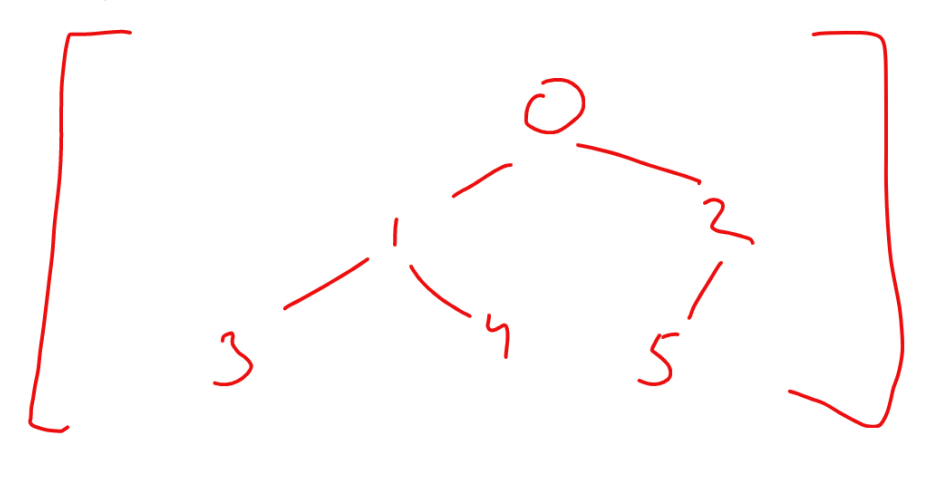
in[] = [3, 1, 4, 0, 5, 2]
pre[] = [0, 1, 3, 4, 2, 5]
I/P

