CS 2341

Chapter 4

Trees

Tree

path
len 2

root node — edges

parent

height = 1

child

sib tree

depth = 2

leaf nodes

This is a 3-ary tree

# of edges = N - 1
                    ↑
              # of nodes

Binary Tree (2-ary tree)

root

key   left   right

left   right

null ptr   null ptr

$$\text{avg. depth} \approx \sqrt{N}$$

# of nodes

(for random trees)

Bad Binary Tree

root

(99)

left ·
˙ ↙
nullptr

right

(5)

(100)

(1)

(8)

(9) → nullptr

d
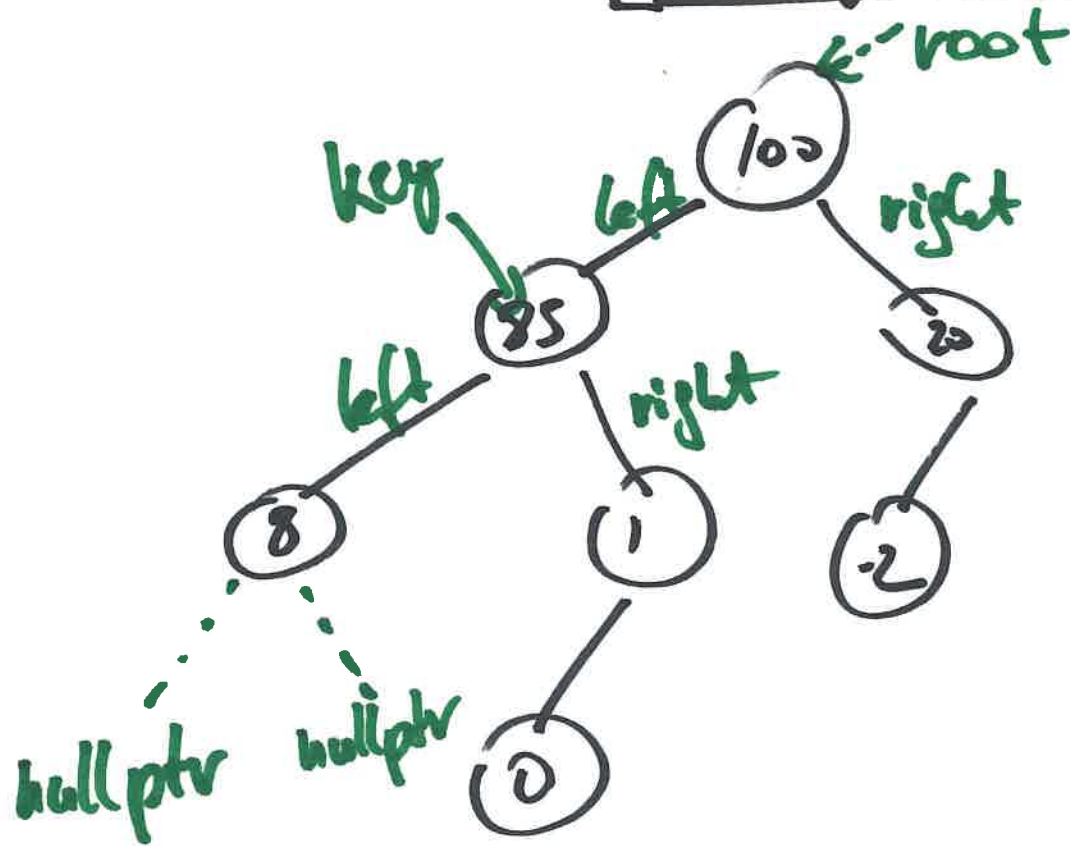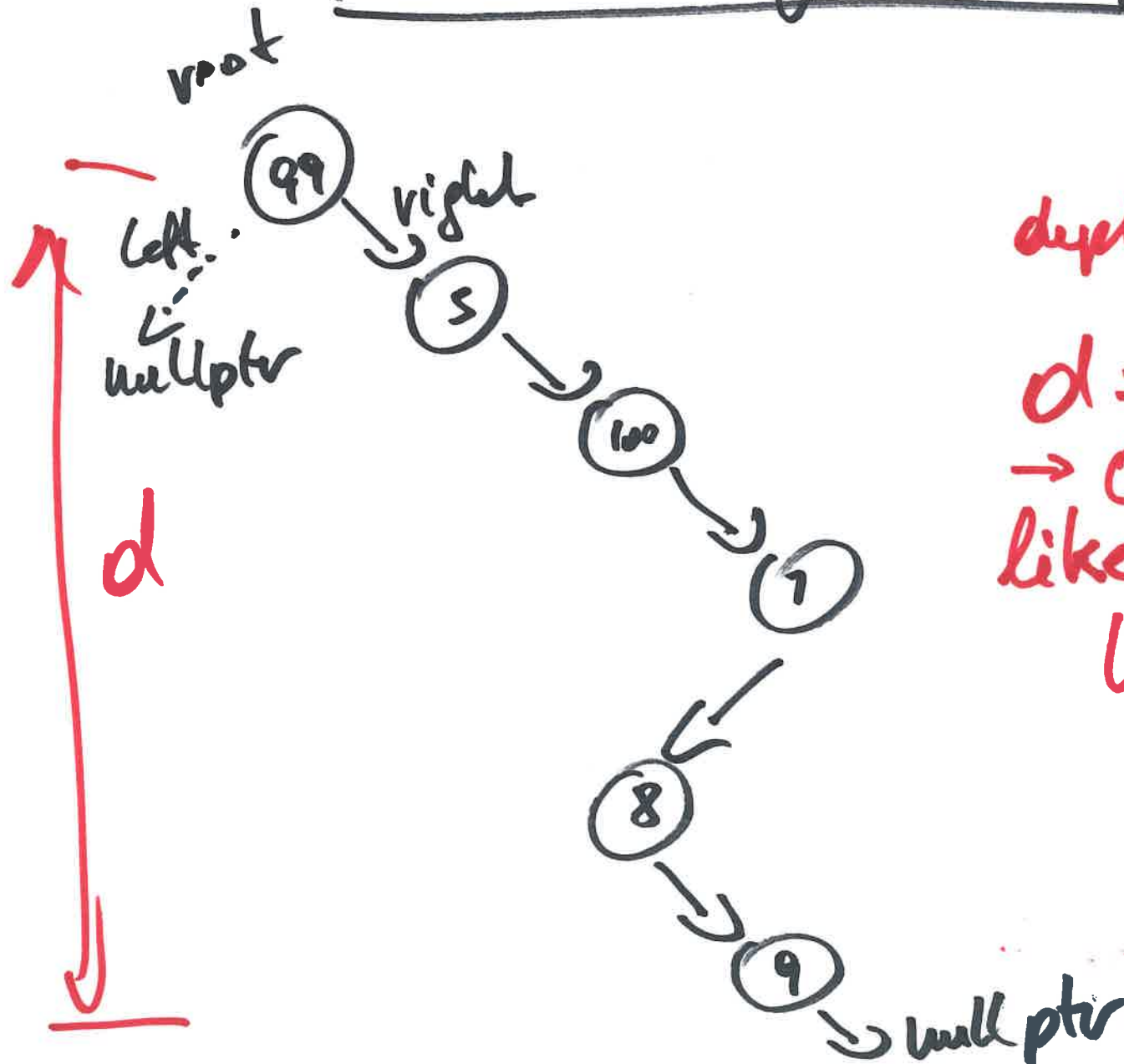
depth:

