GIT Department of Computer Engineering
CSE 222/505 - Spring 2022
Homework # Report

Student Name: Muhammed Alperen Karaçete
Student Number: 171044052

1. SYSTEM REQUIREMENTS

User:

In Question 3:

User can get depth of tree.

User can get height of node in tree.

User can see if tree is a full tree.

User can see if tree is a complete tree.

User can insert an element to tree.

User can change a value in a node .

In Question 4:

User can insert new elements to tree.

User can search a specific element in tree.

User can search and get reference a specific element in tree.

User can remove element from tree.

User can remove element from tree and get this element.

2. CLASS DIAGRAMS

I could not to part 3 and 4.

3. PROBLEM SOLUTION APPROACH

In 1st question:

I explained it on bottom part.

In 2nd question:

I explained it on bottom part.

4. TEST CASES

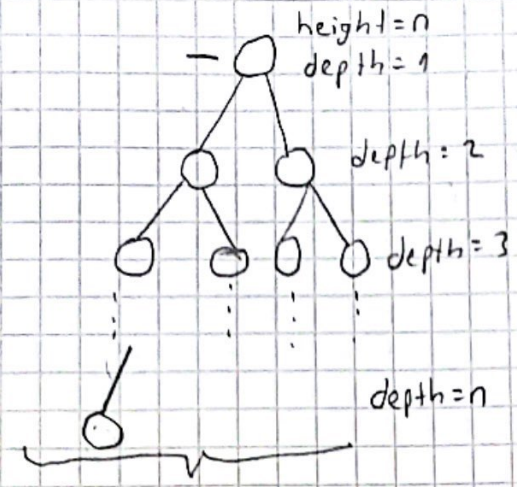
I could not to part 3 and 4.

5. RUNNING AND RESULTS

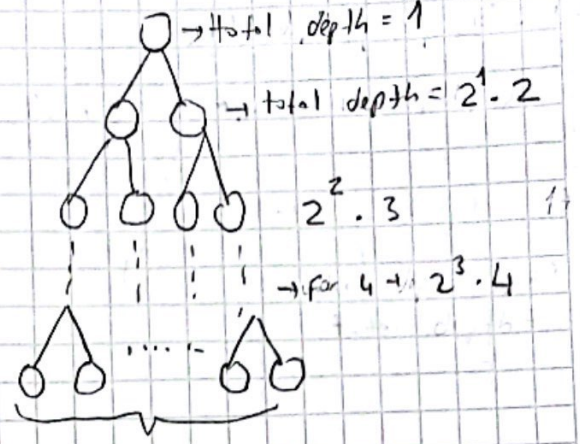
I could not to part 3 and 4.

HOMEWORK - 5

1) a



This is for minimum state. In minimum There can be one node on the left of the tree.



This is for maximum state. In maximum all nodes must be full.

In every step $2^{depth-1} \cdot depth$ is equal to total depth in that depth. for example In depth $1 = 2^0 \cdot 1 = 1$, I depth 3, $2^2 \cdot 3 = 12$ so total depth is equal sum of this equation.

$2^0 \cdot 1 + 2^1 \cdot 2 + 2^2 \cdot 3 + 2^3 \cdot 4 + \dots 2^{n-1} \cdot n$ = This is for maximum

$$= \sum_{1}^{n} 2^{n-1} \cdot n$$

n is depth.

for minimum in nth depth there must be only one node so

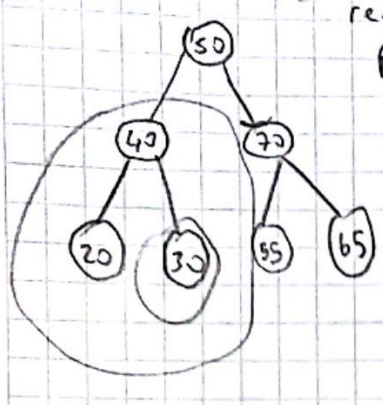
$2^1 \cdot 1 + 2^1 \cdot 2 + 2^2 \cdot 3 + \dots 2^{n-2} \cdot n + n \cdot 1$