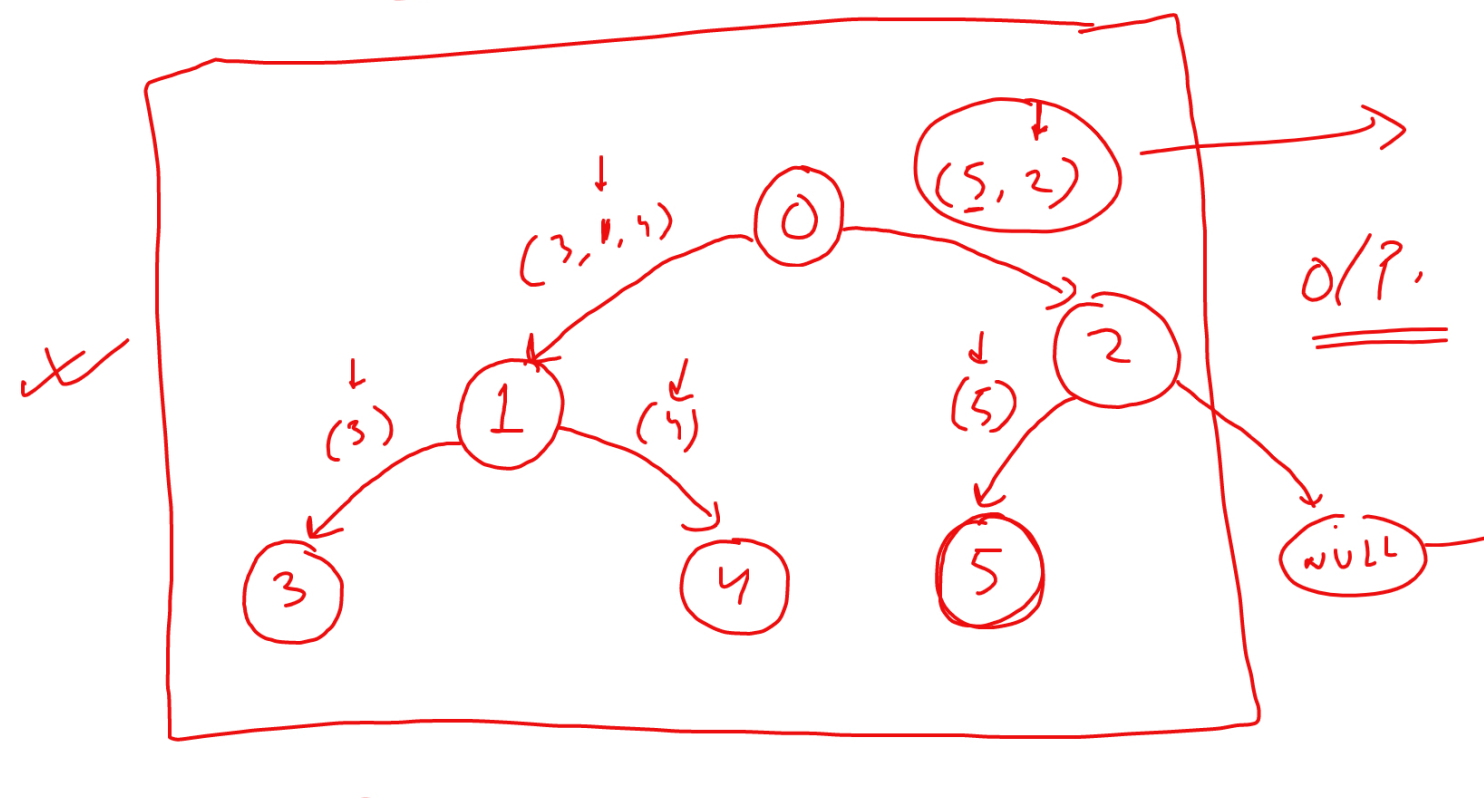


O/P

inorder: left subtree → root → right subtree
preorder: root → left subtree → right subtree

in() = [3, 1, 4, 0, 5, 2]
pre () = [0, 1, 3, 4, 2, 5]

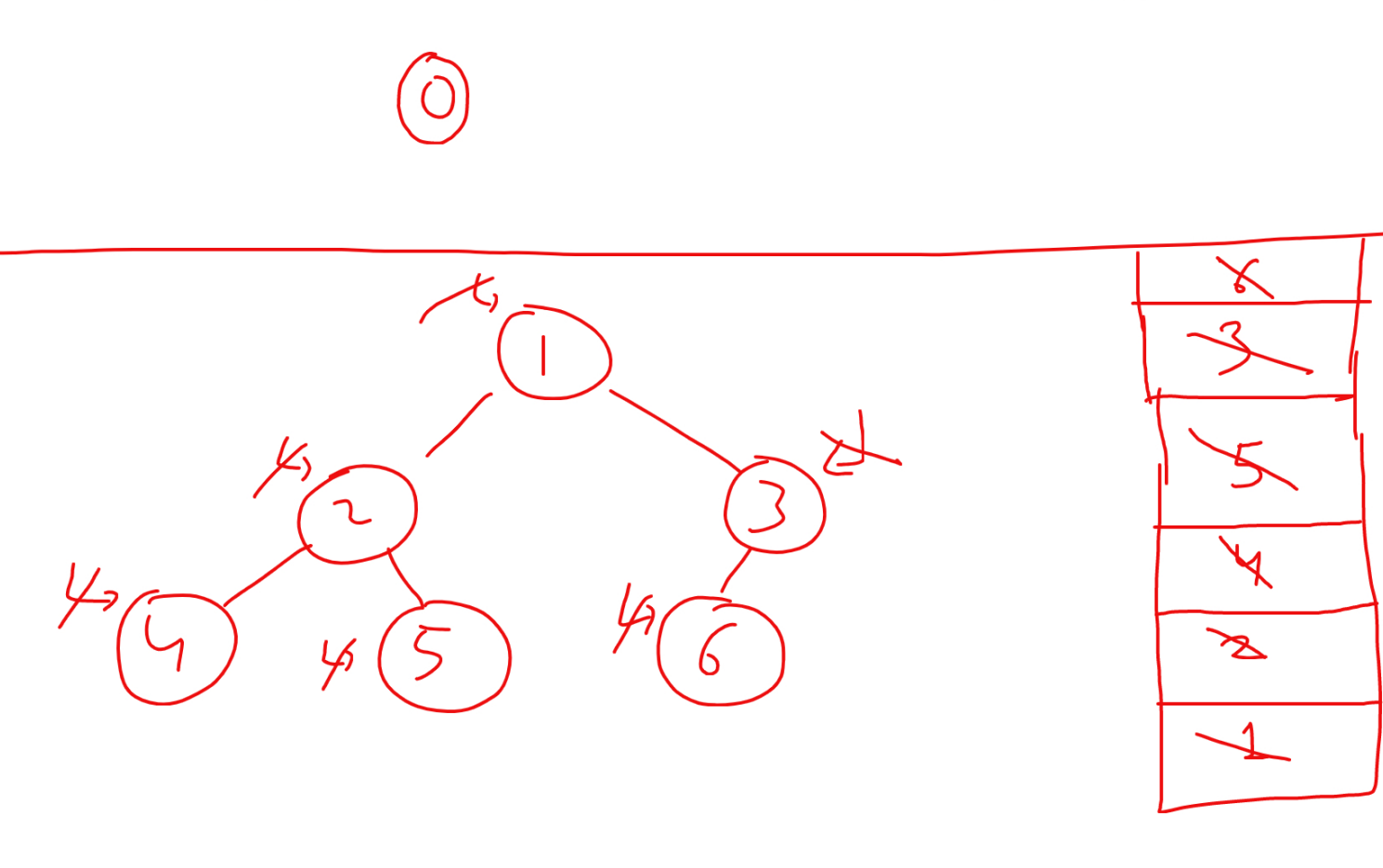
in_start = 0
in_end = 5
index = 3


(3, 1, 4)
0
(5, 2)
(3)
1
(4)
(5)
2
3
4
5
NULL
O/P.

