Static Data structure Vs Dynamic Data structure

Fixed sized while creating array

non-fixed sized while creating List

Date: 11/01/2022

M T W T F S S

Array uses continous memory blocks physically.

Time-Complexity

Big-O (worst case analysis)

Amotized Analysis (Average case analysis)

13/01/2022

Preliminary Data Types (PDT) {int, float, double}

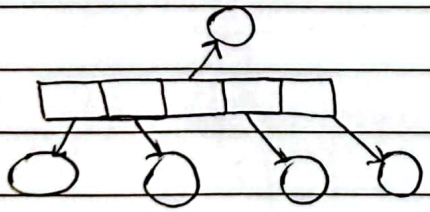
Abstract Data Types (ADT) { Stack, Queue, Point2D}

Abstraction is a method to expose some features and hide some.

arrays are homogenous, because we can acces ~~$~~ any ~~the~~ element by base pointer.

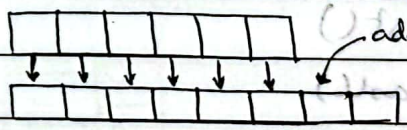
Python lists are hetrogenous.

Python list makes an array of pointers.

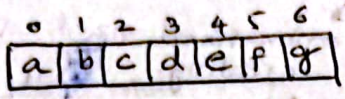


Int Type occupy 2 bytes. If we create an array of size 10. It occupies 20 contigous bytes in the memory.

Python list grows dynamically. when a new thing is appended a new list is formed with a bigger size and copy all elements to it.



stack (LIFO)



search O(n), O(nlogn)

add O(1)

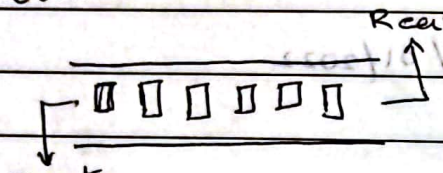
remove O(1)

Date: 18/01/2022

M T W T F S S

circular queue.

Queue



Enqueue O(1)

Dequeue O(n)

20/01/2022

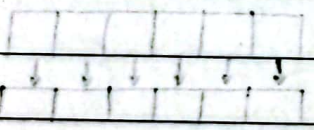
DEQUE

(Double-Ended Queue)

You can enque and deque at both ends.

Enq { Add-first()
Add-last()

Dq { Remove-first()
Remove-last()



Set (Unsorted/Unsorted)

collection of distinct objects.

add()

remove()

find()

size()

Amortized Analysis.

Append in not fully exhausted dynamic array is O(1)

Append in fully exhausted dynamic array is O(n)

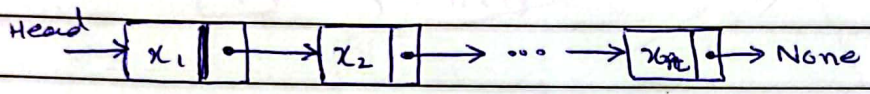
Date: 25/01/2022

M T W T F S S

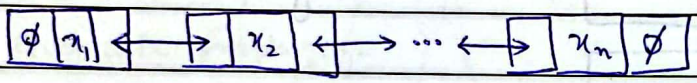
, because of resize ~~fun~~ of array.

27/01/2022

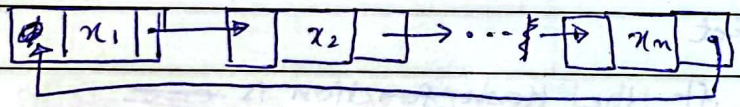
Linked List.



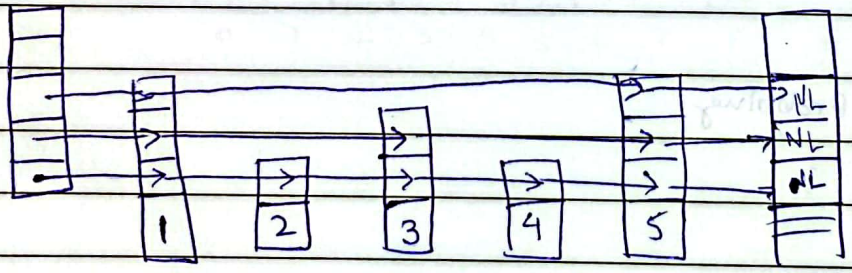
Doubly Linked List.



Circular Linked List.



Skip List

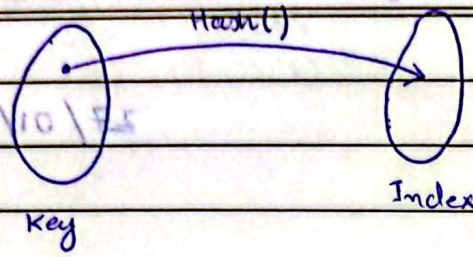


Hashing

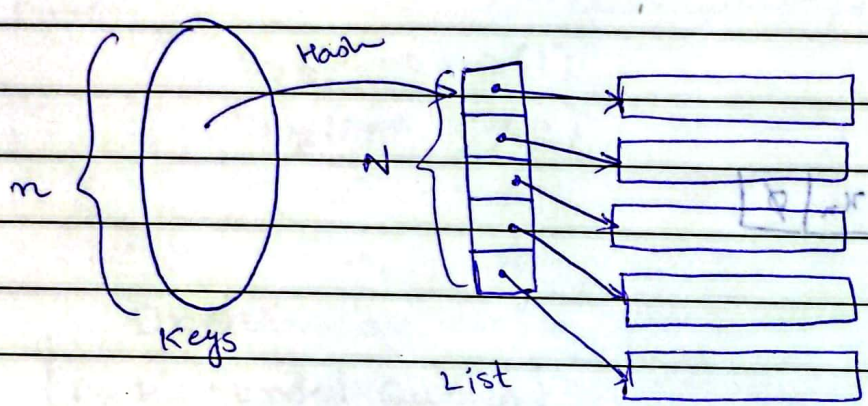
Hash(Key) ⟶ Index.

Date: 15/02/2022

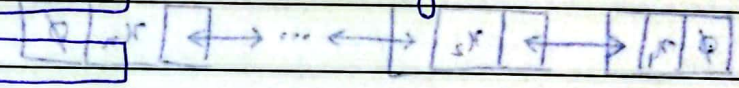
M	T	W	T	F	S	S

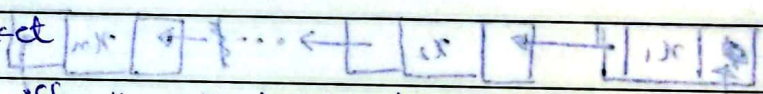


Collision, when 2 keys mapped to one Index.



Chaining.





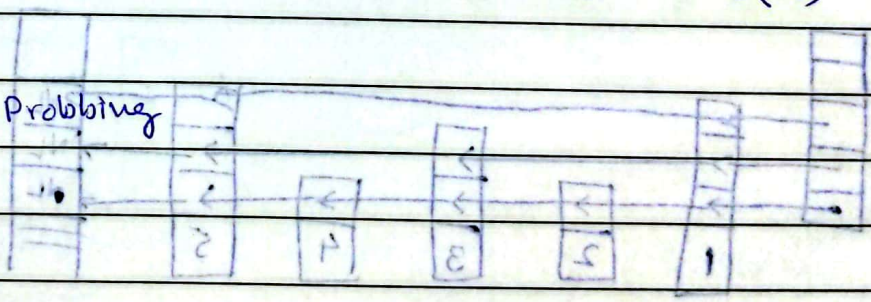
Size of Buckets will be equal iff the Hash function is ~~equal~~ evenly distribute the values. $O(n/N)$ on average.

In worst case all elements will be in one bucket that is $O(n)$

Open Address : Linear Probbing

Quadratic Probbing.

Double hashing.



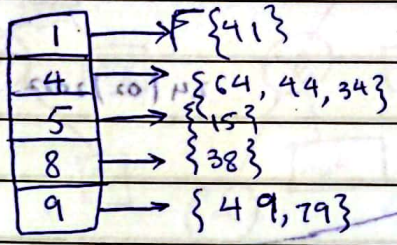
{49, 64, 38, 79, 41, 44, 15, 34}

$H(x) = x^3 \% 10$

Date: 17/02/2022

M	T	W	T	F	S	S

1. Chaining.



2. Linear Probing.

0	1	2	3	4	5	6	7	8	9
79	41			64	44	15	34	38	49

3. Qudratic Probbing.

0	1	2	3	4	5	6	7	8	9
79	41		34	64	44	15		38	49

4. Double hashing $H'(x) = \left\lfloor \frac{x}{2} \right\rfloor$ index = $\left(H(x) + i\,H'(x)\right) \% M$

0	1	2	3	4	5	6	7	8	9
	41	34		64	15	44	79	38	49

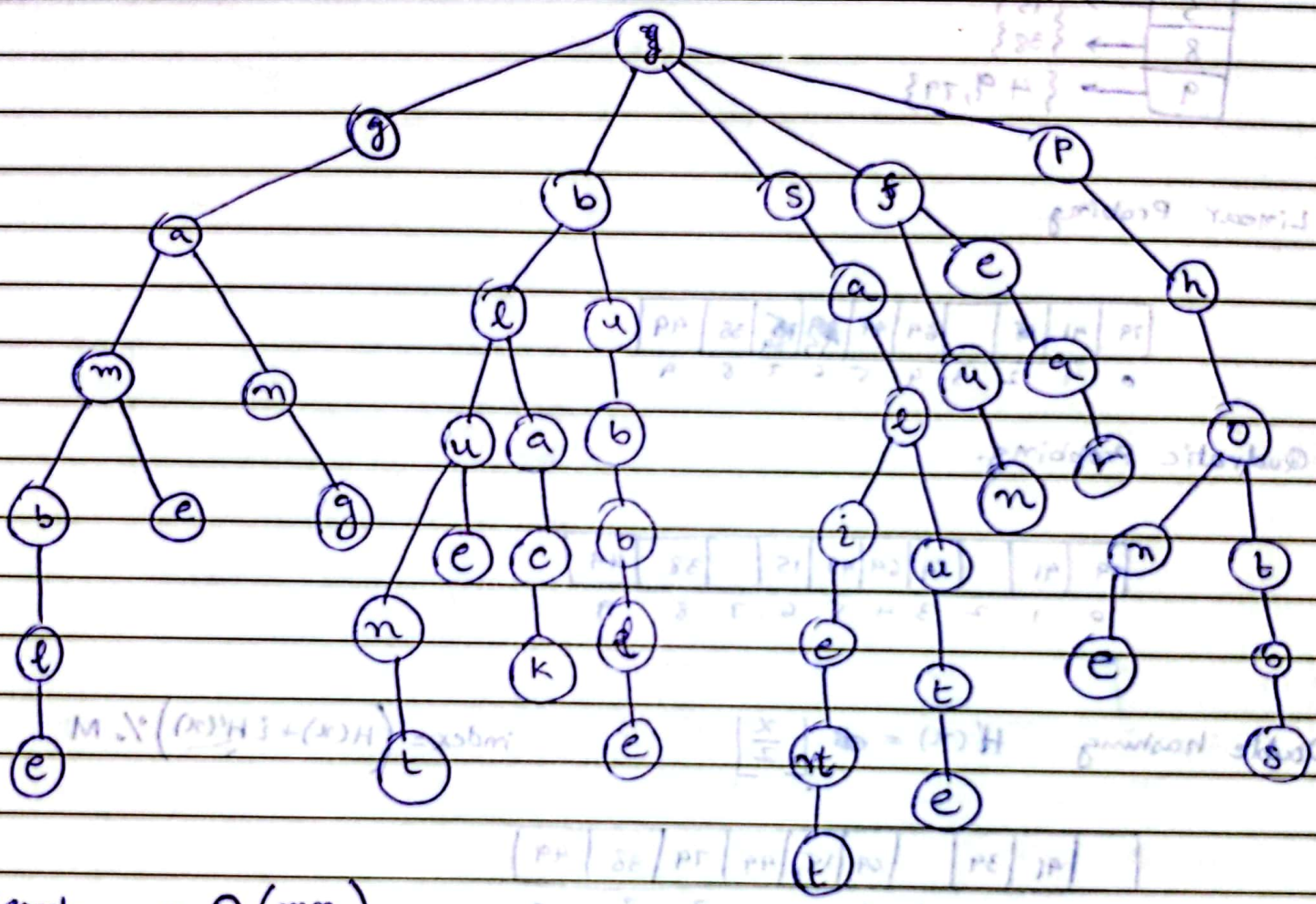
Tries

- A tree based structure.
- Supports fast pattern matching.

