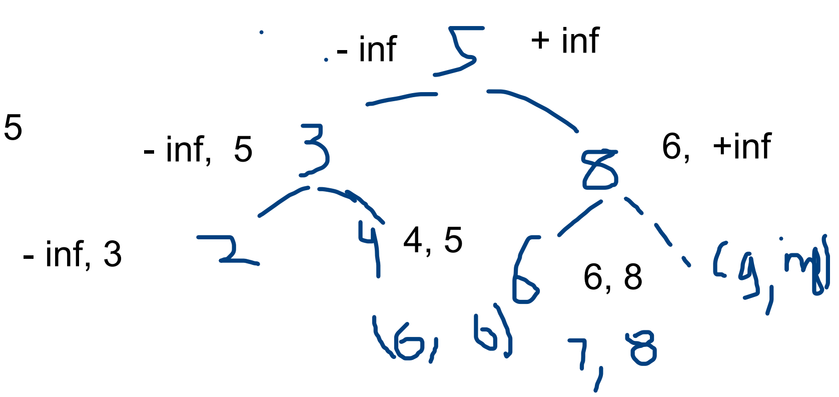
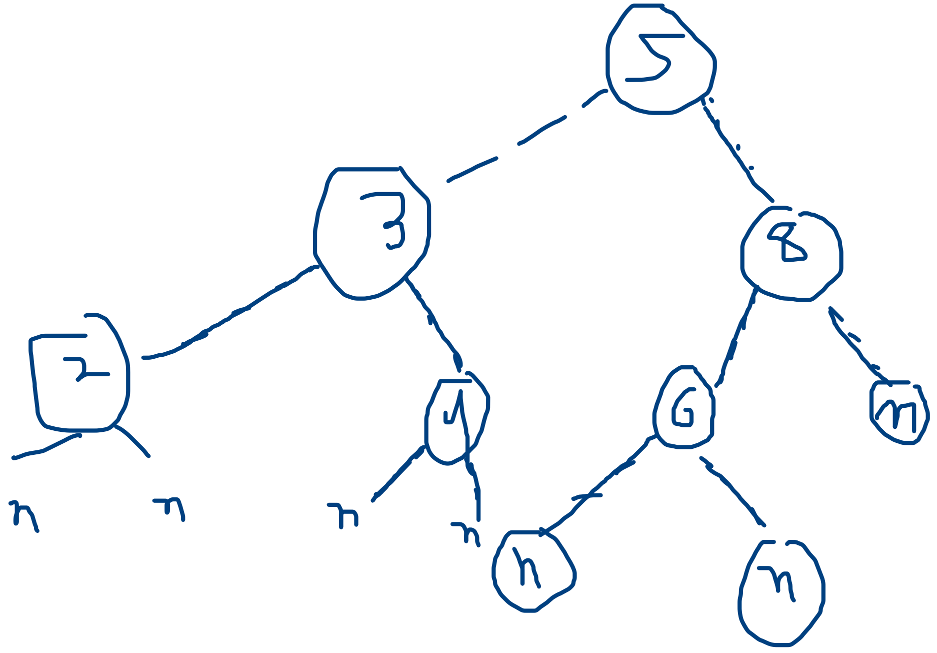
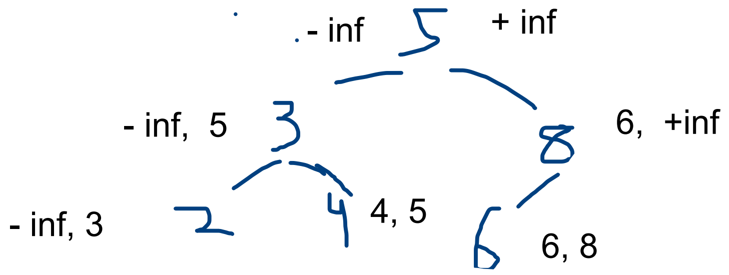
Construct BST from post order

post order: 2 4 3 6 8 5

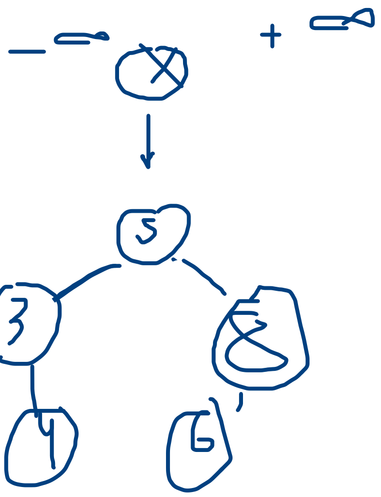




Construct BST from Level order



Level order: 5 3 8 2 4 6



queue:

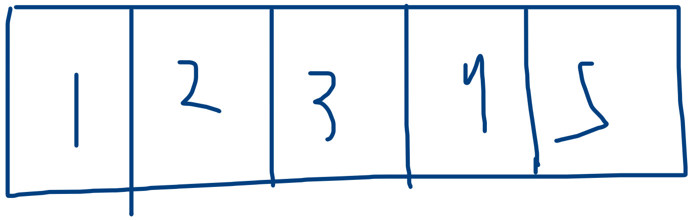
1. BinaryTreeNode parent
2. int left
3. int right

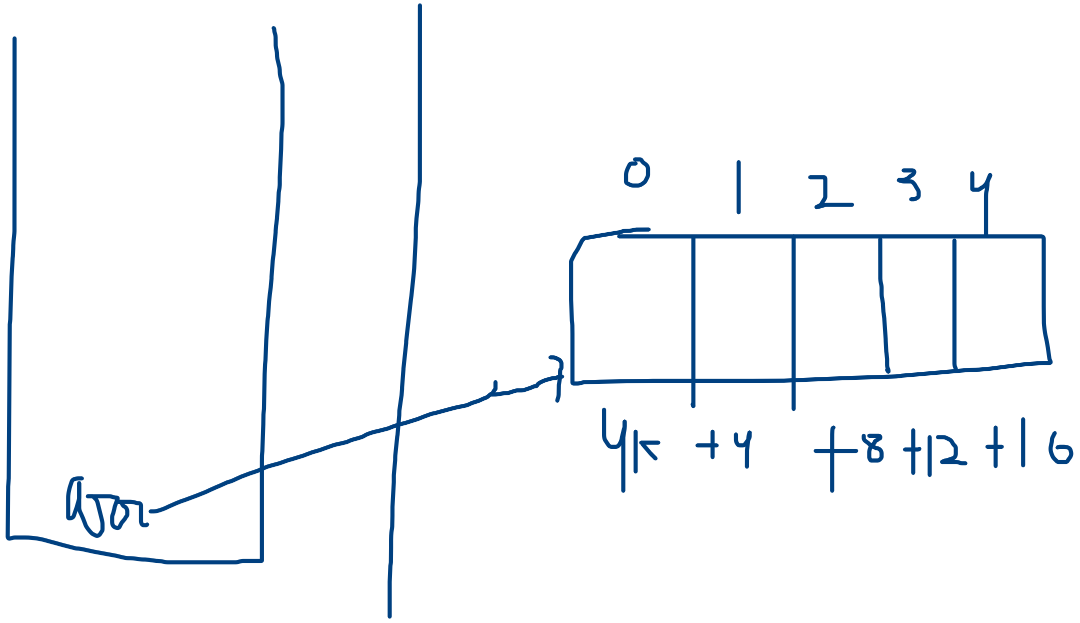
2, - inf, 2
2, 2, 3

8, 8, +inf

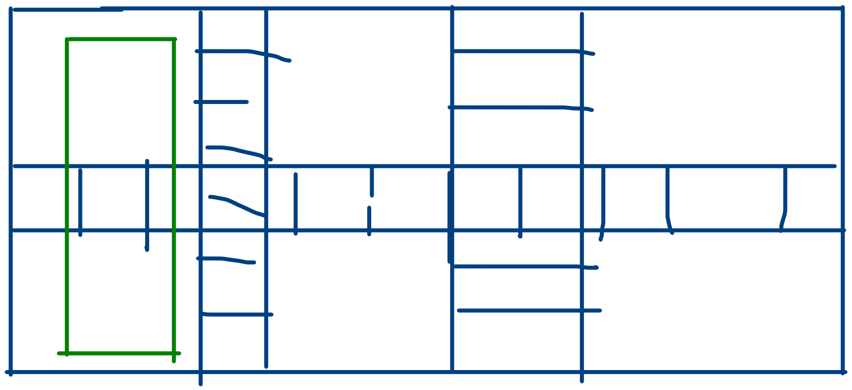
Linked List

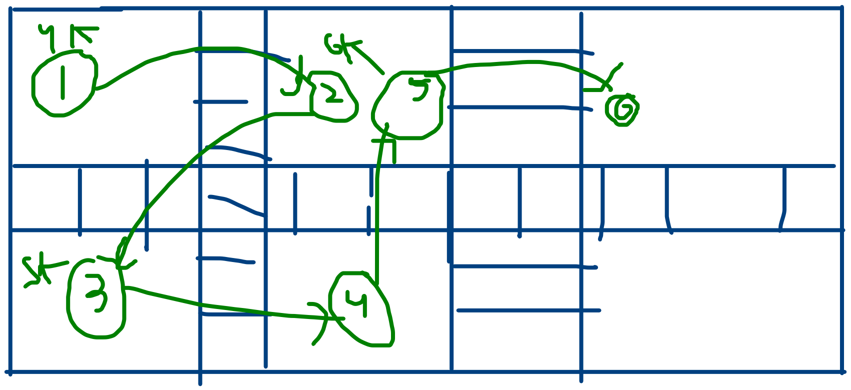
ARR





Memory is limited and occupied in different things