$$= \sum_{1}^{n-1} 2^{n-2} \cdot n + n$$

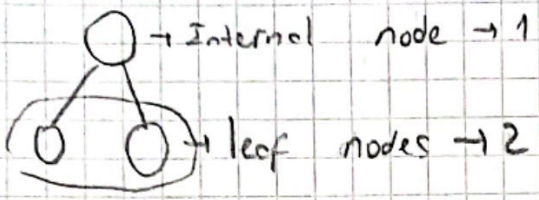
so total dept must be

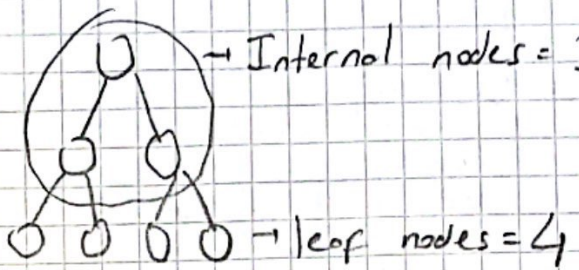
$$\sum_{1}^{n-1} 2^{n-2} \cdot n + n \leq \text{total depth} \leq \sum_{1}^{n} 2^{n-1} \cdot n$$

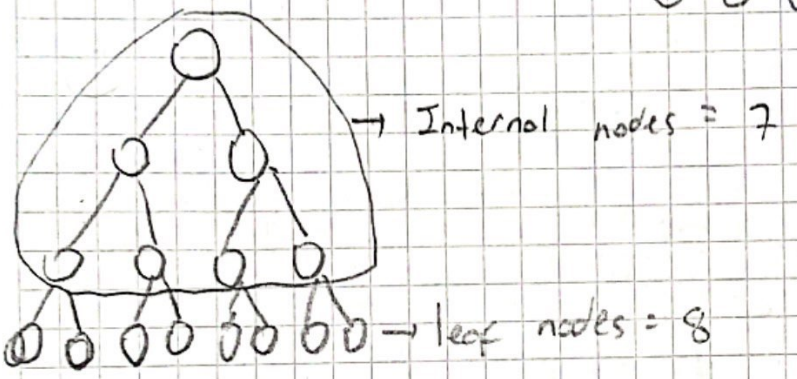
b) In a complete binary search tree before comparisons remove half of tree. for example If I look for 30 I go left and I don't have to look for half of tree. for that comparisons decrease half for every step so; $O(\log_2 n)$ overage numbers of comparisons for n elements



c)







So every time if full binary tree has n internal nodes, it has $n+1$ leaf nodes.

So if total nodes are n there must $x+1$ leaf, and x internal node so total node $= 2x+1 = n$

$2x+1 = n$

$x = \frac{n-1}{2}$ → internal node

$x+1 = \frac{n-1}{2} + 1$ → leaf nodes.

If height is h on every heigh there must at most 2^{h-1} node if we accept root's heigh 1. So