$d = N-1$!
→ $O(N)$ ·
like a
linked
list !

Tree Traversal

Depth-first traversal

NLR $\rightarrow$ log(N)

L   R

NLR

NLR

NLR

NLR

Breadth-first-traversal

level 1

2

3

4
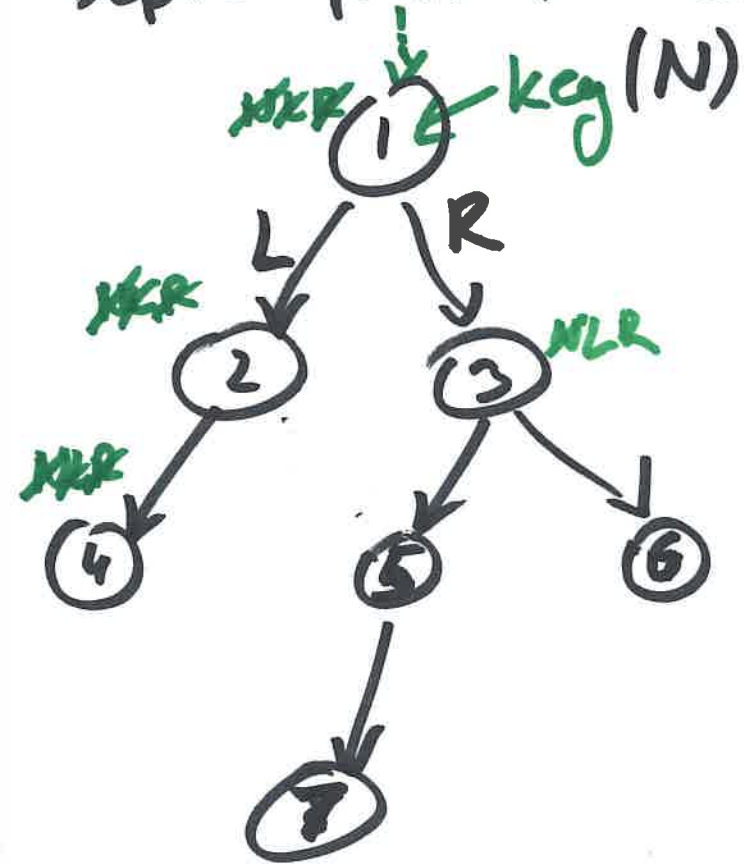
1, 2, 3, 4, 5, 6, 7
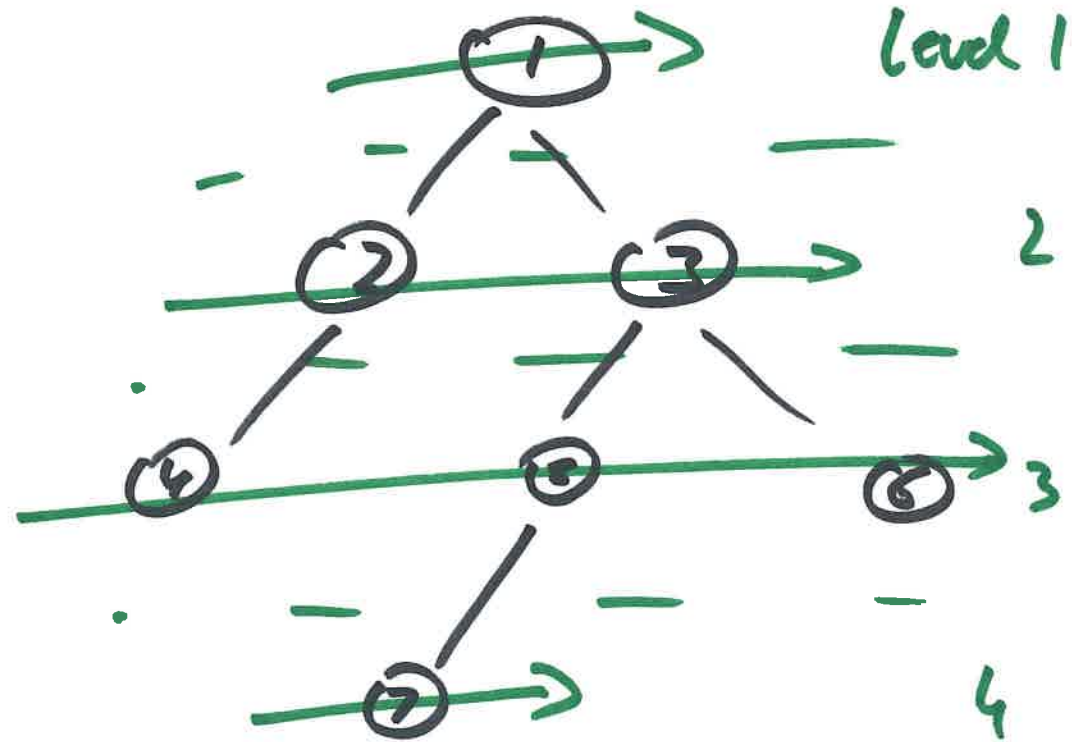
NLR (pre-order t.): 1, 2, 4, 3, 5, 7, 6
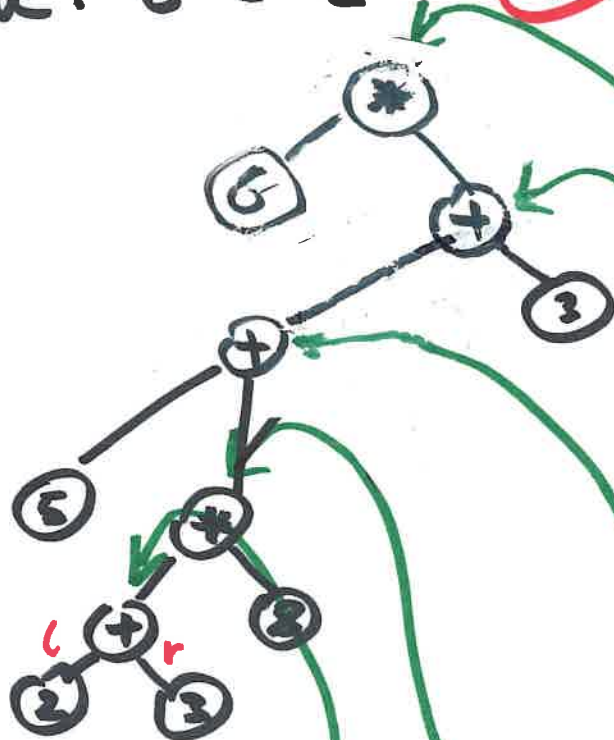LNR (in-order t.): 4, 2, 1, 7, 5, 3, 6
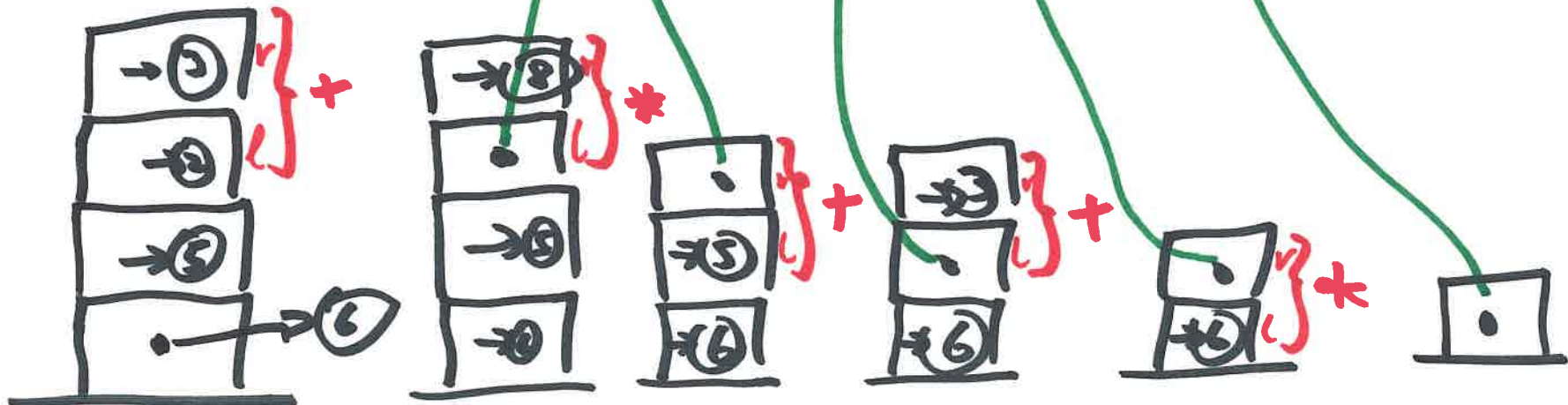LRN (post-order t.) 4, 2, 7, 5, 6, 3, 1

