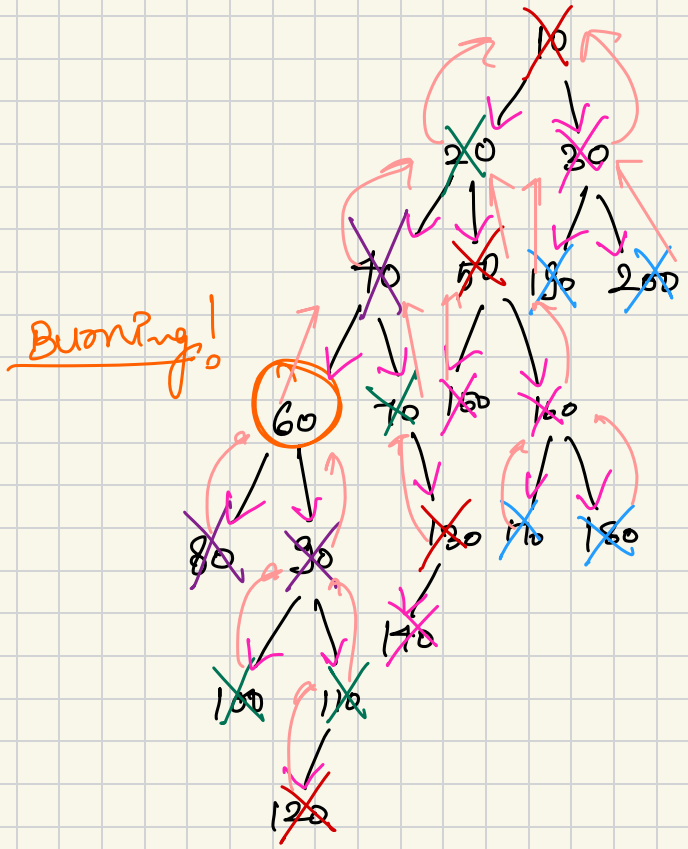
Agenda

- Burning of a Binary Tree
- Flatten a Binary Tree
- Construct a Binary Tree using PreOrder & InOrder
- Serialize and Deserialize of Binary Tree
- Write pre/post/in iteratively

Construct a BT

Burning of a Binary Tree.

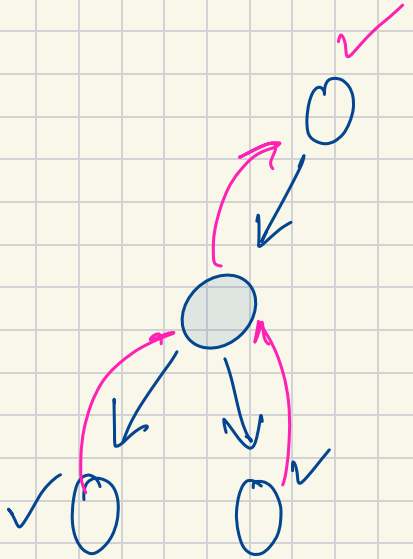


Minimum time to burn tree = 5 units of time

HashMap

Child → parent ✓

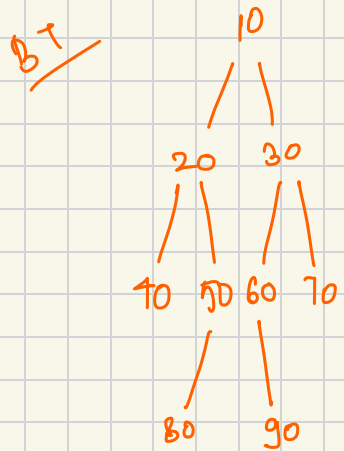
get parent
↳ O(1)



Key (child) Value (parent)