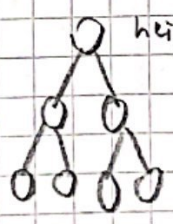
$2^0 \to 1$

$2^1 \to 2$

$2^2 \to 4$

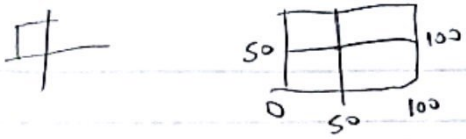
$\vdots$

$2^h \to 2^h$ node

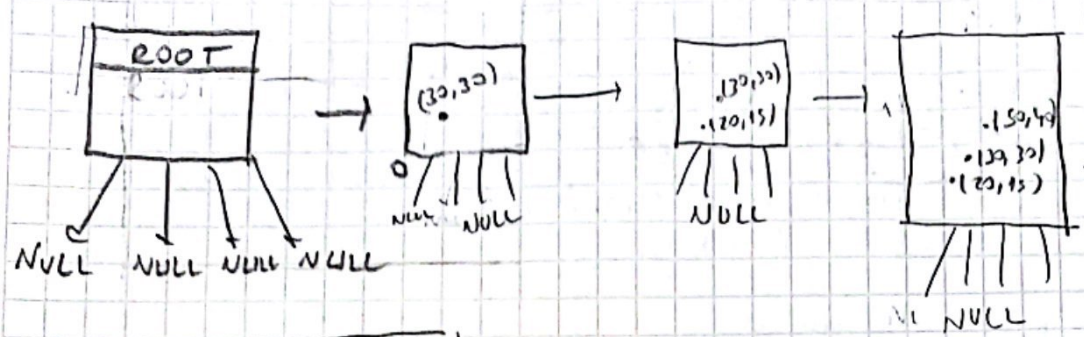


$+$

$1+2+2^1+\cdots 2^h = 2^{(n+1)} - 1$ → total number of nodes in full binary tree.

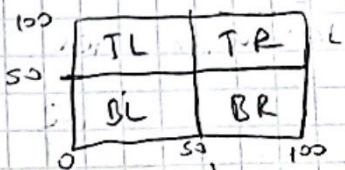


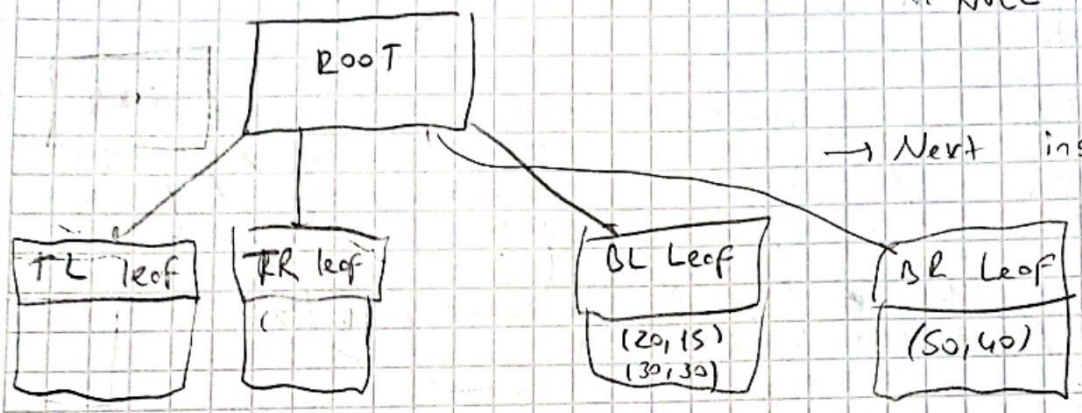
2) Let's say a node split when it has more than 2 elements



TL → Top Left
TR → Top Right
BL → Bottom left
BR → Bottom Right

→ We have more than 2 elements it will split.





→ Next insert (10, 12)

TL leaf TR leaf

BL Leaf
(20,15)
(30,30)
(10,12)
→ It has more than two elements it splits

BR Leaf
(50,40)

TL leaf TR leaf

BL branch

TL TR BL BR

(30,30) (10,12) (20,15)

BR leaf → insert (40,20)
(50,40)