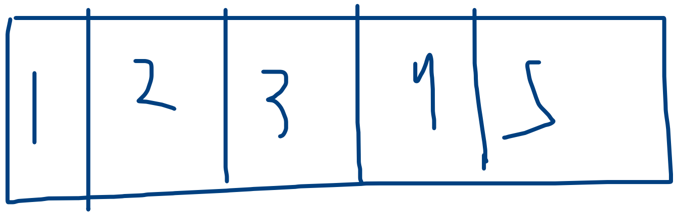




Pro

1. It is not continous in memory so it can store data in these type of cases

ARR



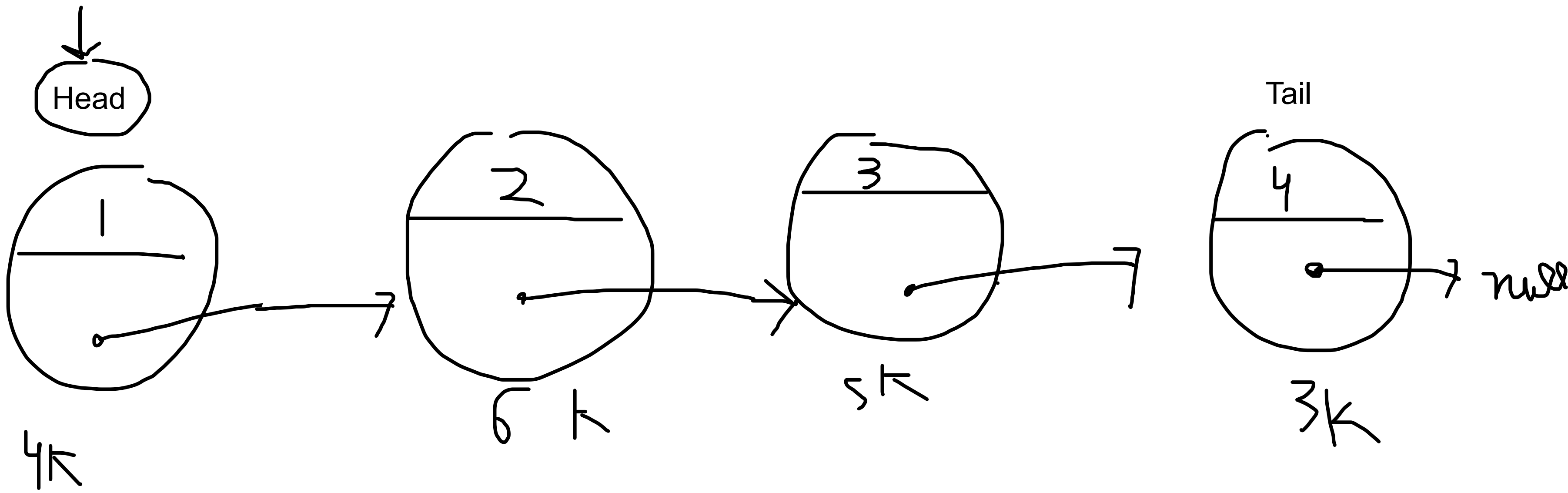
arr --> insertion, deletion, updation --> time = O(1)

LinkedList --> insertion, deletion, updation --> depends but it is not O(1)