Date: 22/02/2022

M	T	W	T	F	S	S

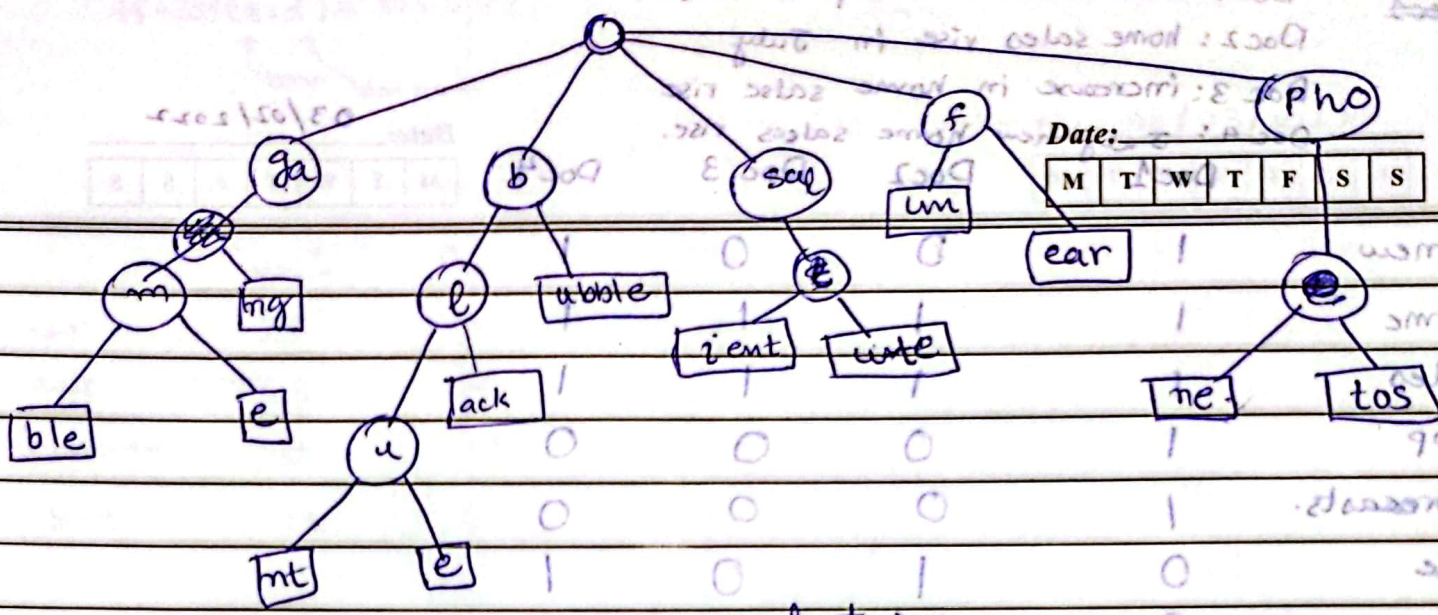
24/02/2022



Search complexity $= O(mn)$

m ← number of alphabets in the language

n ← size of word.



Date: M T W T F S S

compressed tries.

01/03/2022

Inverted Index.

Brutus → | 1 | 2 | 4 | 11 | 31 | 45 |

Ceasar → | 1 | 2 | 4 | 5 | 6 | 16 |

Calpurnia → | 2 | 31 | 54 | 101 |

Dictionary

Posting.

Each word in a document is called document.

Stop words

Abbreviations.

Maintaing Synonyms

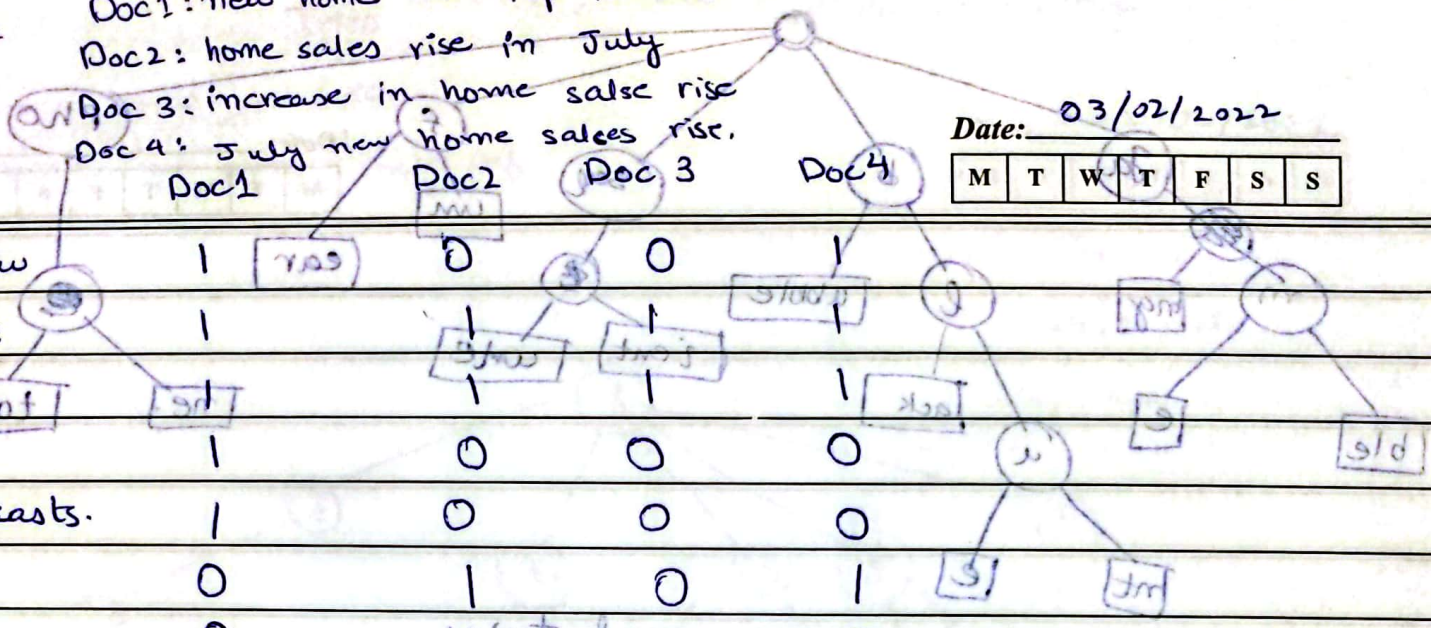
Stemming → use root words, by chopping ends.

Lemmatization → use root words.

Doc1: new home sales top forecasts.
Doc2: home sales rise in July
Doc 3: increase in home salse rise
Doc 4: July new home sales rise.

Date: 03/02/2022

M	T	W	T	F	S	S



	Doc1	Doc2	Doc 3	Doc4
new	1	0	0	1
home	1	1	1	1
sales	1	1	1	1
top	1	0	0	0
forecasts.	1	0	0	0
rise	0	1	0	1
in	0	1	1	0
july	0	1	1	1
increase	0	0	1	0

Inverted Index.

new → 1, 4
home → 1, 2, 3, 4
sales → 1, 2, 3, 4
top → 1
forecasts → 1
rise → 2, 4
in → 2, 3
july → 2, 3, 4
increase → 3

Ranking Document.

Document Freq. = $\left(\frac{df(term)}{N}\right)$ ← document freq. of term.
N ← size / total number of document.

Inv. DF = $\log\left(\frac{1}{DF}\right)$

Signature _____

$tf\text{-}idf(t,d) = tf(t,d) \times idf(t)$

(t: term, d: document)

Date: 08/03/2022

M	T	W	T	F	S	S

	Doc 1	Doc 2	Doc 3
car	27	4	24
auto	3	33	0
insurance	0	33	29
best	14	0	17

$tf\text{-}idf(auto, Doc1) = \frac{3}{44} \times \log\left(\frac{3}{2}\right) = 0.012006$

$tf\text{-}idf(auto, Doc2) = \frac{33}{70} \times \log\left(\frac{3}{2}\right) = 0.08301$

$tf\text{-}idf(auto, Doc3) = 0$

Cosine similarity.

$$\cos(\vec{a}, \vec{b}) = \frac{\vec{a} \cdot \vec{b}}{\|\vec{a}\|\|\vec{b}\|} = \hat{a} \cdot \hat{b}$$

10/03/2022

Precision = $\frac{\#\text{ of true positives retrived}}{\#\text{ of documents retrived}}$

Recall = $\frac{\#\text{ of true positive retrived}}{\#\text{ of all true positives}}$

F-measure = $\frac{2RP}{R+P}$

Page No. _____ Signature _____

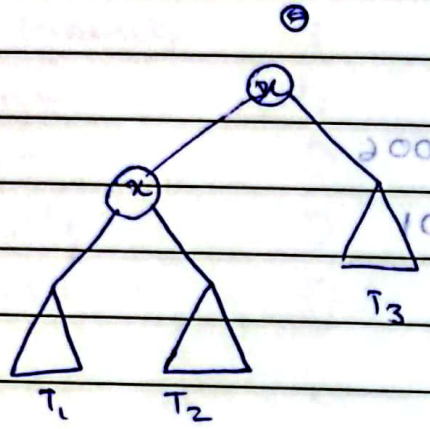
Treap : Randomized Binary Search Tree

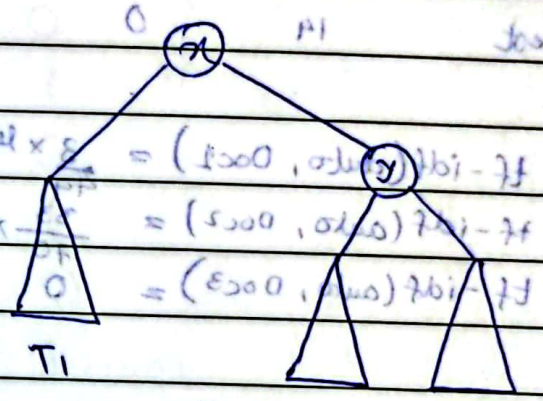
1. Key follow standard BST ordering .