Expression tree from postfix notation

Example: 6 5 2 3 (+) 8 (*) (+) 3 (+) (*)

- numbers → stack
- operator: pop
  2 numbers
- make a subtree
- push pointer to
  subtree onstack

$\boxed{\text{In - order traversal}}$ of the Expression tree

$(L) \ N \ (R) \rightarrow$ infix notation

$(6) * ((5) + ((2+3) * 8) + 3)$

$= 6 * (5 + (2+3) * 8 + 3)$

$\boxed{\text{Post - order traversal}}$ LRN $\rightarrow$ post fix notation

$6 \ 5 \quad 2 \quad 3 + 8 * + 3 + *$

Binary Tree
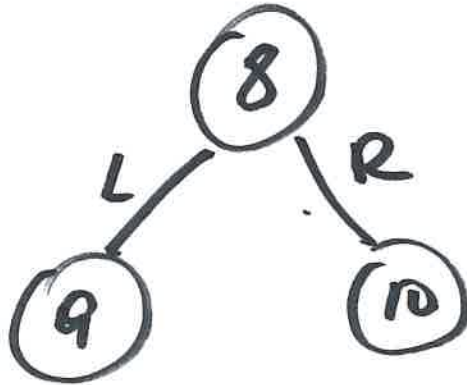
$L < N < R$

$9 \cancel{<} 8 < 10$

Binary Search Tree
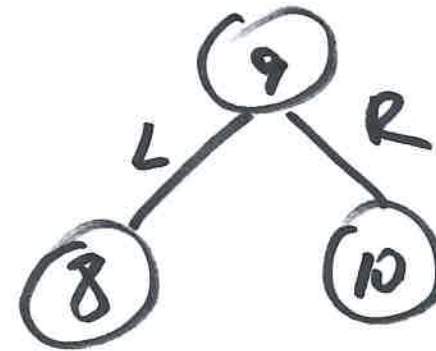
order property: $L < N < R$

