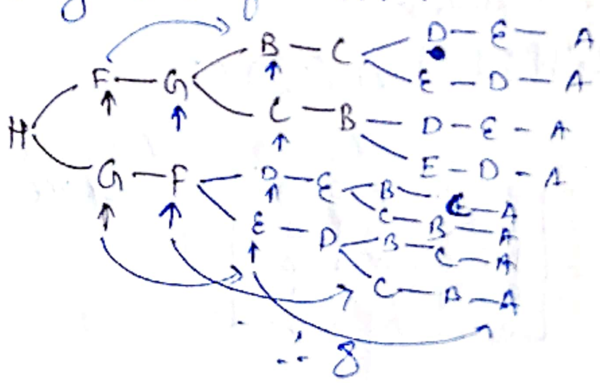


Visited = [1, 2, 3, 4, 5, 6, 7, 8]

#

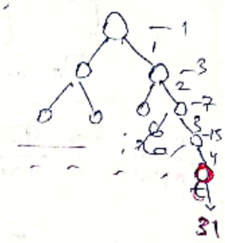


How many BFS from H. ?



BFS is started on a binary tree beginning from root. The vertex 't' at a distance 4 from root. If 't' is the n^{th} vertex in the BFS traversal, then max possible val. of n is.

31 →



Searching Operation

worst case:

Array	LL.	BST	AVL
O(n)	O(n)	O(n)	O(logn)

(can be 8, can't 8)

Direct Addressing: Element is searched by key value.

↳ Random Access

↳ Searching O(1) constant Time

Disadvantage
- Huge size of Array Req.
- Some Index may not present, (wastage of space)

Hashing – Rather than maintaing a big table or array we maintain a small size table called <u>Hash table</u> and to distribute the key a function is called Hash function.

Hash fun should uniformly distribute the key.

Table size (m) if a <u>prime No.</u> then it leads to even distribution.

m = size   n = nodes(elements)

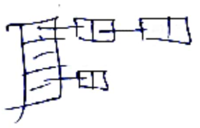
Each location of table is called bucket, slot.

Load factor (α): Ratio of no. of keys to No. of slots. $\alpha = \frac{n}{m}$

Collision: 2 key mapped to same location.

Resolution Techs
- ↳ open hashing
- ↳ closed hashing / open Addressing

Open hashing: In case of collision, collided key store outside of table. 1 tech: Chaining or seperate chaining (LL.



Consider a hash table with 100 slots. Collision are resolved using chaining. Assuming simple uniform hashing. what is the probability that the first 3 slots are unfilled after first 3 insertion.

First insertion	Second	Third
Not inserted in first 3 → 97/100	97/100	97/100

$\rightsquigarrow \frac{97 \times 97 \times 97}{100^3}$

— In chaining slot is not blocked. because we dont store within the table

Closed Hashing

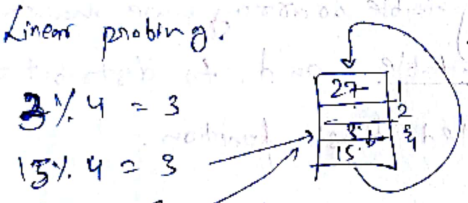
In case of collision we store the key within the table in different position.

(i) Linear probing → Incase of collision occurs the next empty position search in linear fashion in the table.

(ii) Quadratic "

(iii) Double Hashing

(i) Linear probing.



$27 \% 4 = 3$

$15 \% 4 = 3$

$23 \% 4 = 3$

Table starts from 0 index.

Checks that next position is empty or not. If not empty move to next position.

How many diff. insertion seq. of the key value using the same hash function & linear probing will result in the hash table.

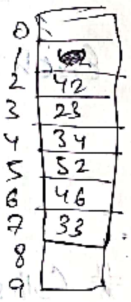
A 10
B 20
C 30
D 40

(42, 23, 34, 46) (52, 33)

At their position — In this order

$\therefore$ 4! comings $\times$ 1

Insertion seq. = 24 Ans (Not in option)



(42, 23, 34, 52) (46, 33)

52 — Fixed (there was a gap)

$3! \times 1 = 6$ Insertion seq.

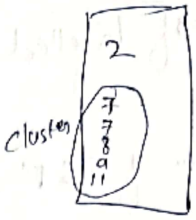
$\therefore$ Ans. is $24 + 6 = 30$

Linear probing suffers from primary clustering.

Once the cluster forms bigger it get faster it grew.

Insert & search will take more time.

performance degrade.



Overcoming

Quadratic prob.

12 % 10 = 2
13 % 10 = 3
22 % 10 = 3
Collision

∴ $(1^2 + 1^2)$ % 10 = 3 Collide

$(2 + 2^2)$ % 10 = 6

2^3 :

2^4 :

Collided no + 1 no

Prob.

Some key value cycle through the list.

Size of table m is prime → max. Location search.

Solution

Double Hashing

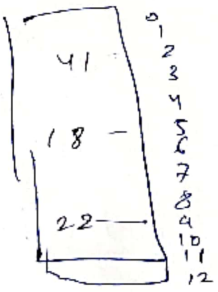
1 primary hash function.

1 secondary " " for getting the offset.

18, 41, 22, 44, 59, 32, 31, 73

$h_1(x) = x \bmod 13$

$h_2(x) = 8 - (x \bmod 8)$ ← plan B



44 % 13 = 5 collide probe

8 − (44 mod 8) = 4

prob 1 = (5+4) % 13 = 9 collid.

pr. 2 = (9+4) % 13 = 0 ✓