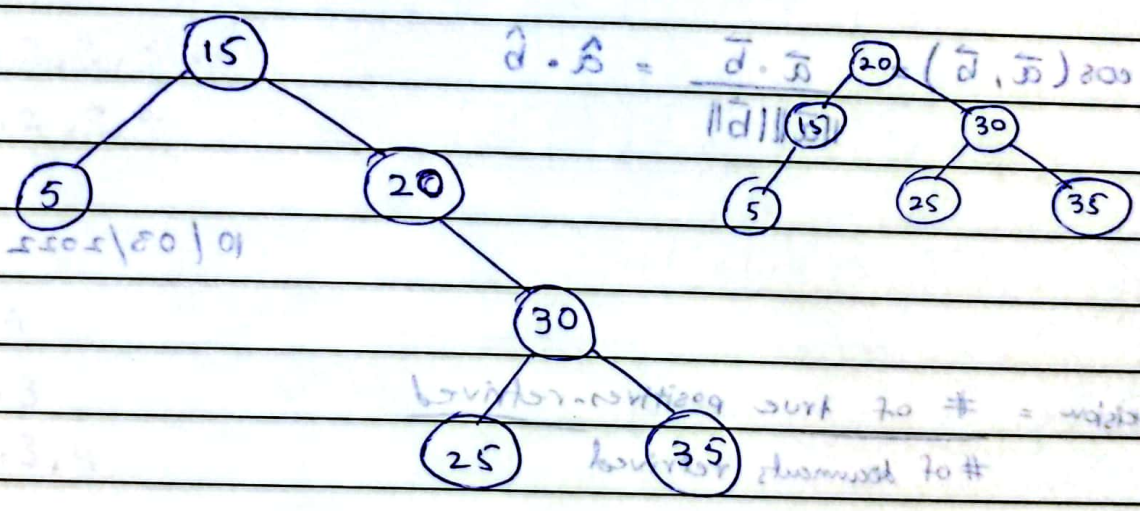
Date: 15/03/2022

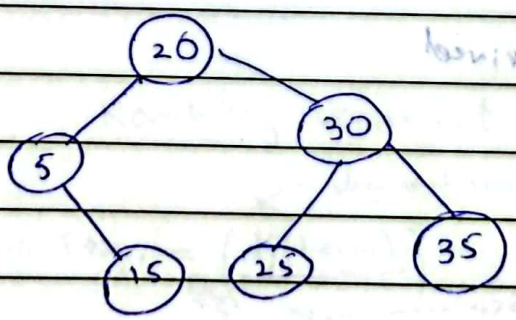
M	T	W	T	F	S	S

2. Priority randomly assigned value that follows Max-Heap property.







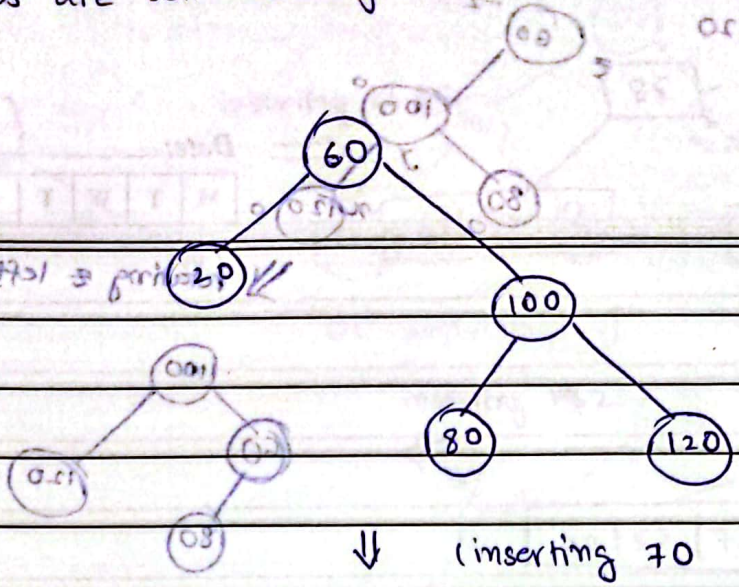


~~AVL~~ Trees are self-balancing trees.
AVL

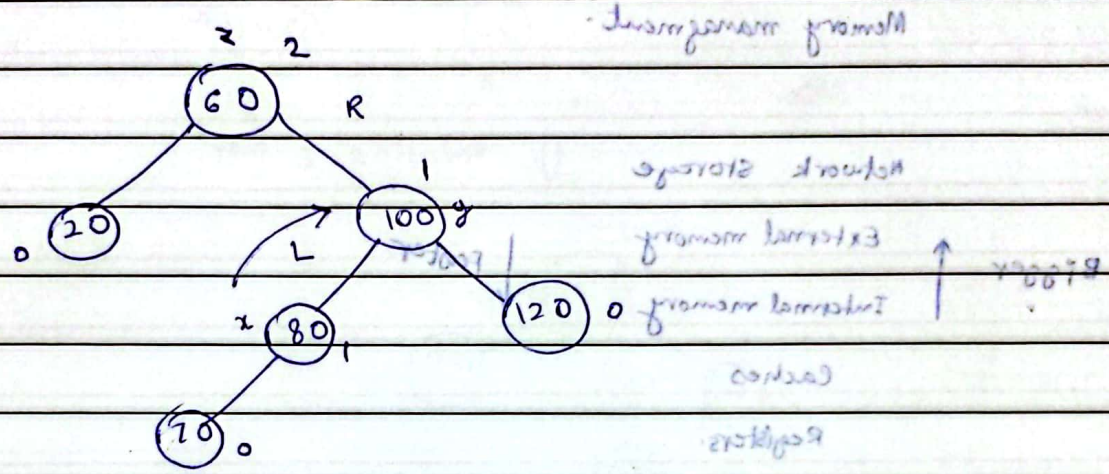
Date: 29/03/2022

M	T	W	T	F	S	S

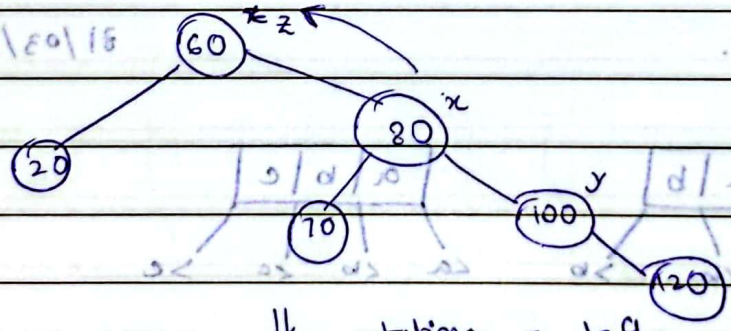
: AVL - Tree



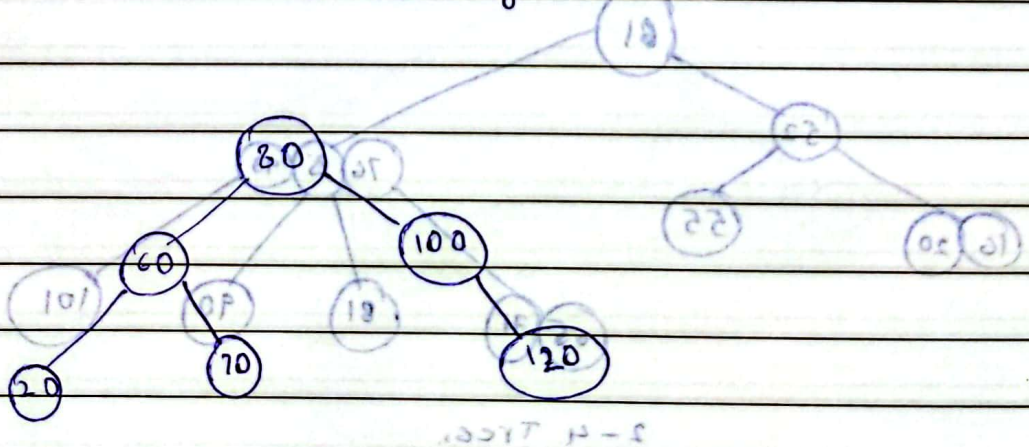
⇓ (inserting 70



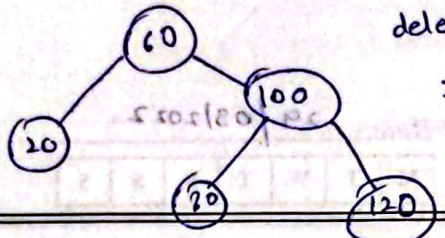
⇓ rotating y right.



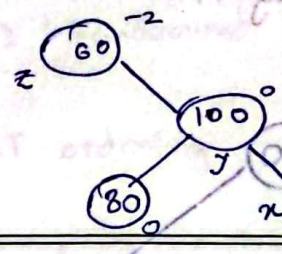
⇓ rotating z left.



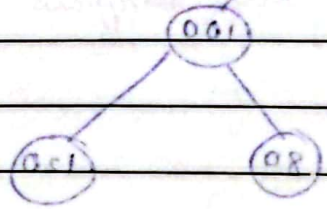
Page No. _____ RC Signature _____



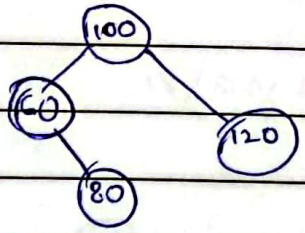
deleting 20 ⇒



Date: M T W T F S S



rotating ⇓ left.



Memory managment.

Network Storage

External memory

Bigger ↑ ↓ Faster

Internal memory

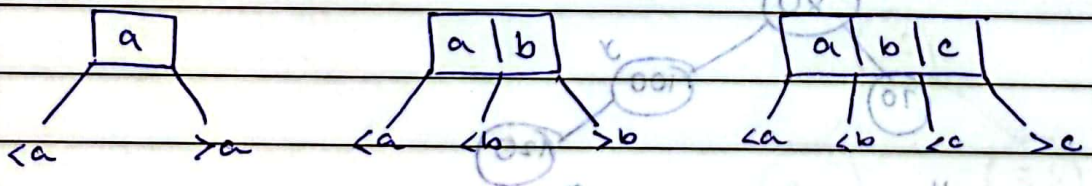
Caches

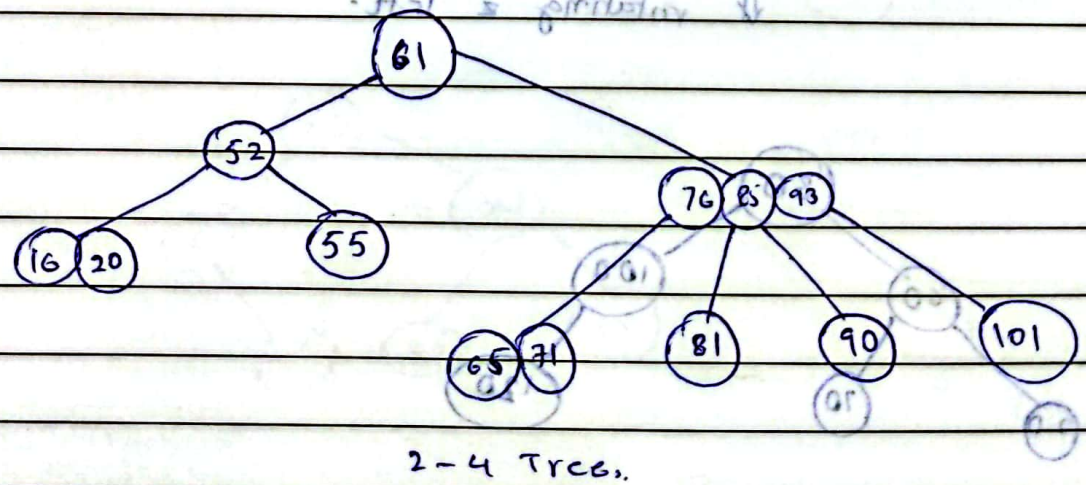
Registers.

CPU

B-Trees.

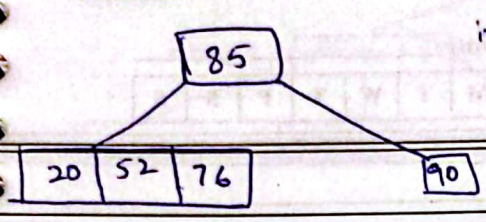
31/03/2022



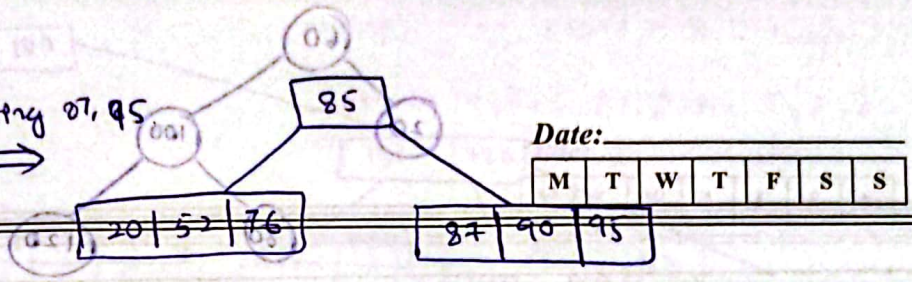


2-4 Trees.

Splitting.



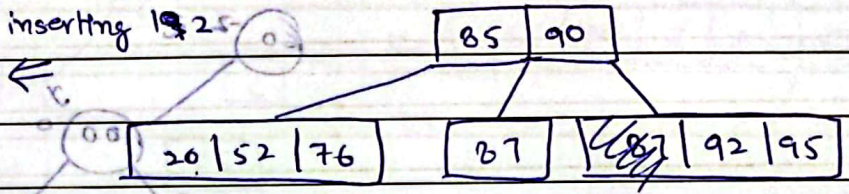
inserting 87, 95 ⟹



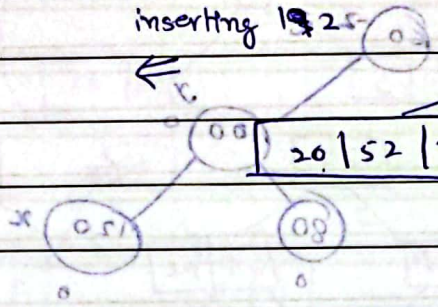
Date:

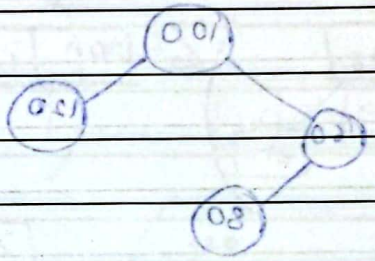
M	T	W	T	F	S	S

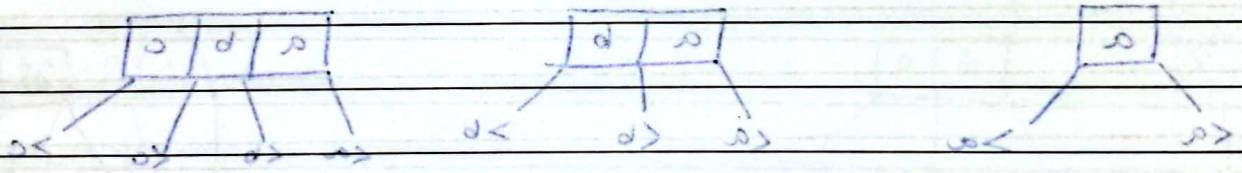
inserting 92

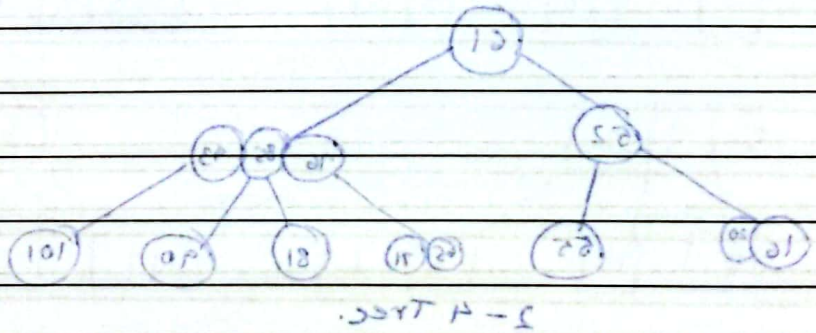


inserting 25





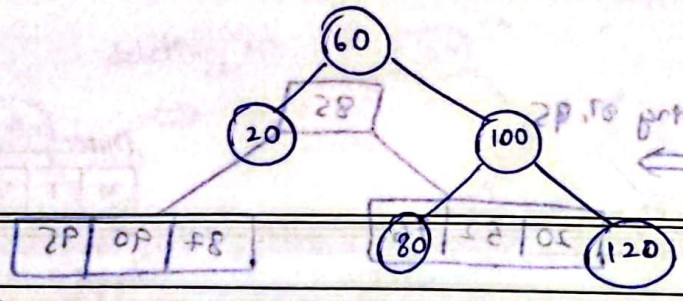




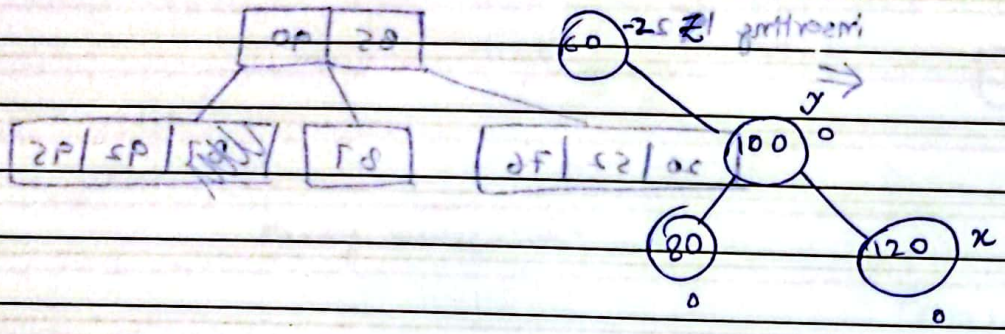
Page No. _____ RC Signature _____

Date:

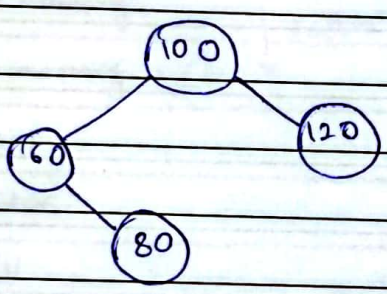
M	T	W	T	F	S	S



⇓ deleting 20.