$L < N < R$

$8 < 9 < 10$ ✓

Insert into Binary Search Tree
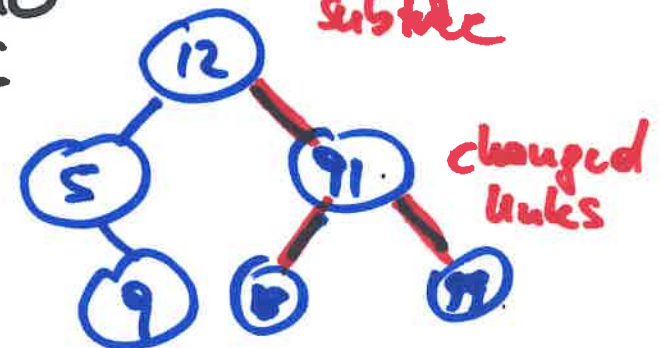## + Delete

insert: 12, 90, 3, 5, 18, 9, 99, 91

root → nullptr

Insertions =>

root
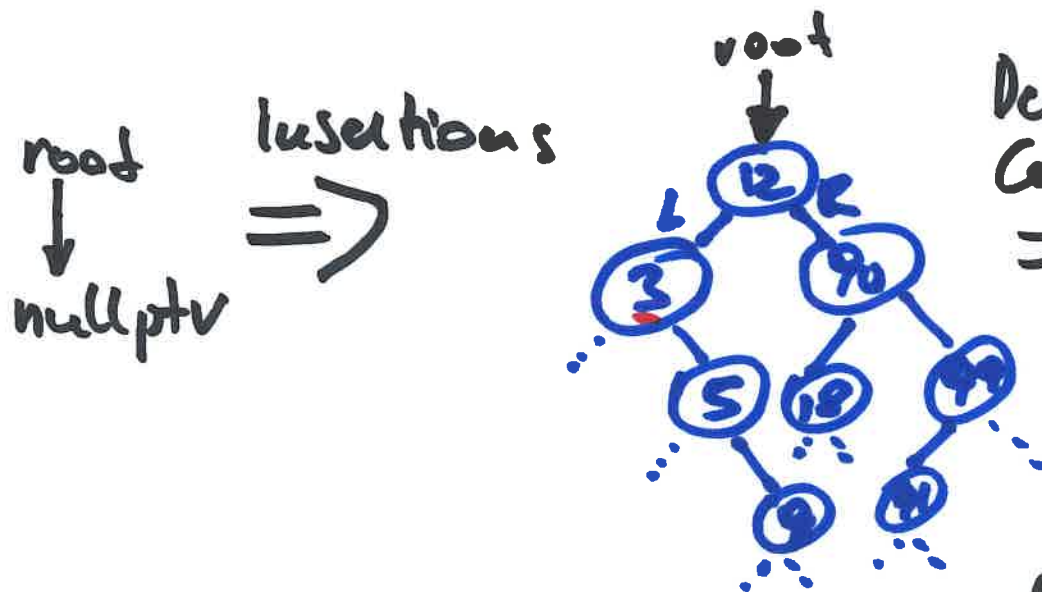
Delete 3
Case B
=>
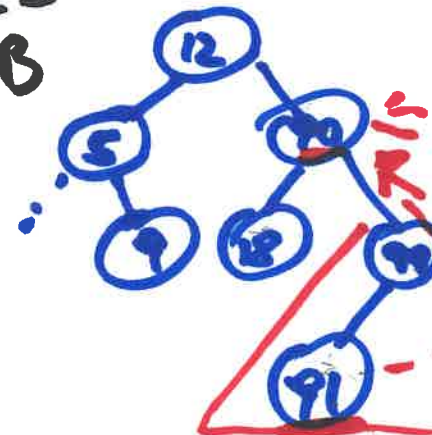
Delete 90
Case C

Delete 90 Case C

Steps
1. unlink 91
2. relink in place of 90
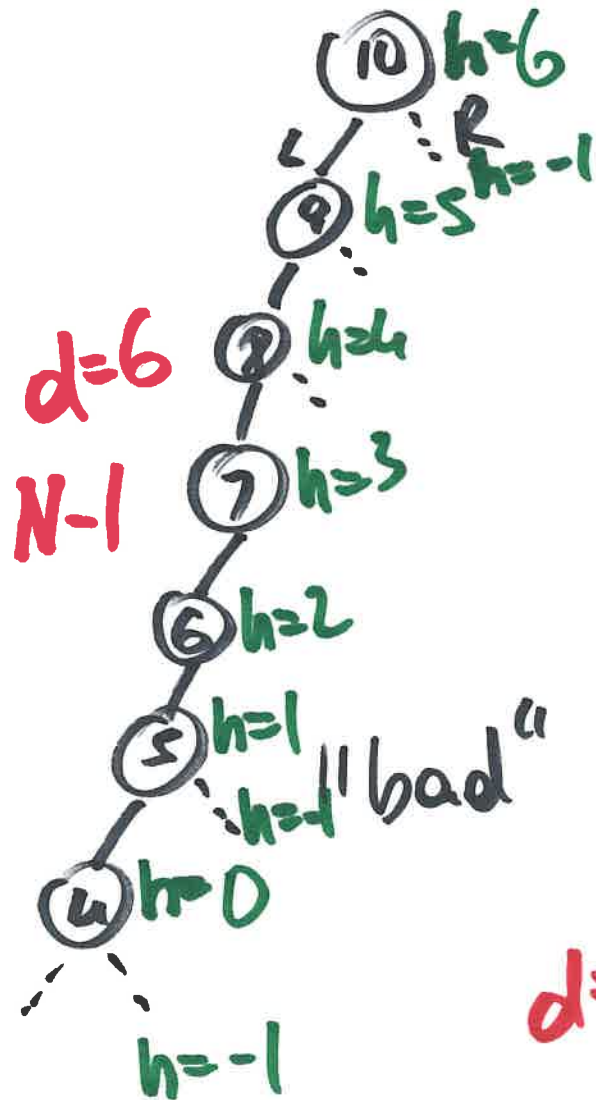3. delete node 90