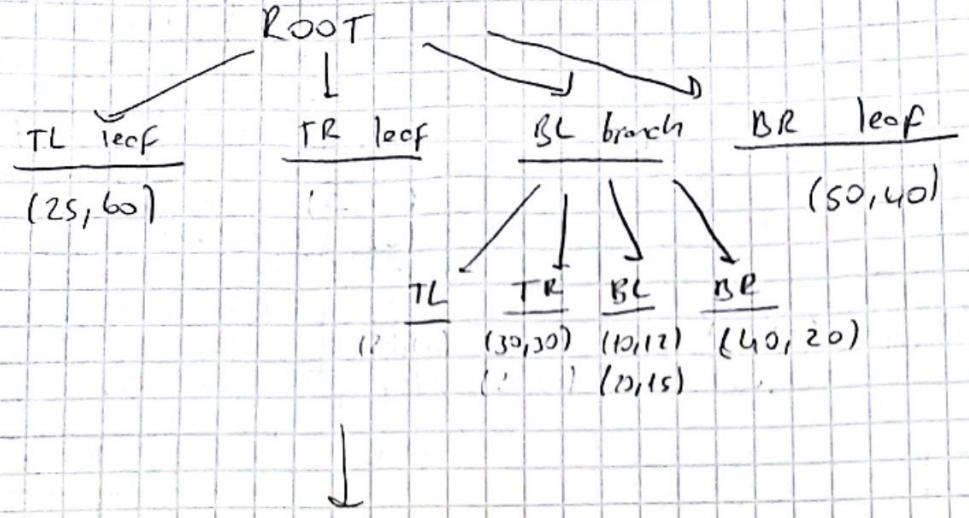
TL leaf TR leaf

BL branch

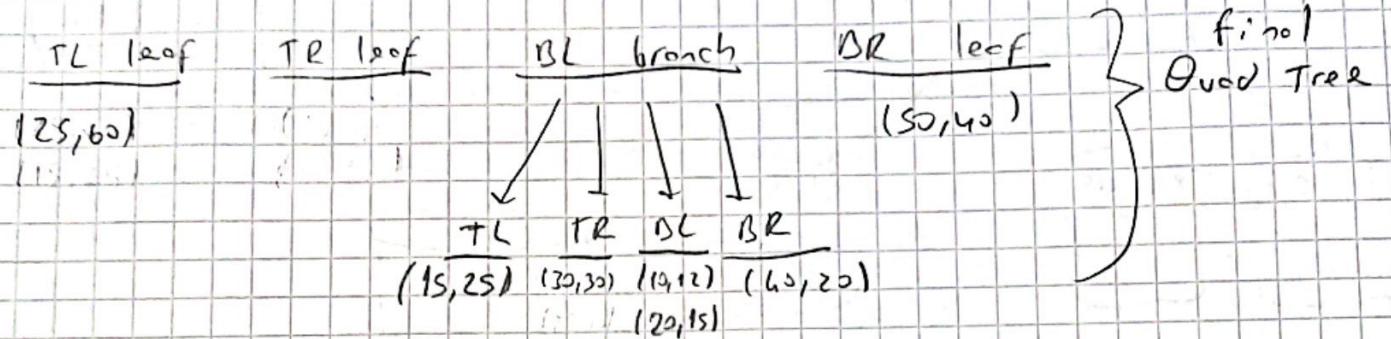
TL TR BL BR

(30,30) (10,12) (20,15) (40,20)

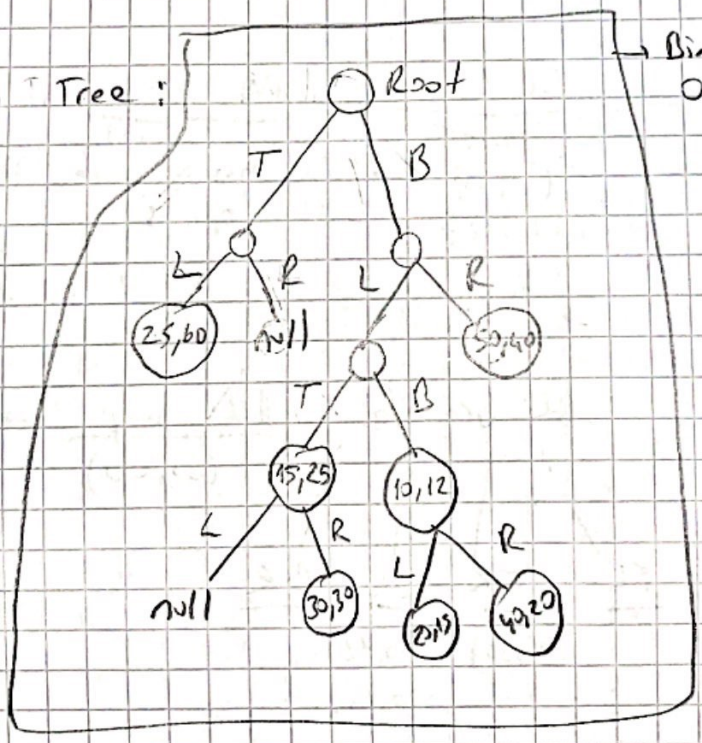
BR leaf → insert (25, 60)
(50,40)



→ insert (15,25)



Convert Binary Tree :



→ Binary tree representation of quad tree.