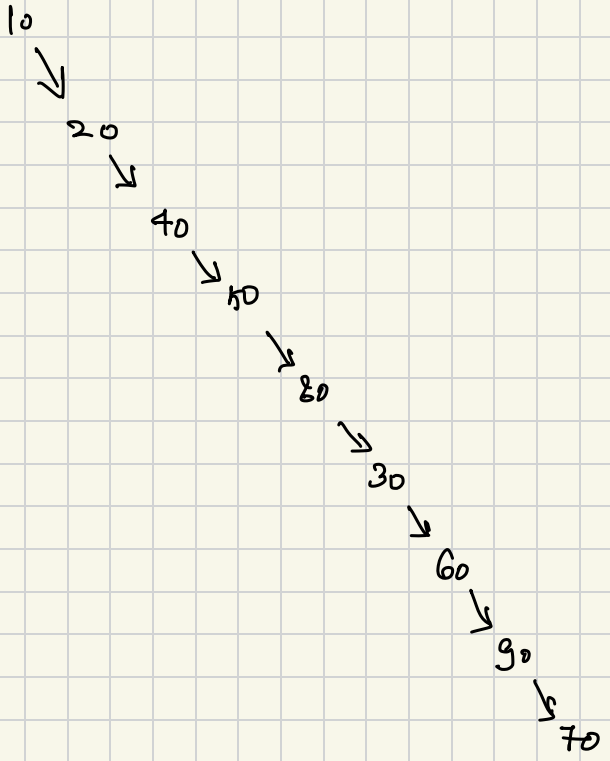
Flatten a Binary Tree

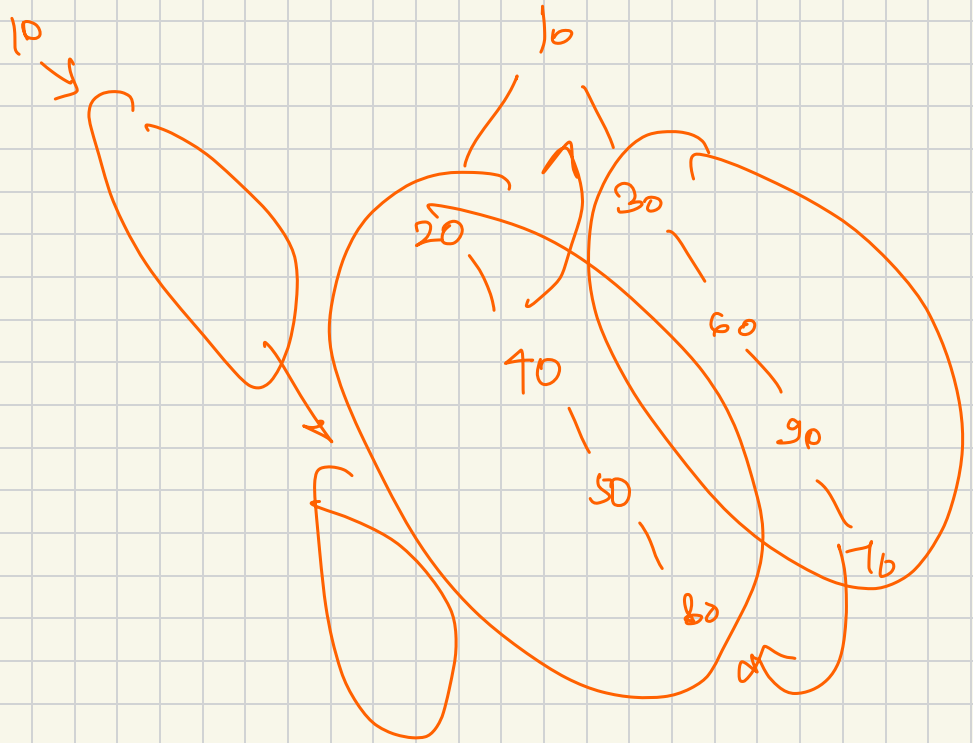
{ in place }



flatten( ) → BT

after it flattens till




10
16
30
20
60
40
90
50
76
80

Construct a BT using PreOrder and InOrder traversal

root LST RST

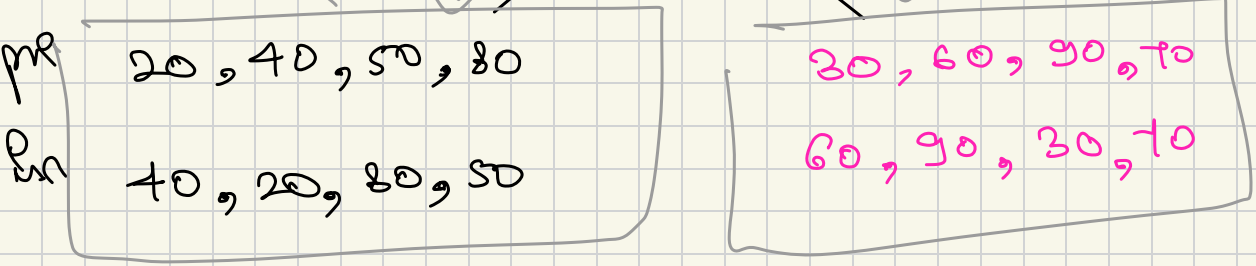
LST root RST

pre-order: 10 20 40 50 80 30 60 90 70