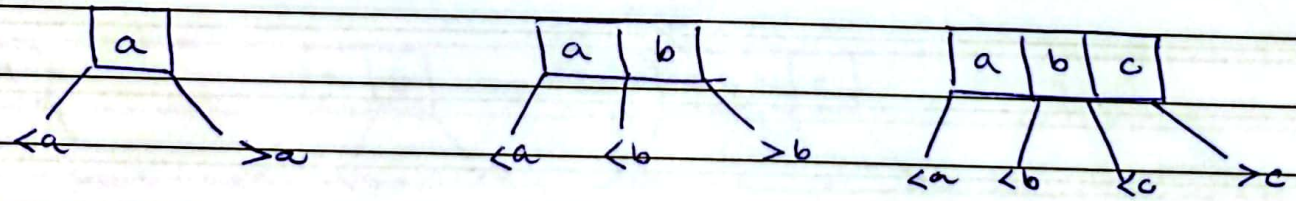


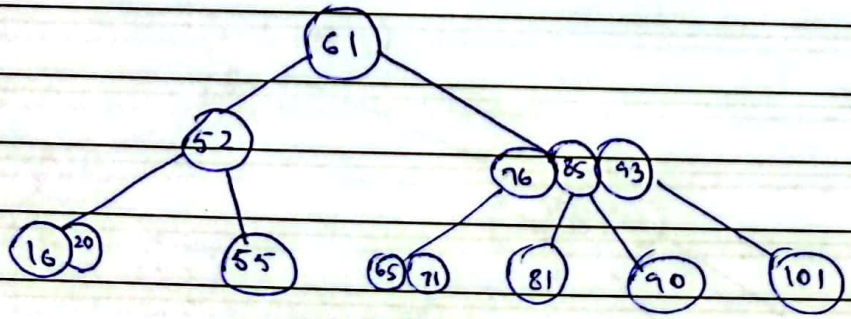
⇓ rotating z left.



B-Tree

31/03/2022





2-4 Tree.

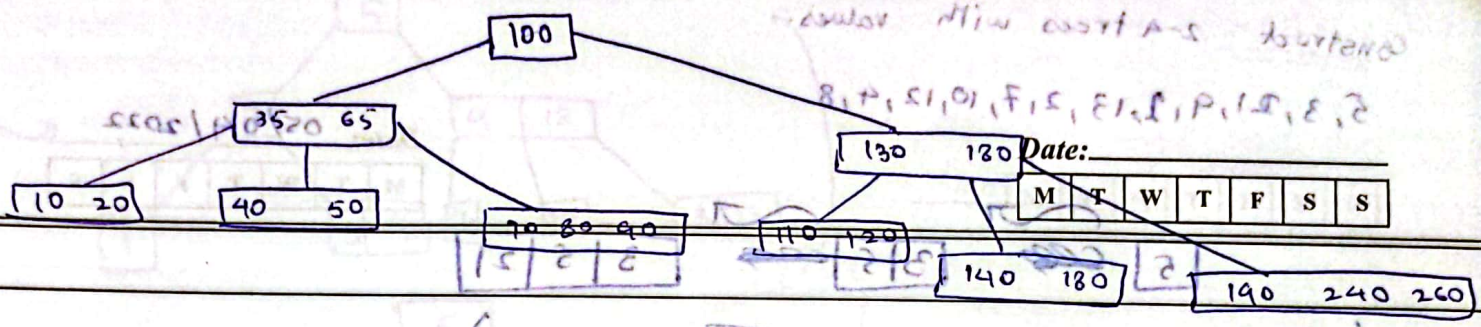
Page No. _____

RC

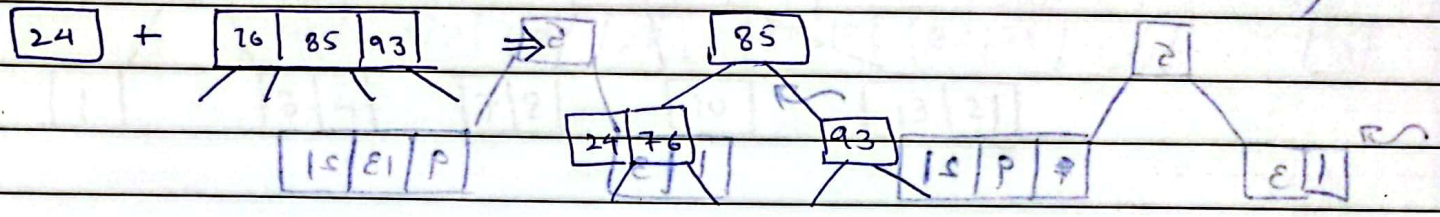
Signature _____

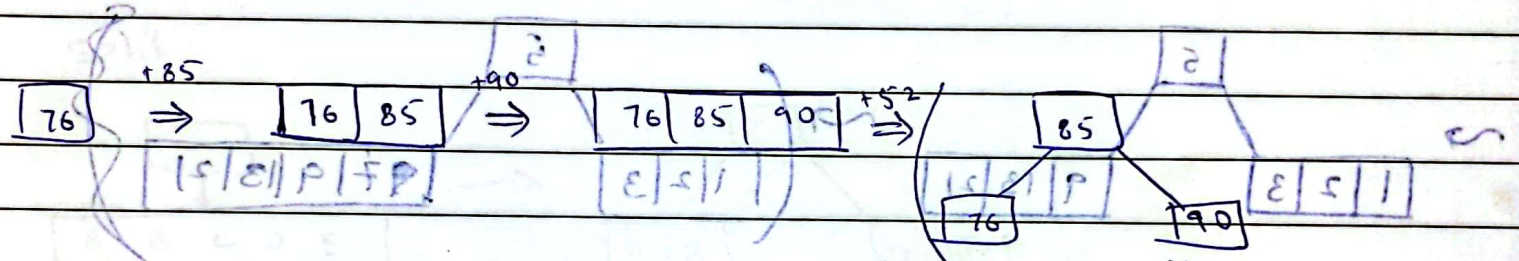
Date:

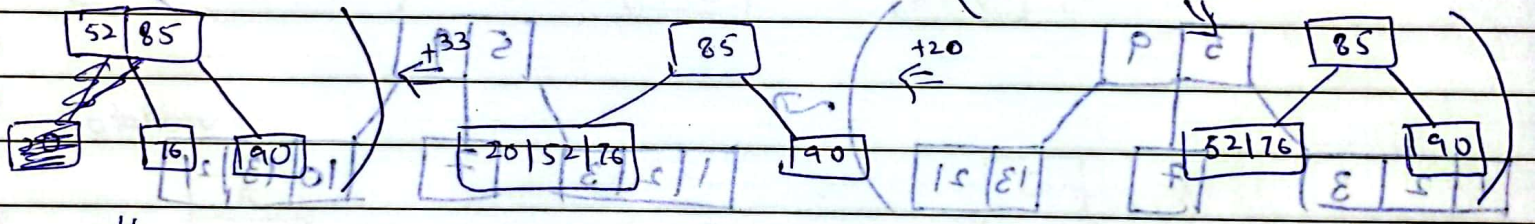
M	T	W	T	F	S	S

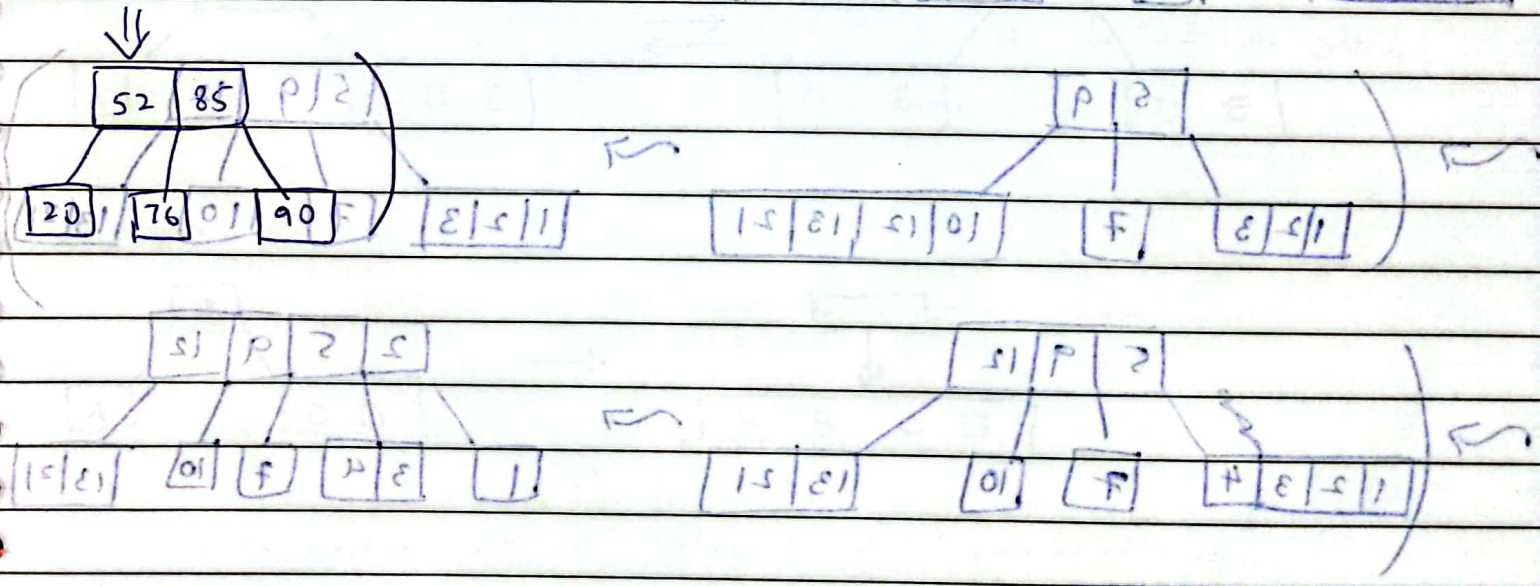


Splitting.







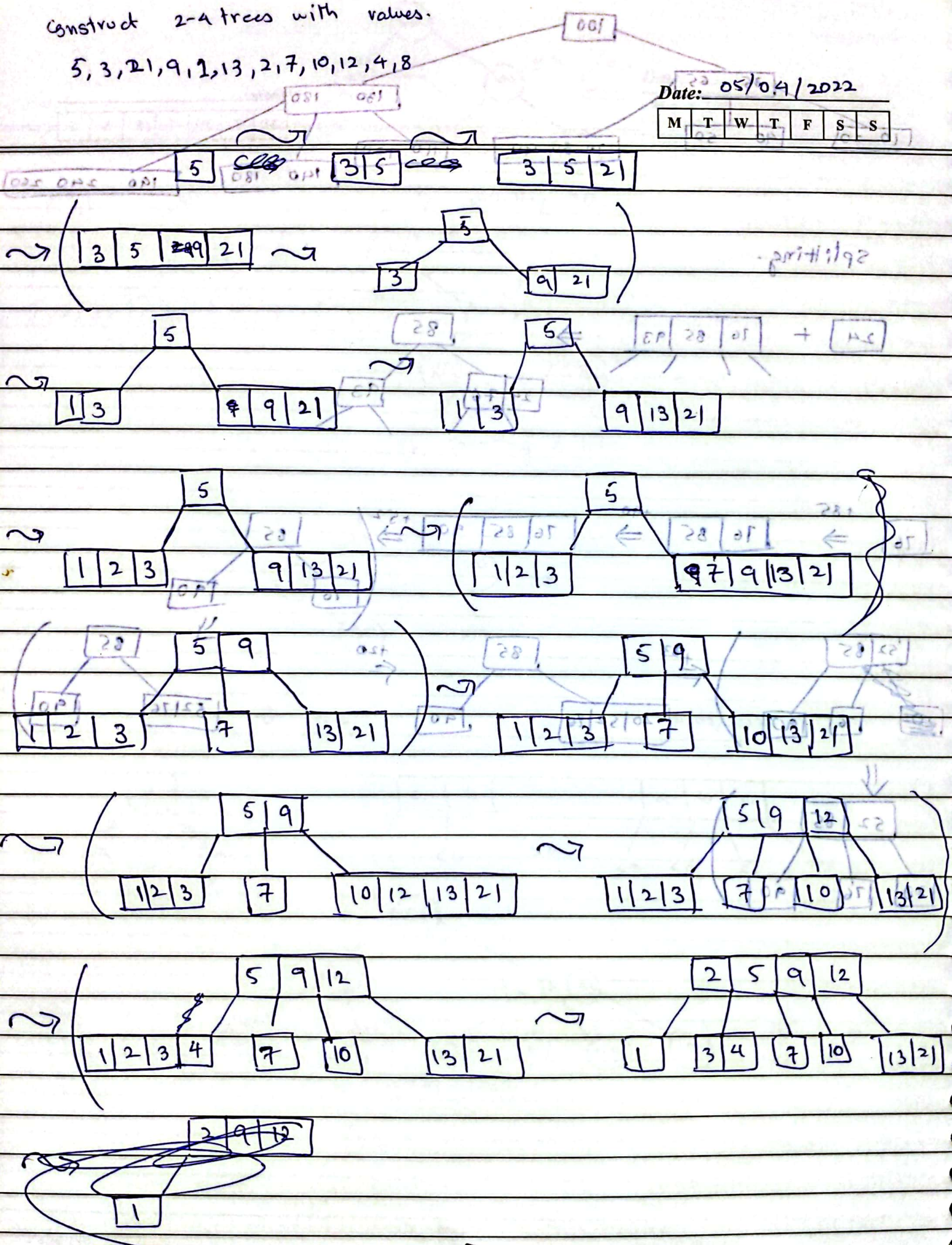


Construct 2-4 trees with values.