Smallest in right subtree

changed links

delete : 3, 90

inorder traversal : 5, 9, 12, 18, 91, 99
(LNR)

B. Search Trees & Balance

insert: 10, 9, 8, 7, 6, 5, 4 $N = 7$

10 $h=6$

L R $h=-1$

9 $h=5$

8 $h=4$ $d=6$ $N-1$

7 $h=3$

6 $h=2$

5 $h=1$

$h=-1$ "bad"

4 $h=0$

$h=-1$

2

$\Rightarrow$
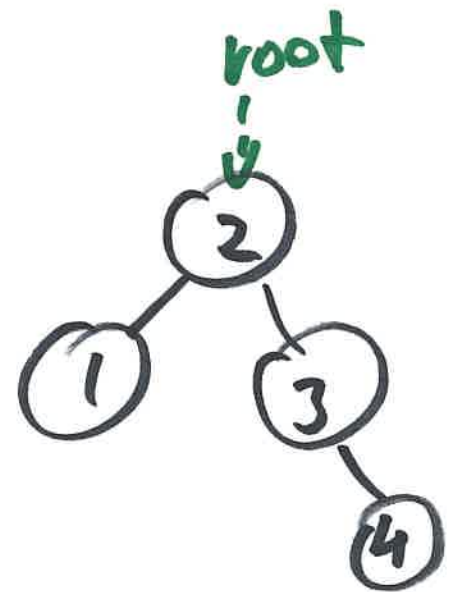
$d=2$

7 $h=2$

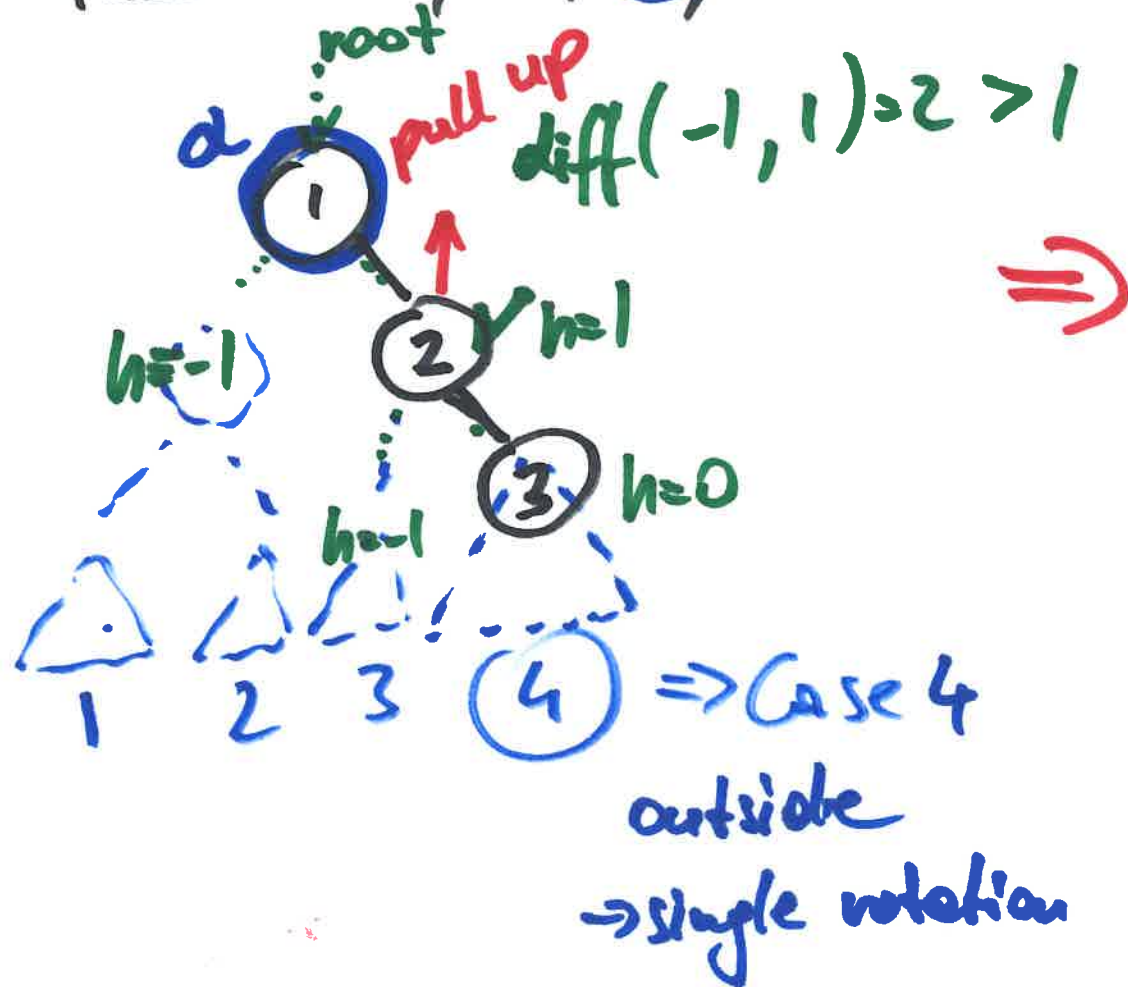5 $h=1$ 9 $h=1$