in-order: 40 20 80 50 10 60 90 30 70

LST

Construct ( ) returns root of the after constructing from pre and in



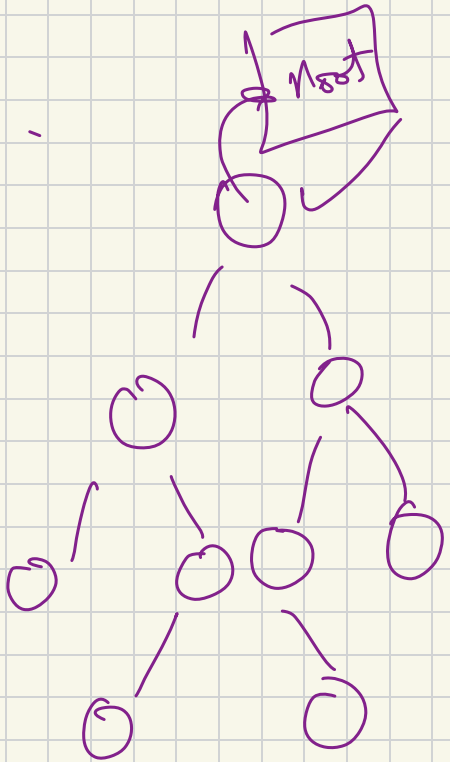
Pre

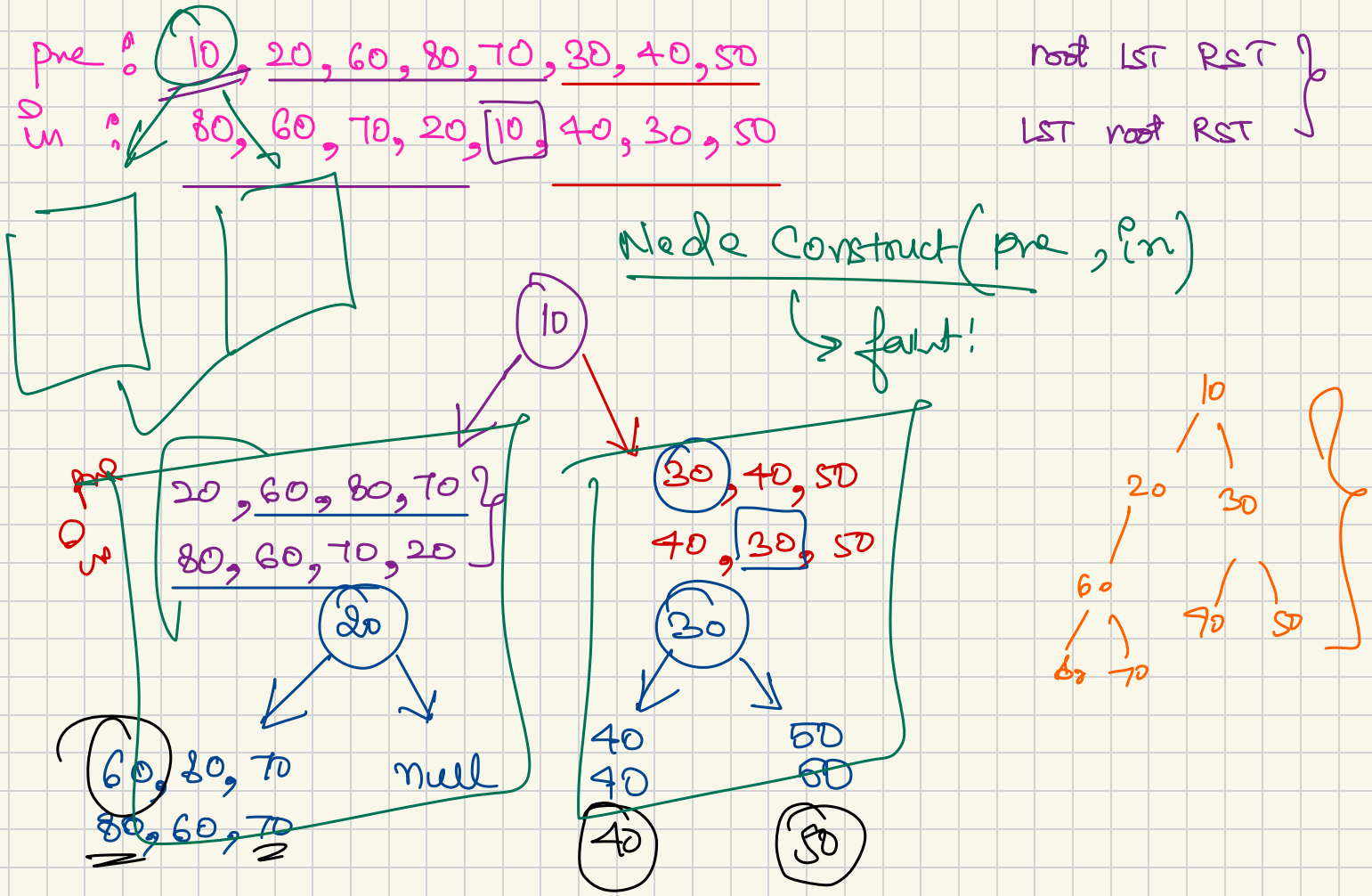
[

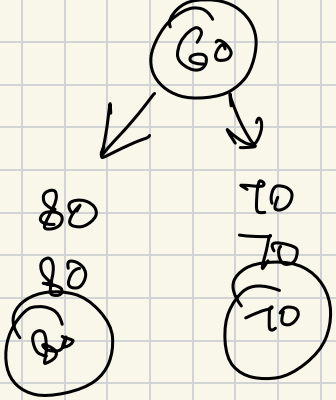
]

[

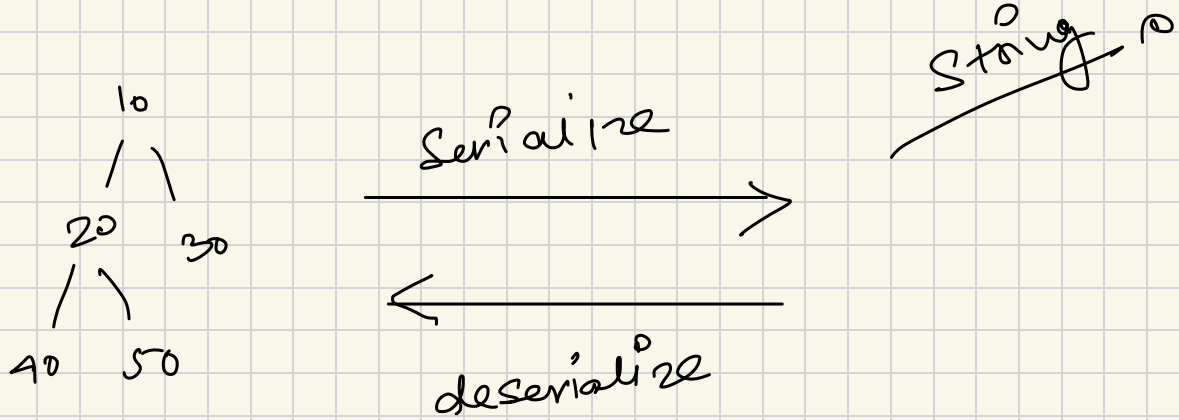
]



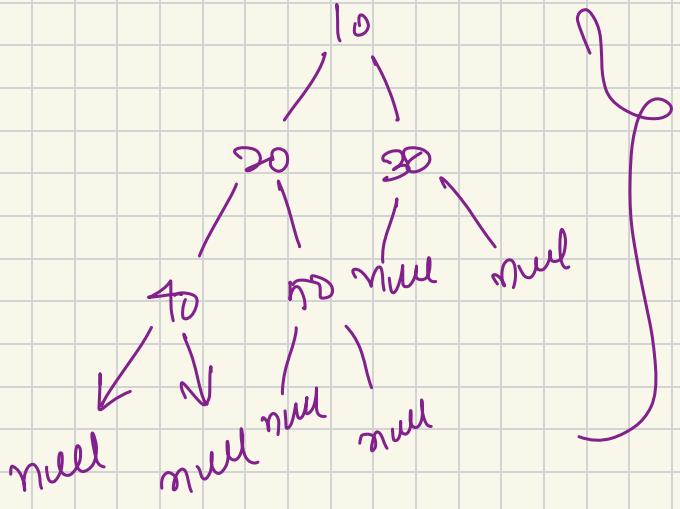

pre : 10, 20, 60, 80, 70, 30, 40, 50
in : 80, 60, 70, 20, 10, 40, 30, 50
root LST RST
LST root RST
Node Construct(pre, in)
fault!
10
20, 60, 80, 70
80, 60, 70, 20
20
60, 80, 70
80, 60, 70
null
30, 40, 50
40, 30, 50
30
40
40
50
50
40
50
10
20
30
60
70
50
60
70


60
80
80
70
70

Serialize And Deserialize.



preOrder: 10, 20, 40, n, n, 50, n, n, 30, n, n



fun()