5, 3, 21, 9, 1, 13, 2, 7, 10, 12, 4, 8

Date: 05/04/2022

M	T	W	T	F	S	S

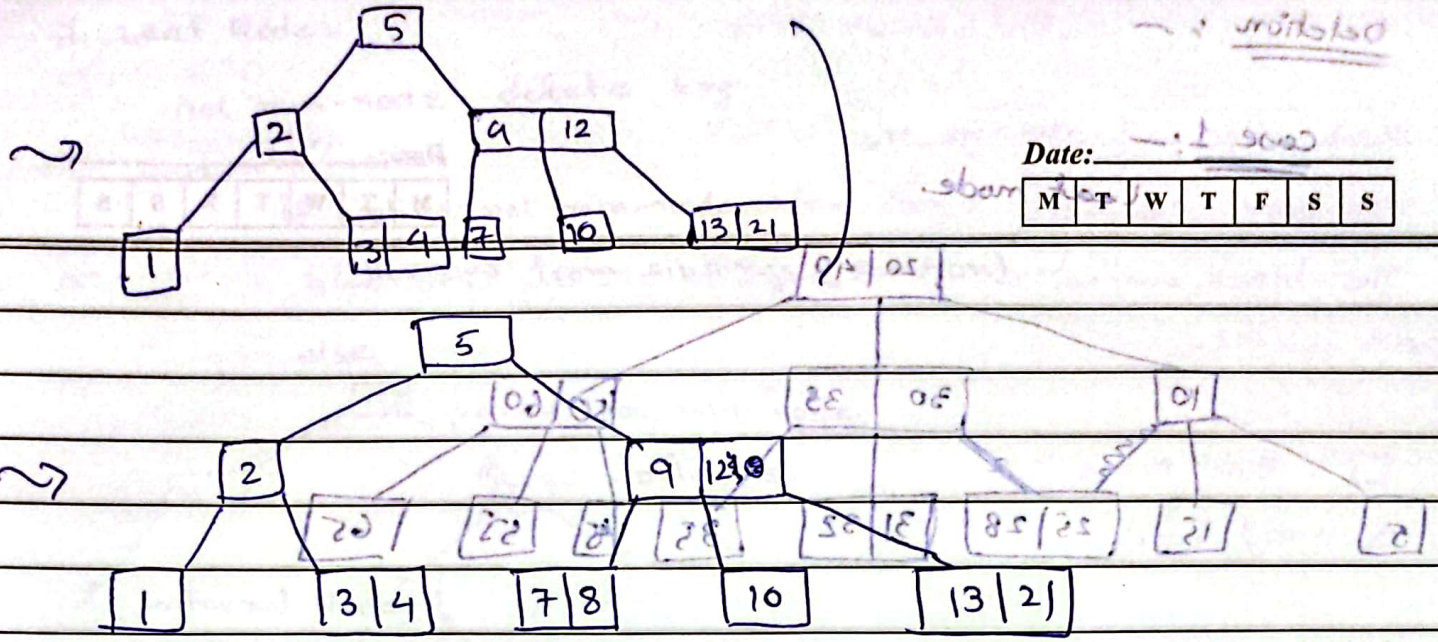


Page No. _____

RC

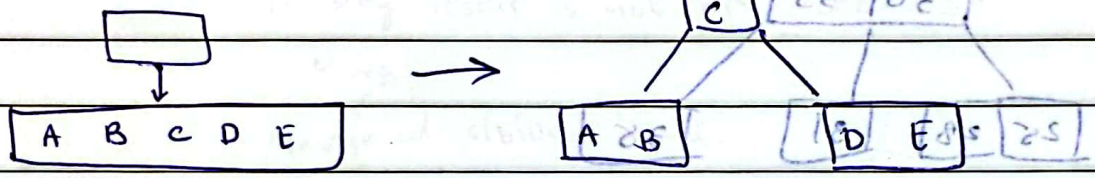
Signature _____

Date: M T W T F S S

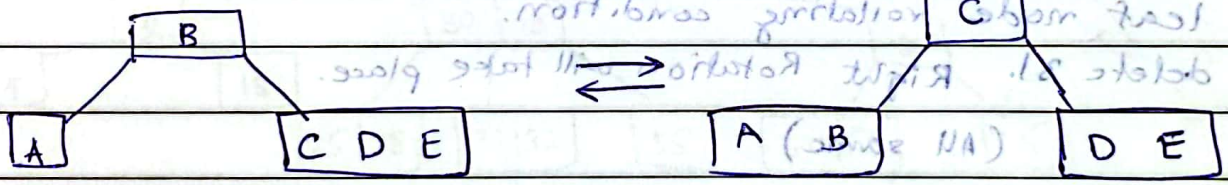


delete 31.

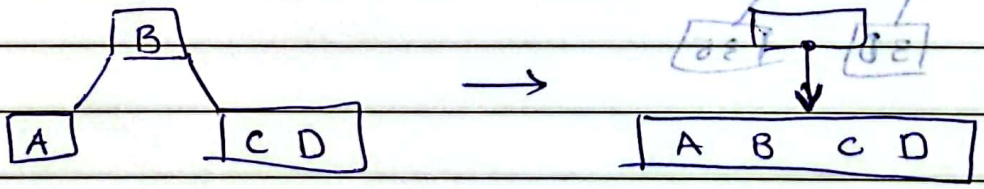
Split.



Rotation



Merge

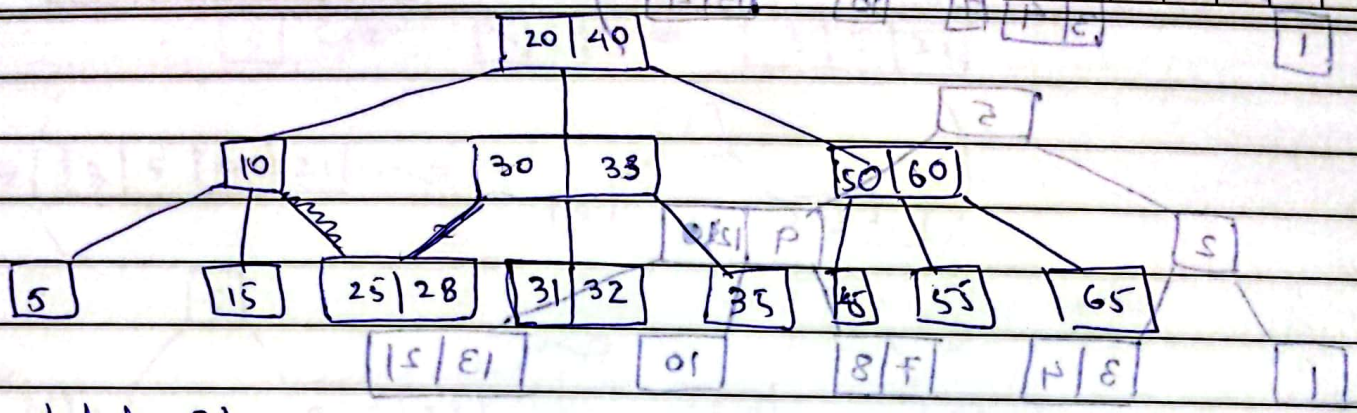


Page No. _____ Signature _____

Deletion :-

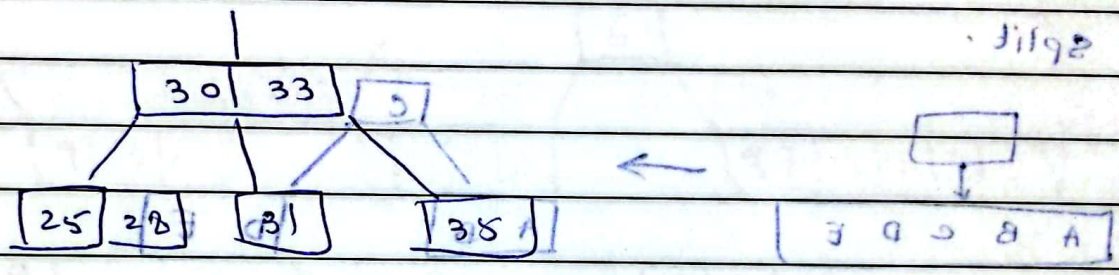
Case 1:- Leaf node

Date: M T W T F S S



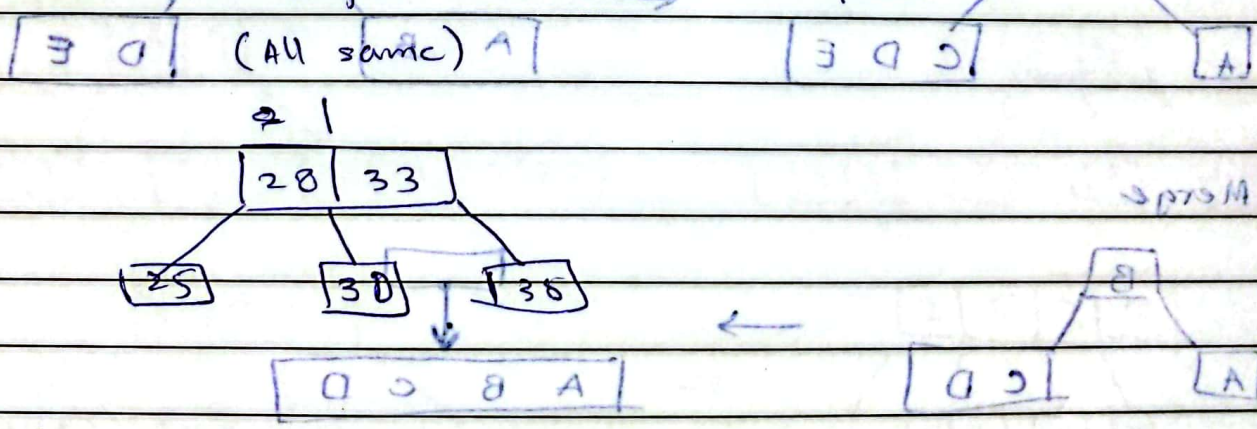
delete 31.

(All same)



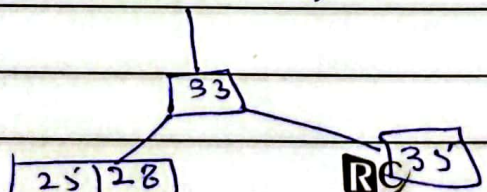
Case 2:- Leaf node voilating condition.

delete 31. Right Rotation will take place.



Case 3:- When siblings has no extra member. delete 30.

(All same)



Page No. _____ Signature _____

if Leaf Node:

Not min-node: delete key

if min-node:

if sibling not min-node:

Borrows from sibilings (Rotation)

else

if parent not min-node

Merge sibilings.

if Internal Node:

if Child not Min-Node.

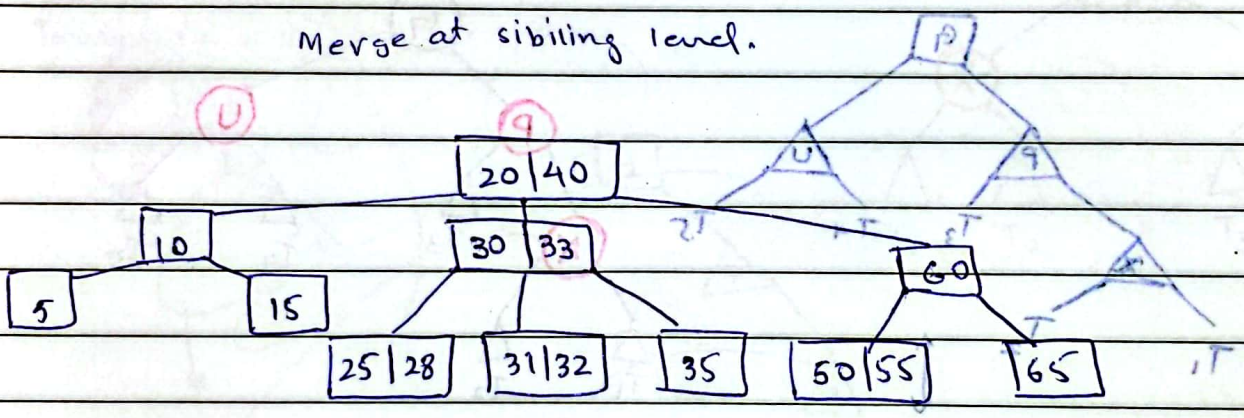
Replace by predecessor / succesor.

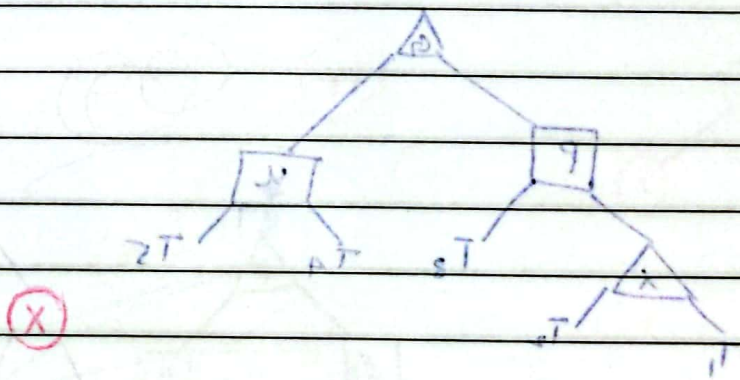
else

if key itself is not min-Node

Merge

Merge at sibiling level.





Red - Black Tree.