in_start = 4
in_end = 5
index = 5

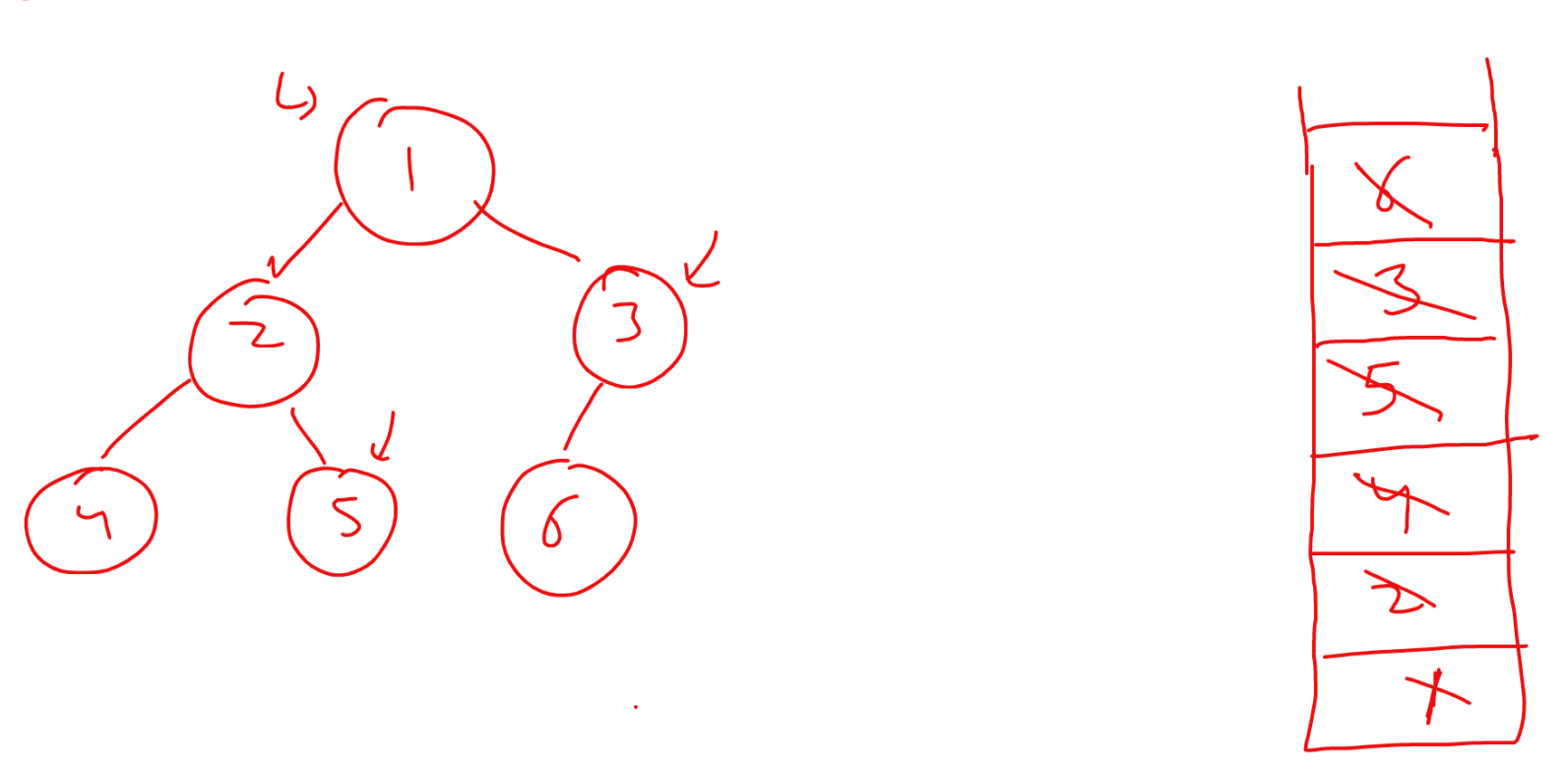
in_start : 6
in_end = 5

⓪

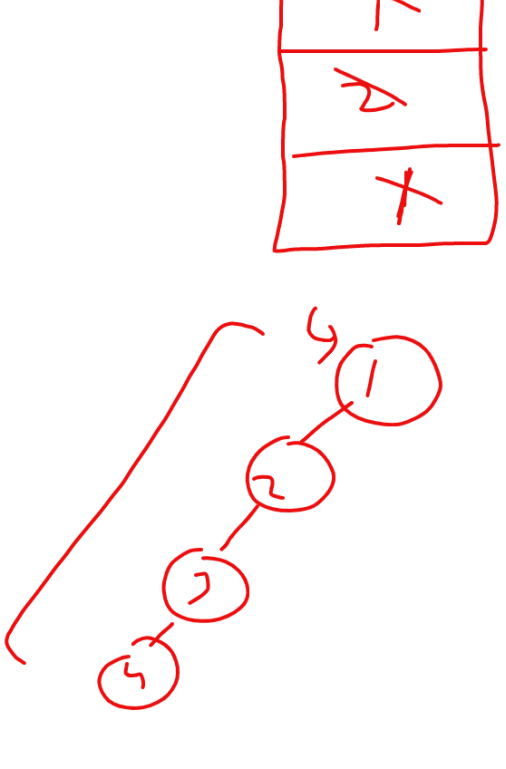

1
2
3
4
5
6
~~6~~
~~3~~
~~5~~
~~4~~
~~2~~
~~1~~

[4, 2, 5, 1, 6, 3]

push_left () → It will keep pushing the subsequent left nodes till it encounters NULL


1
2
3
4
5
6
~~6~~
~~3~~
~~5~~
~~4~~
~~2~~
~~1~~

[4, 2, 5, 1, 6, 3]

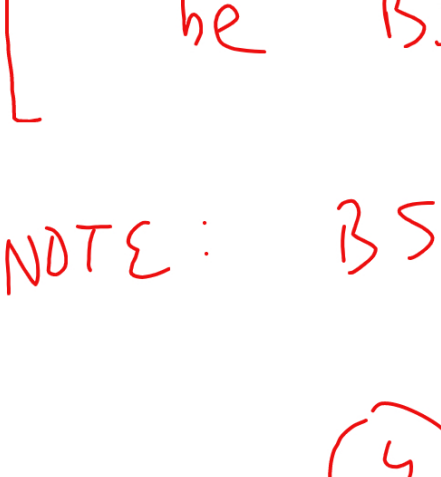


HW:
1) Construct BT from postorder & inorder
2) Iterative preorder traversal