4 $h=0$ 6 $h=0$ 8 $h=0$ 10 $h=0$
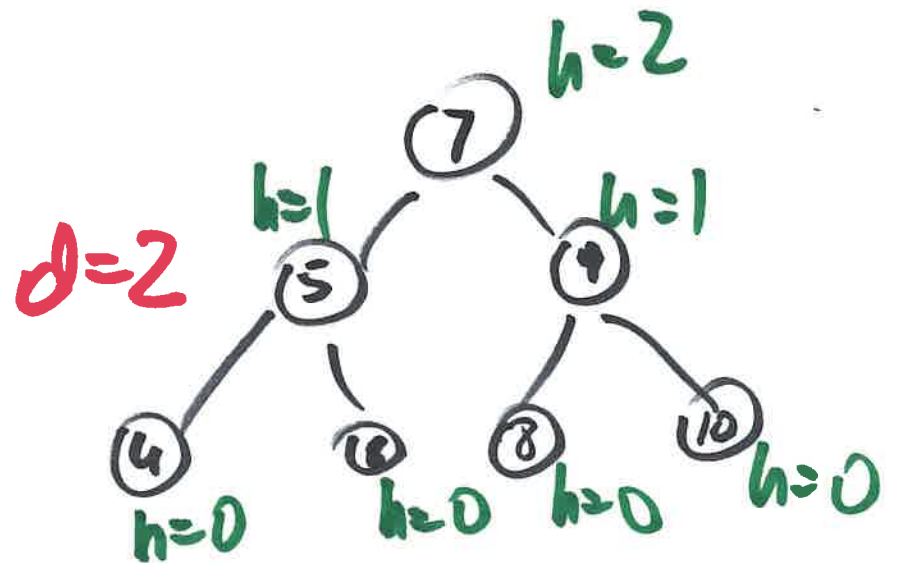
"good"

$$d = \log_2(N+1) - 1 = 3 - 1 = 2$$

AVL Tree

insert: 1, 2, ③, 4

a ① root pull up diff$(-1, 1) = 2 > 1$

② h=1

h=-1

③ h=0

h=-1

1 2 3 ④ ⟹ Case 4

outside

⟶ single rotation

⟹

root

②

①   ③

④

AVL Tree

insert: 5,1,6,3,②

h=3
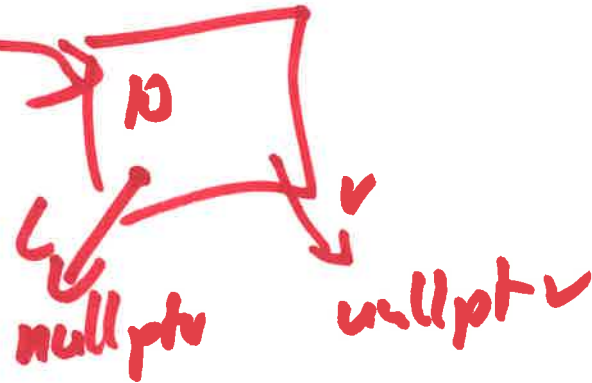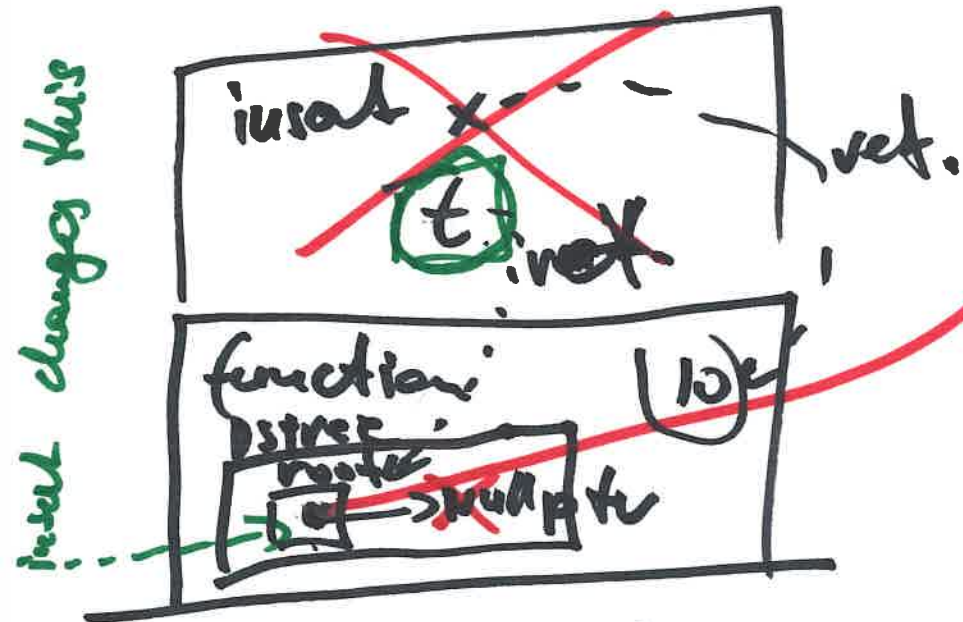
h=0