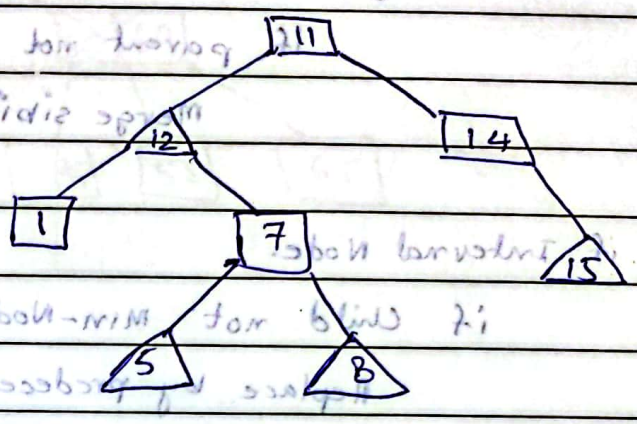
Date: 12/04/2022

M	T	W	T	F	S	S

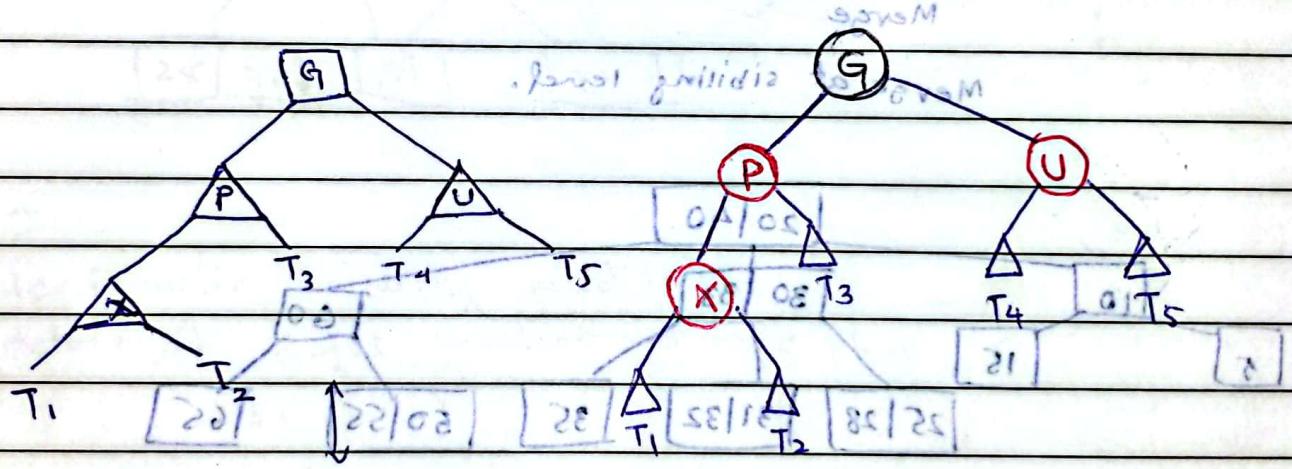
- Root node will always be black.
- Red node has only black nodes.
- The black ancestors of nodes with one ore zero child nodes are same.

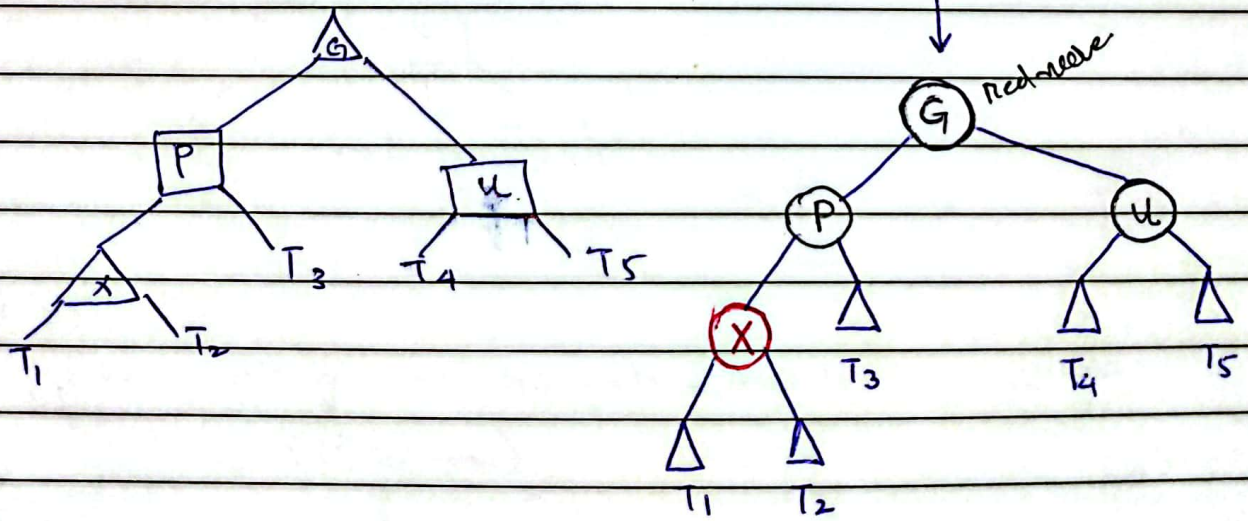
□ = Black Nodes

△ = Red nodes.

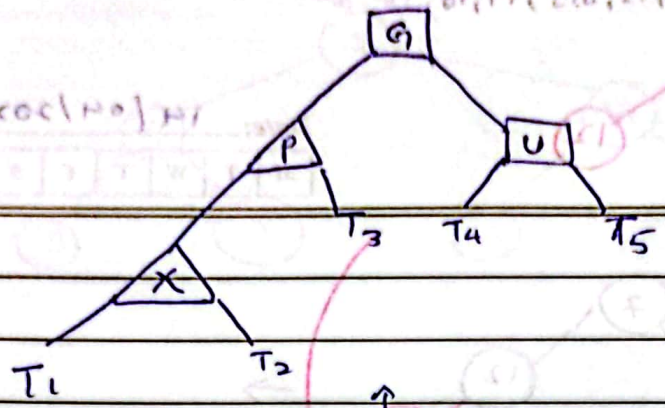


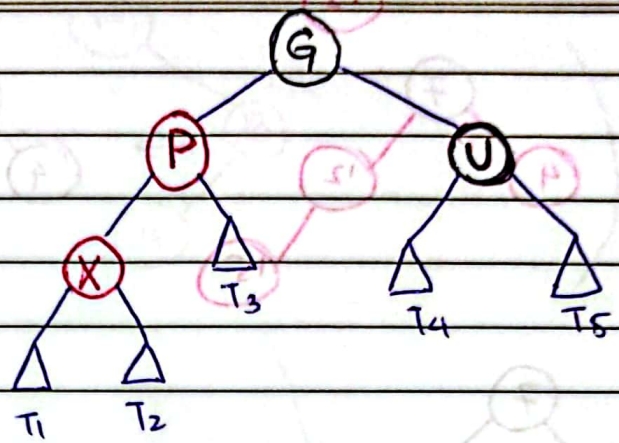
Recoloring.