arr is faster than linkedlist

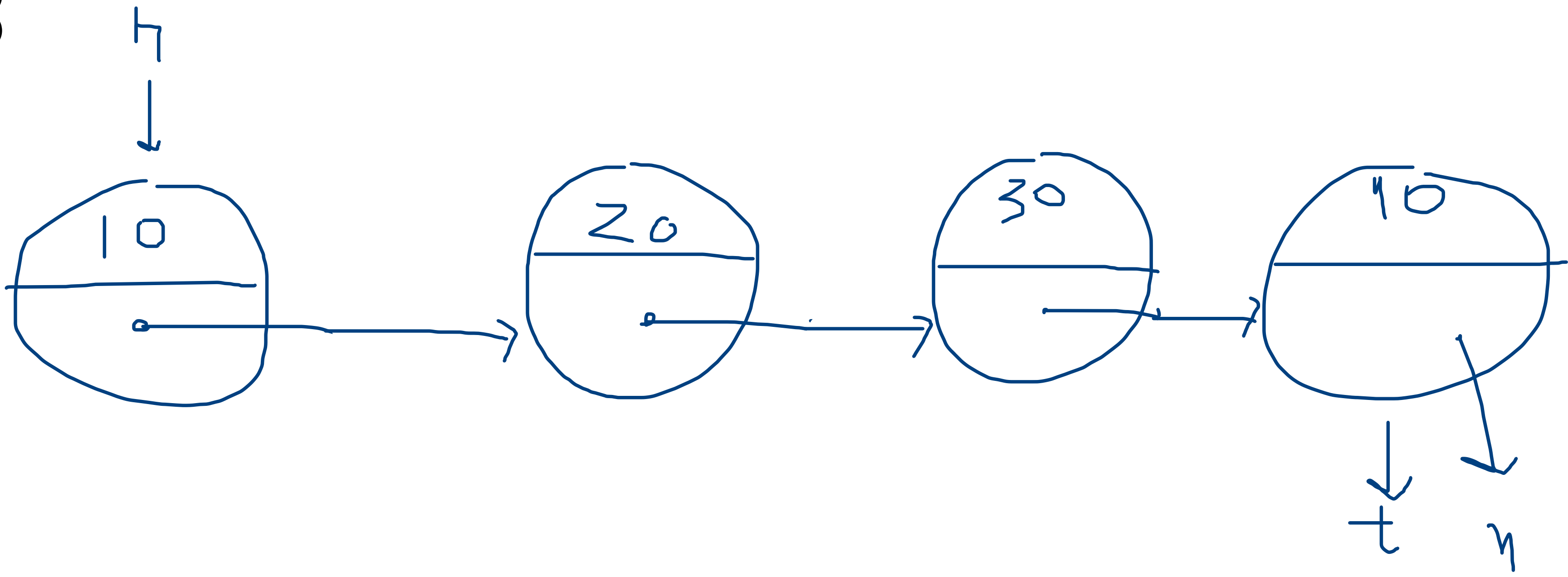
Node

1. data (any similar datatype)
2. Node next (address of your next node)



add Last in LL

addLast(10)
addLast(20)
addLast(30)
addLast(40)
quit



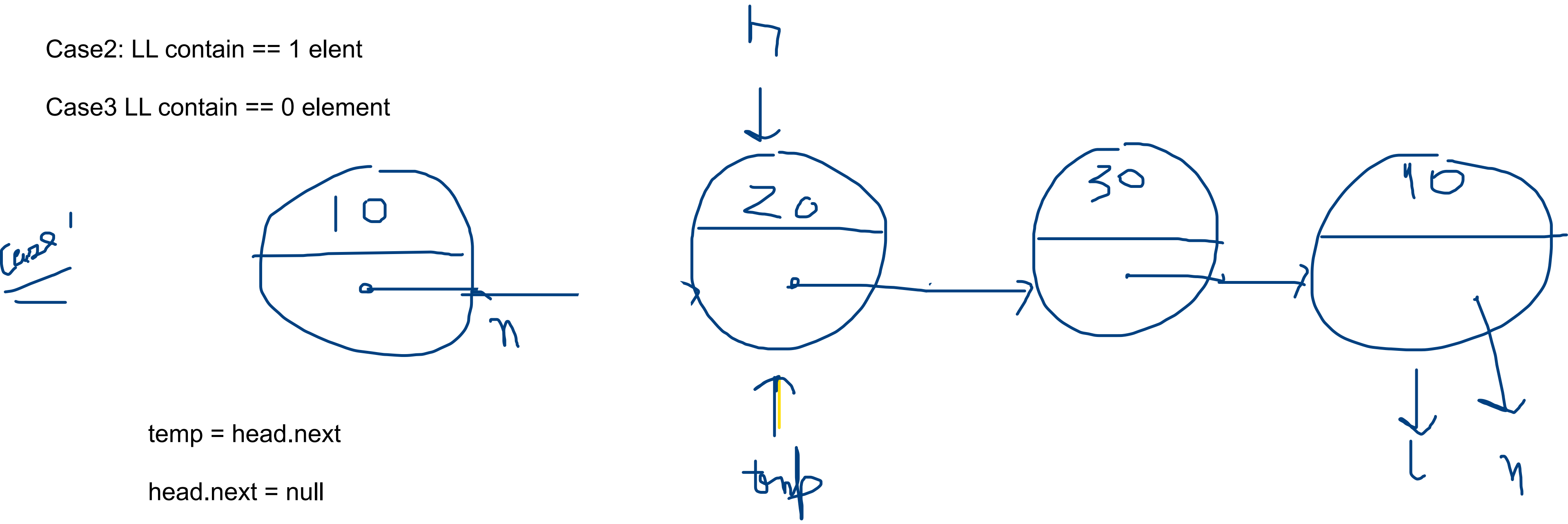
Remove First

null pointer error (NPE) -- try to avoid this

Case1 : LL contain > 1 element

Case2: LL contain == 1 elent

Case3 LL contain == 0 element



temp = head.next

head.next = null

head = temp