α

h=2

h=1 pull up

h=1

h=0

1 2 ③ 4
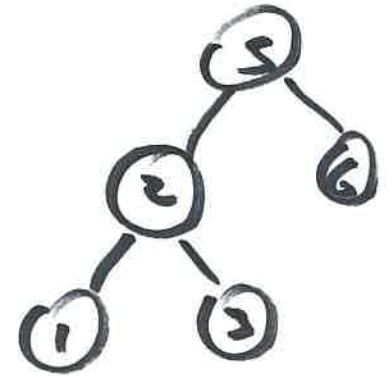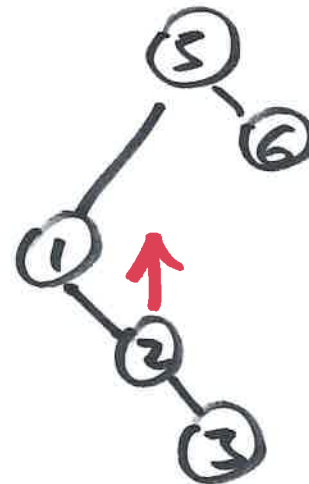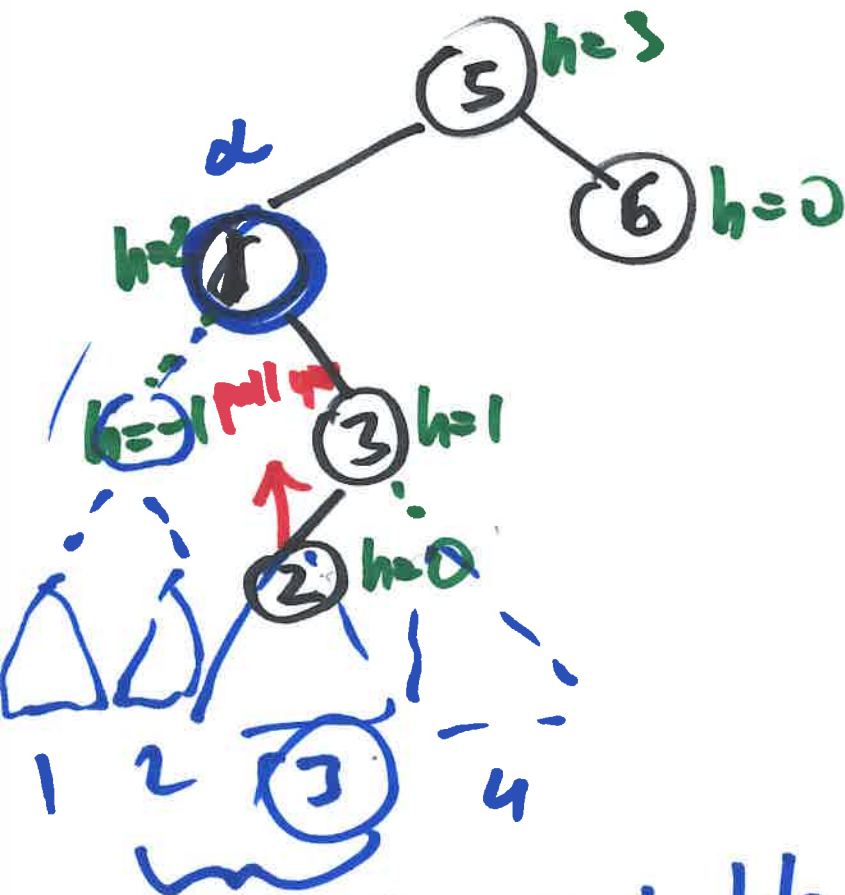
inside → double rotation

1. find insertion point
2. do insertion
3. balance

Why to pass a reference to a pointer?

→ to change the pointer in the calling function

insert ~~x~~ → ret.

~~(t)~~ ~~.ret~~

function:
insert
root → ~~nullptr~~

10

↙ nullptr   ↘ nullptr

ireal changes this

void insert (const int & x, Binary Node * & t)

↑ function just wants to "look at" x

no const!!!
= function wants to change t !!!
Binary Node * & t

ref.

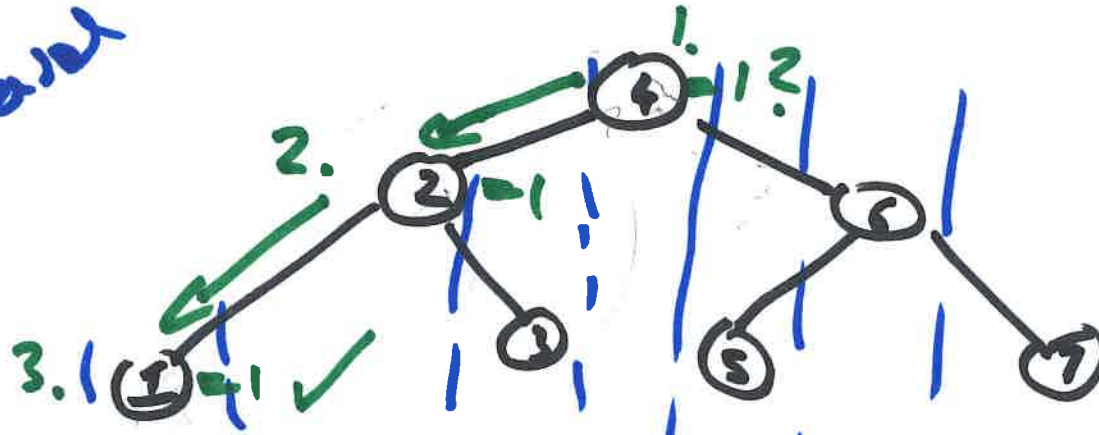insert ((10), root)

insert (100, root)

Time Complexity of an AVL Tree

insert: $O(find) + O(add\ to\ tree) + O(balance)$

$\qquad\quad O(\log N) + O(1) \qquad\quad + O(1)$

$\qquad\quad O(d) \implies \boxed{O(\log N + 2)}$

delete: $O(\log N)$
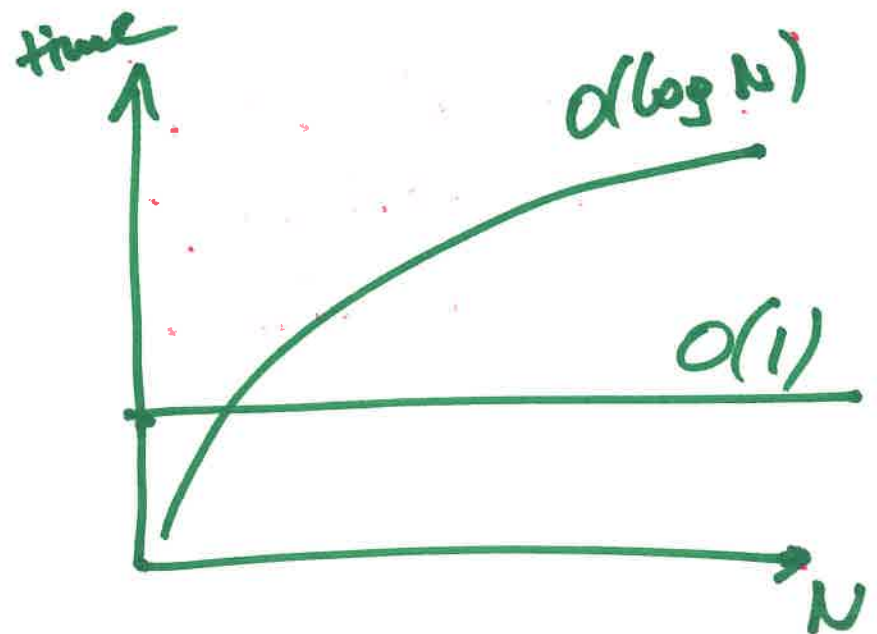
Binary Tree Search : find 1