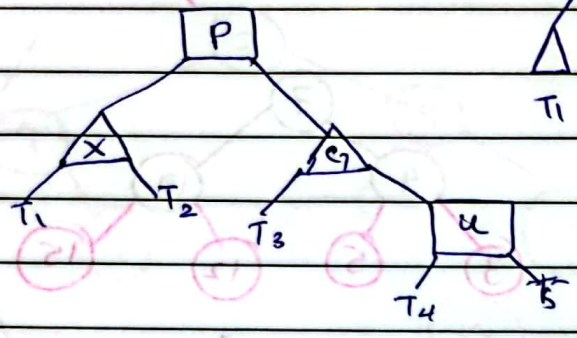


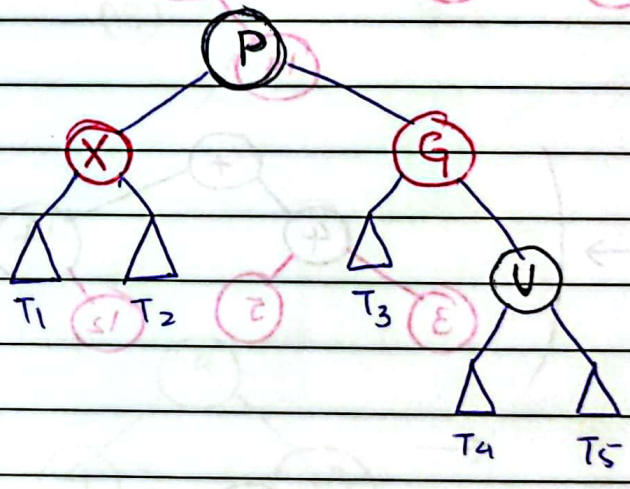


Date: M T W T F S S

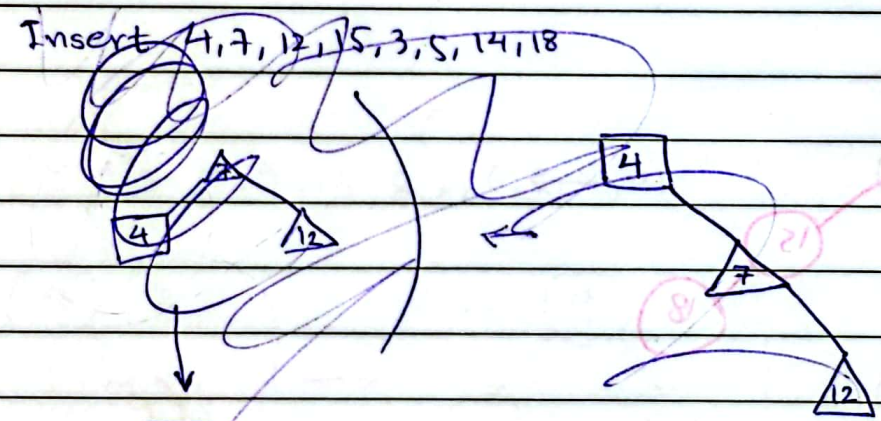


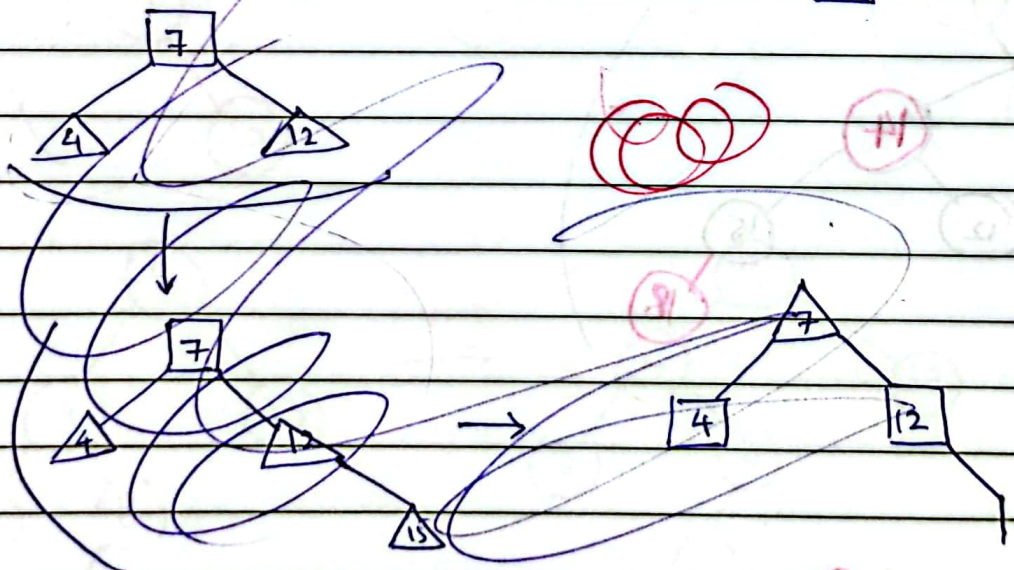






Insert 4, 7, 12, 15, 3, 5, 14, 18



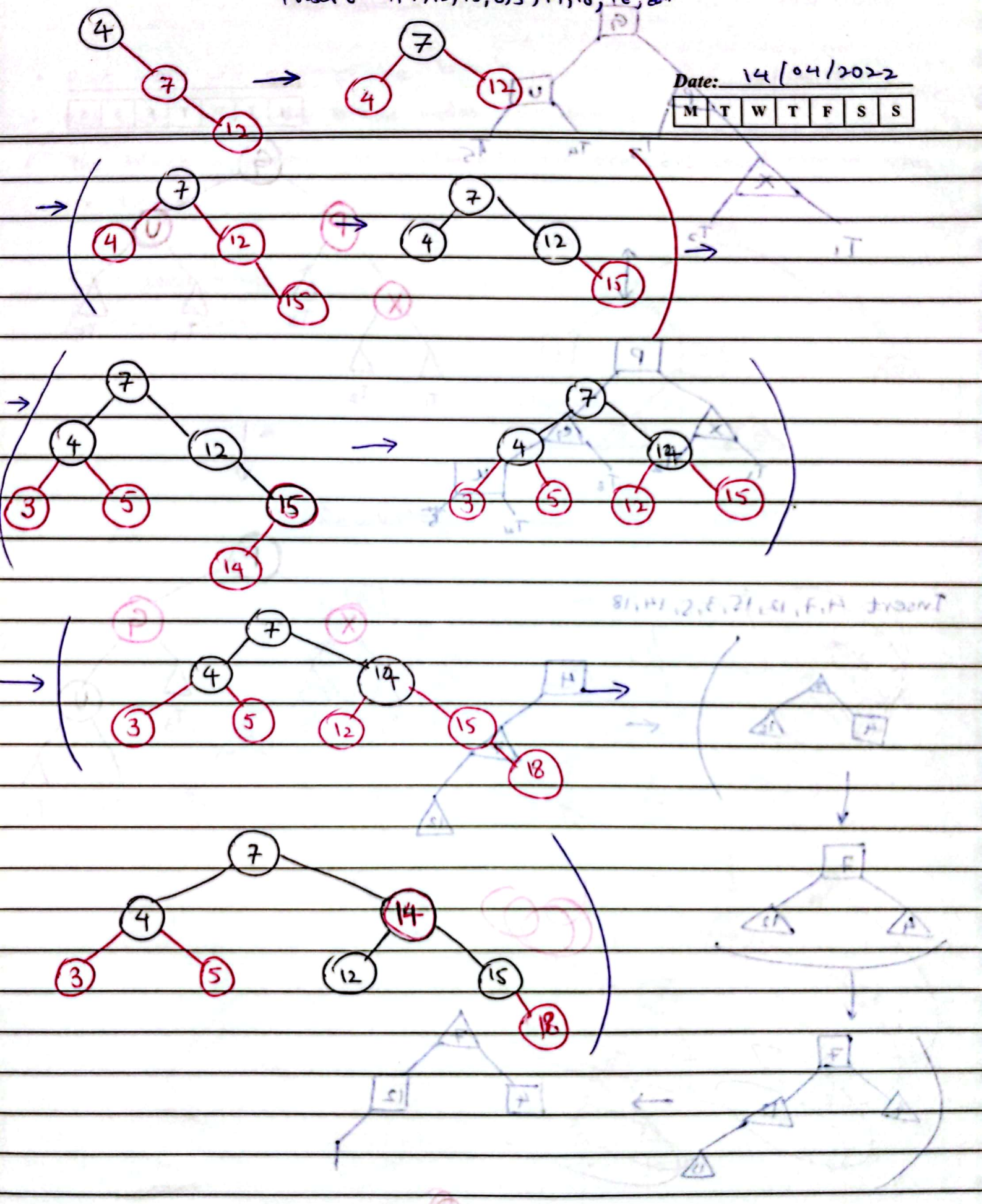


Page No. _____ Signature _____

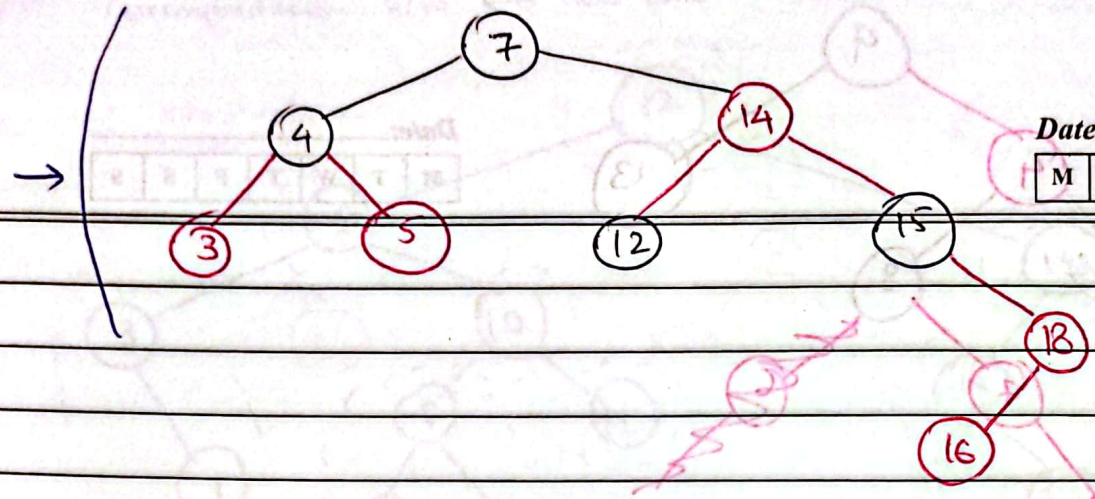
Insert 4, 7, 12, 15, 3, 5, 14, 18, 16,

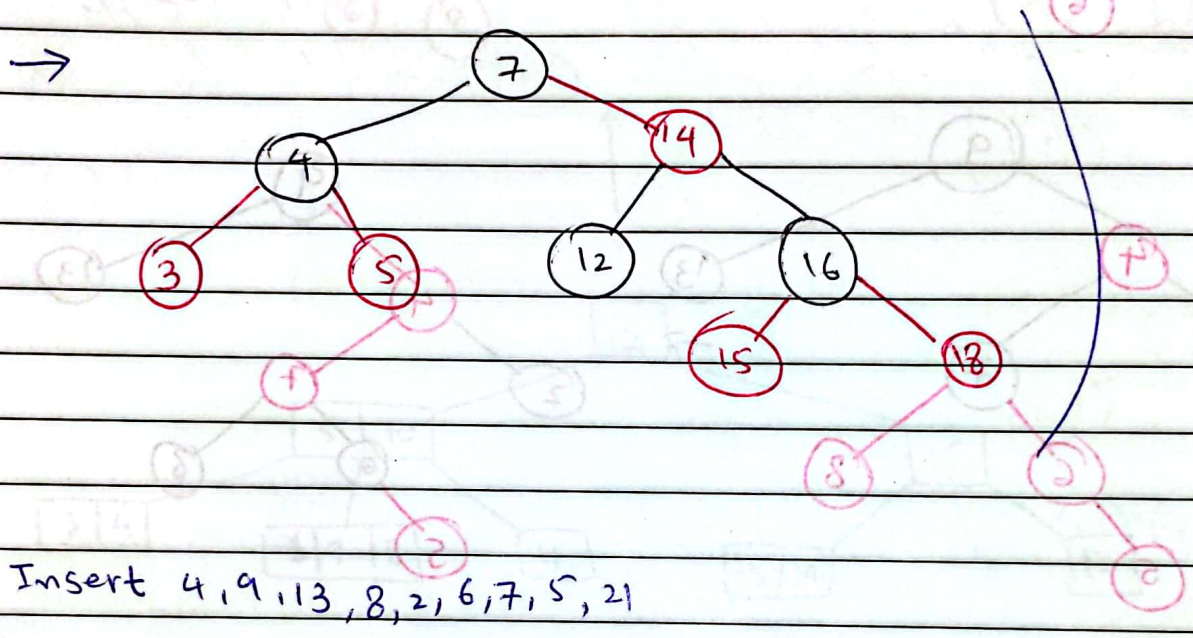
Date: 14/04/2022

M	T	W	T	F	S	S

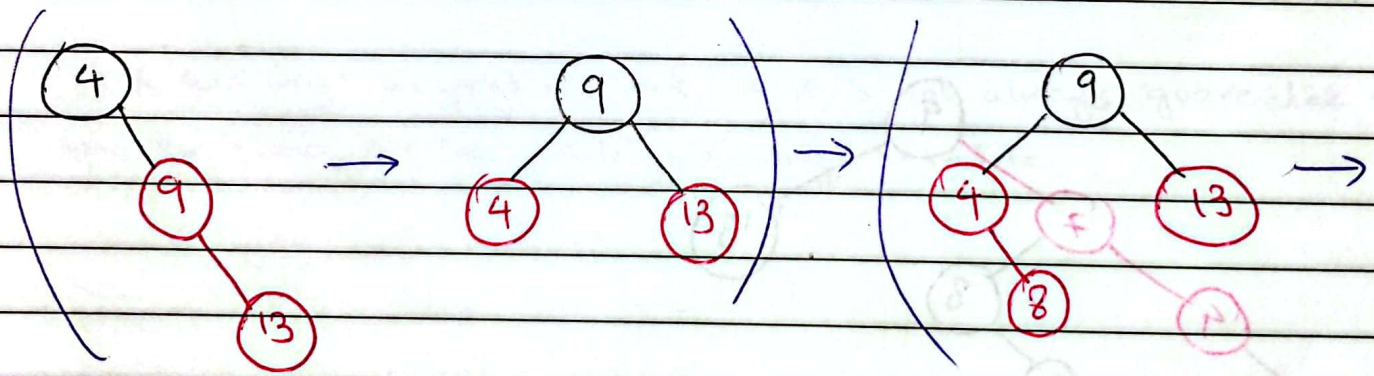


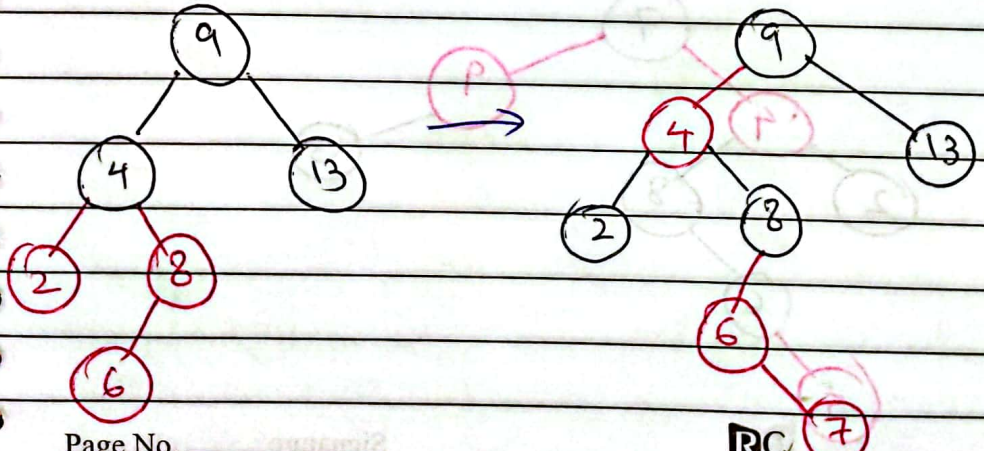
Date: M T W T F S S





Insert 4, 9, 13, 8, 2, 6, 7, 5, 21

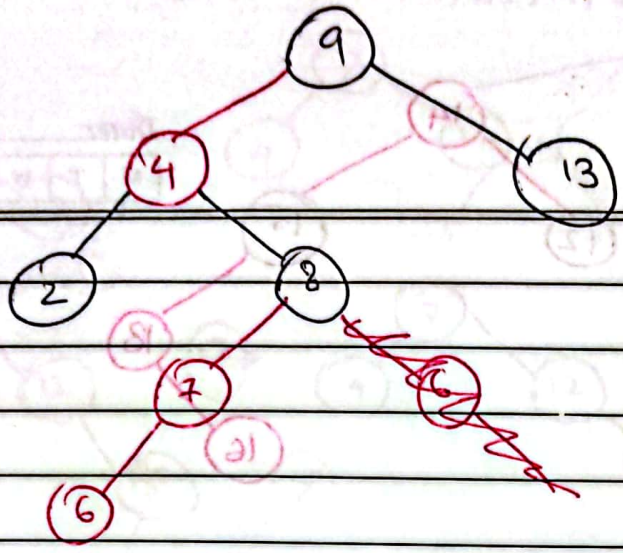


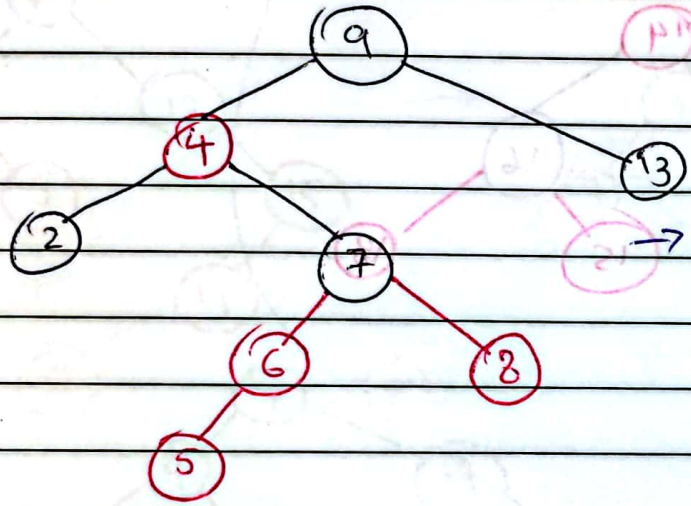


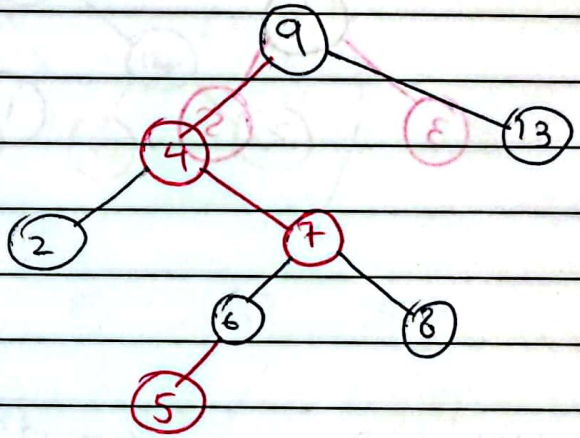
Page No. _____ RC Signature _____

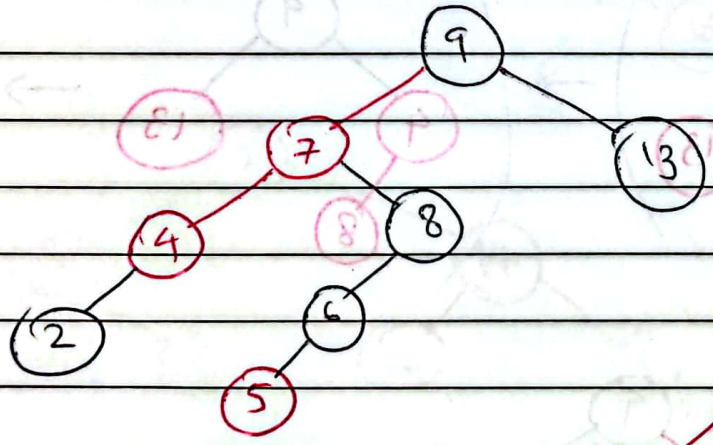
Date: _____

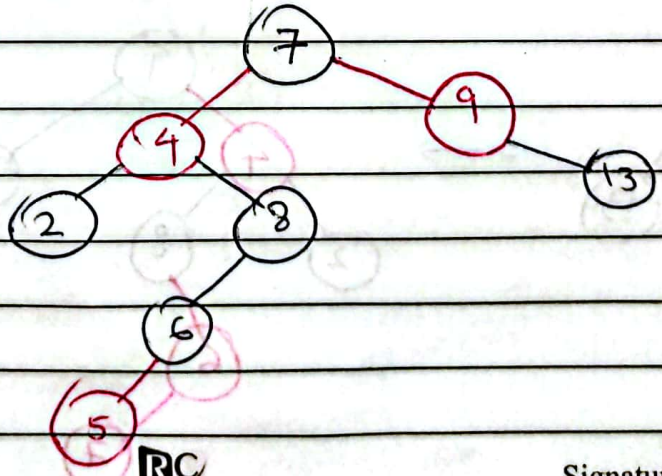