3) Iterative postorder traversal

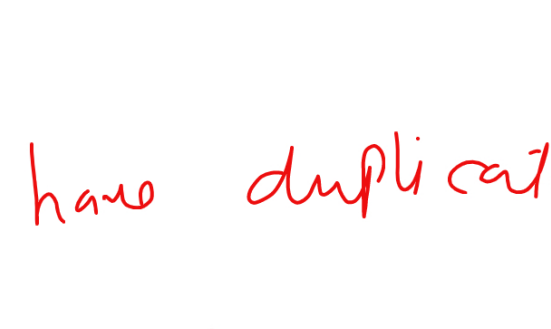
BST

1) The left subtree of a node contains values only less than the node.
2) The right subtree of a node contains values only greater than the node
3) The left & the right subtrees should also be BSTs.

NOTE: BSTs should not have duplicate values.



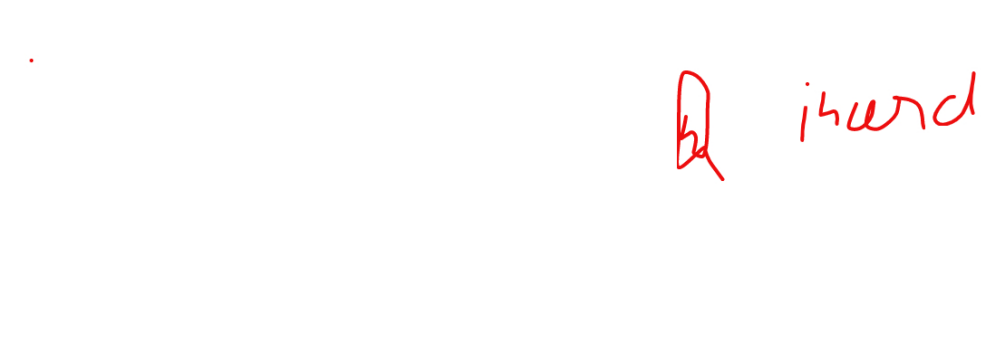
Not BST

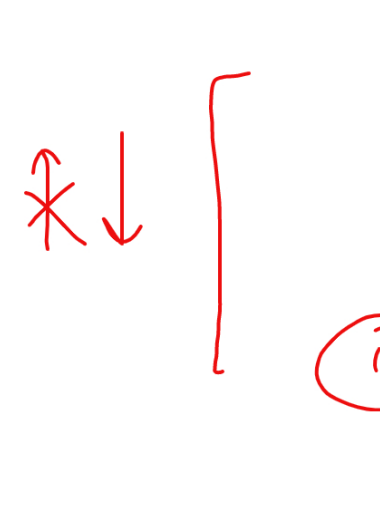


BST.

inorder: [1, 2, 3, 4, 5]

NOTE: Inorder traversal of a BST is always sorted. #



(Search for 4)