in order traversal LNR

1.
2.
3.1

is equivalent

Binary Search : find 1

1	2	3	4	5	6	7
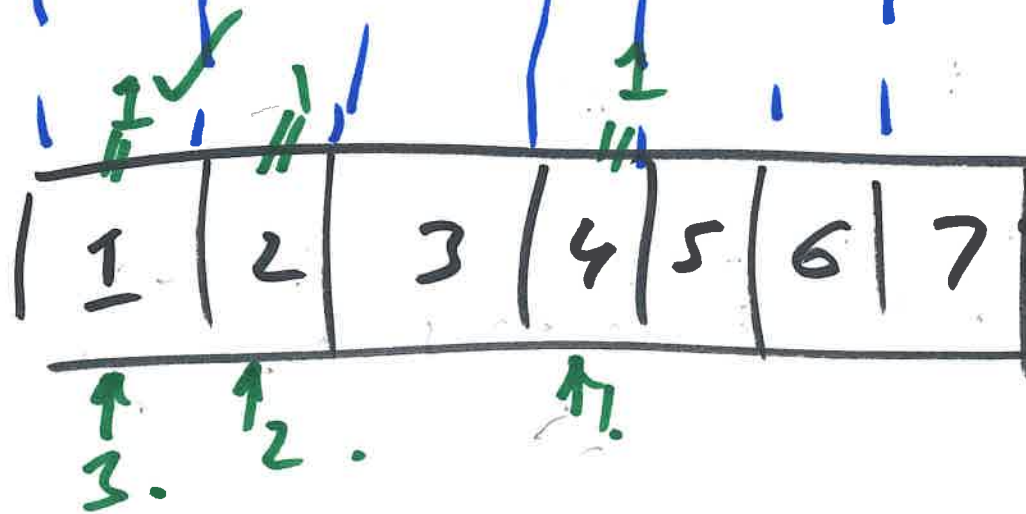
representation of a binary tree as an array

MAP

Set X (snUID)   Set Y (Name)

$y = f(x)$

3348573 ———————————→ "Michael"

9932457 ———————————→ "Save"

8857324 ———————————→ "Peter"

↖ keys   ↖ values
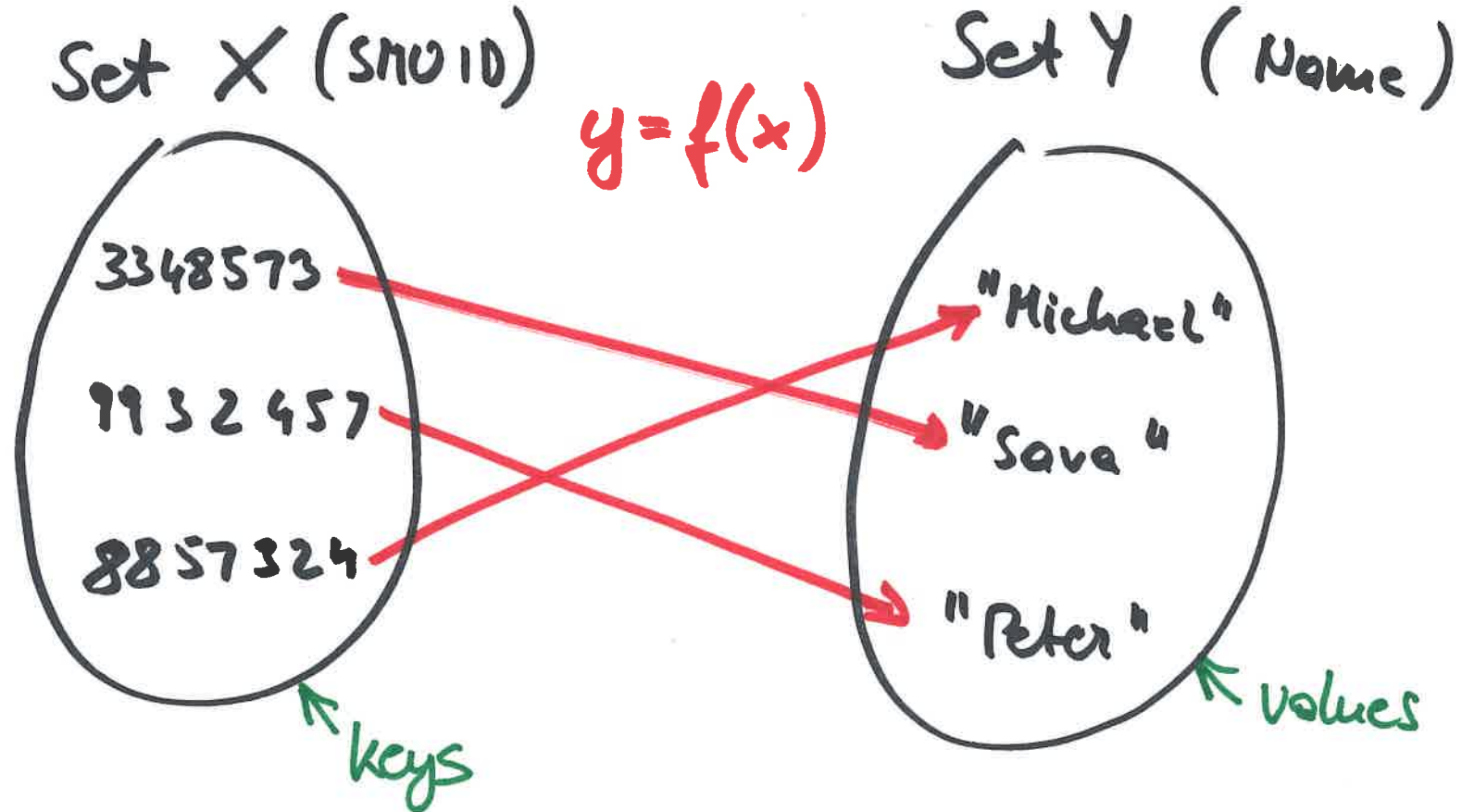
one - to - one mapping : bijection

Implementation : add a value member variable
                 to the node class