M	T	W	T	F	S	S











Page No. _____

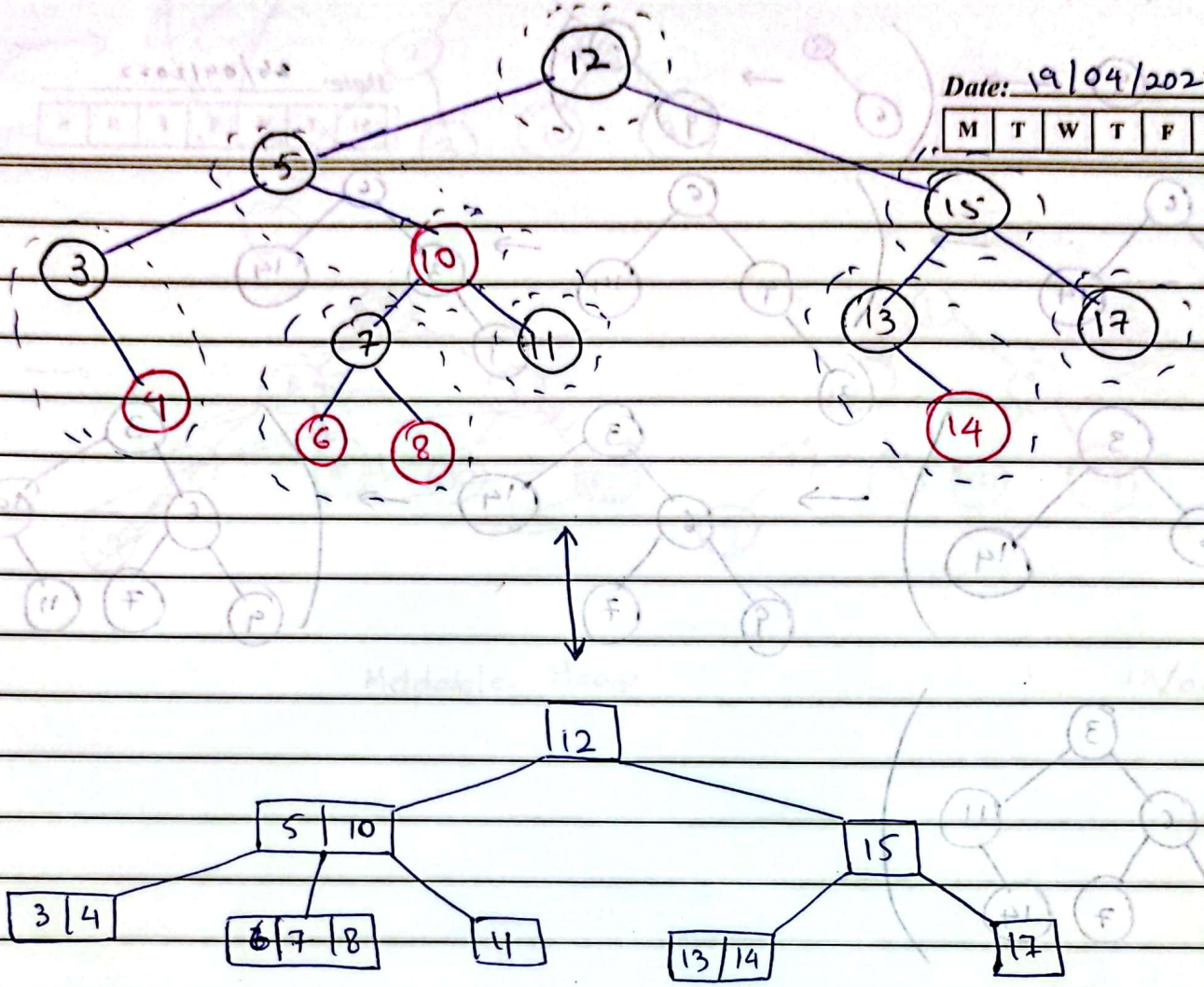
RC

Signature _____

Correspondence b/w R-B Tree and 2-4 tree.

Date: 19/04/2022

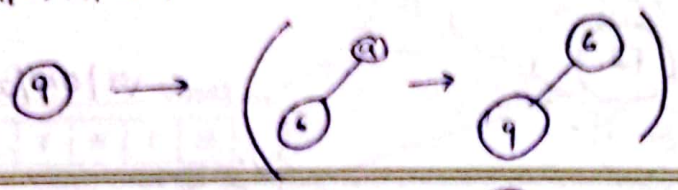
M	T	W	T	F	S	S



It is two ways correspondence, but it is not always guarentee to have the same R-B Tree, while converting 2-4-Tree.

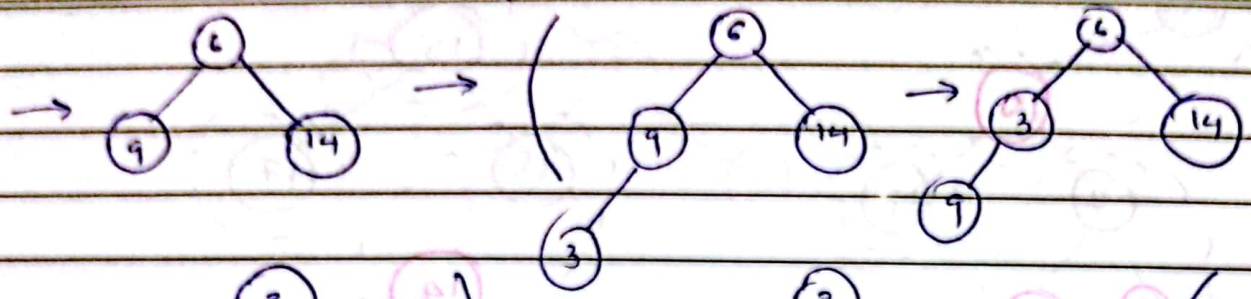
Min Heap.

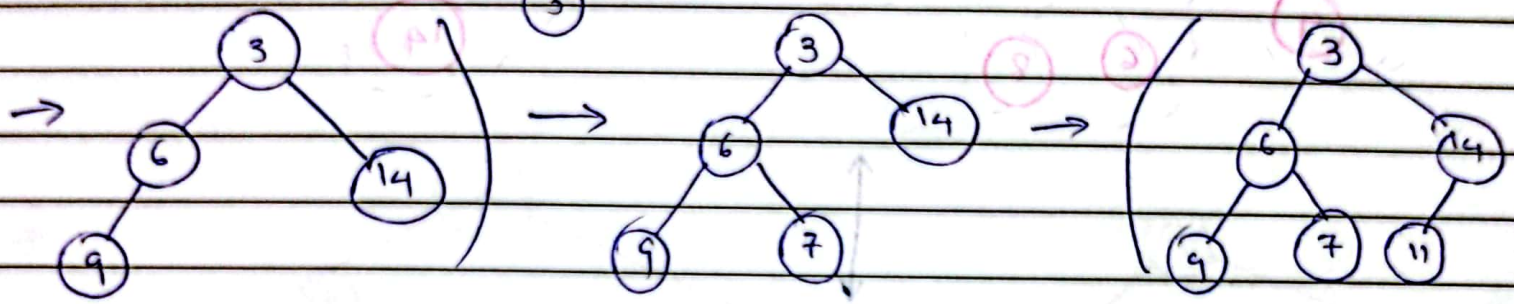
Insert 9, 6, 14, 3, 7, 11 in min heap.

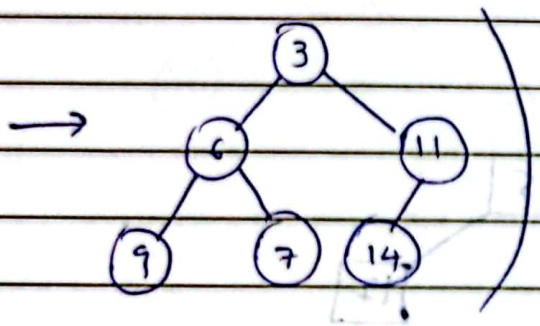


Date: 26/04/2022

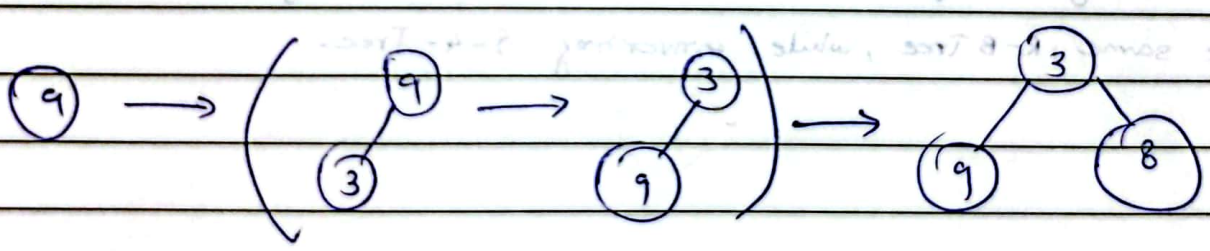
M	T	W	T	F	S	S

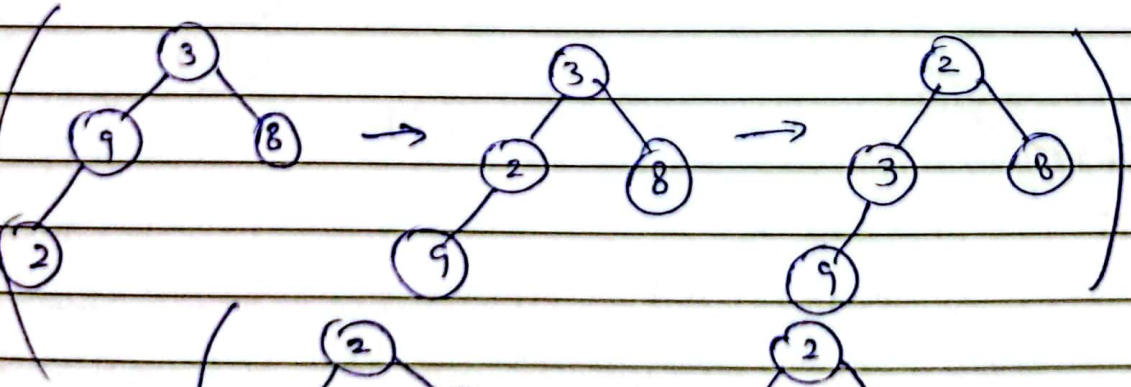


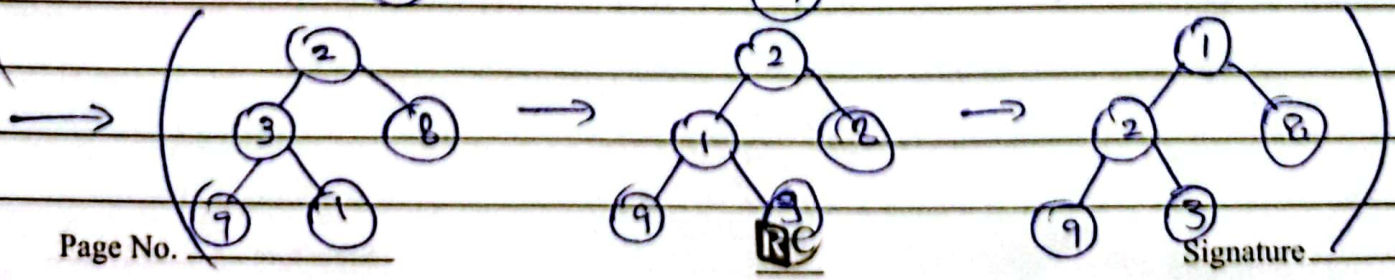


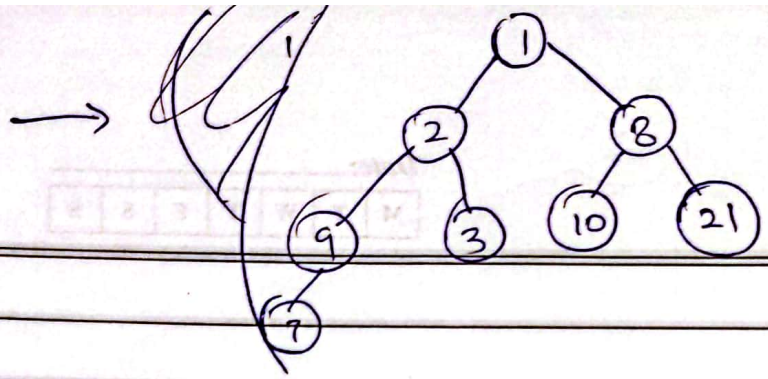


Insert 93, 8, 2, 1, 10, 21, 7



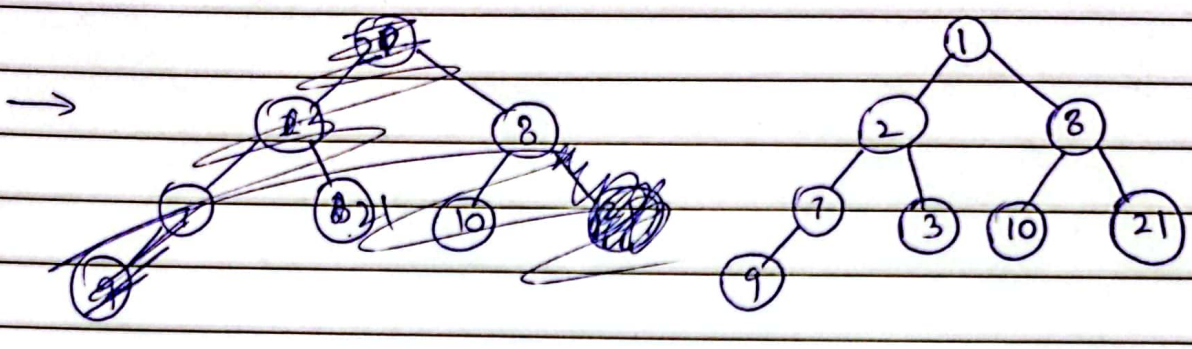






Date:

M	T	W	T	F	S	S



Meldable Heap.

28/04/2022