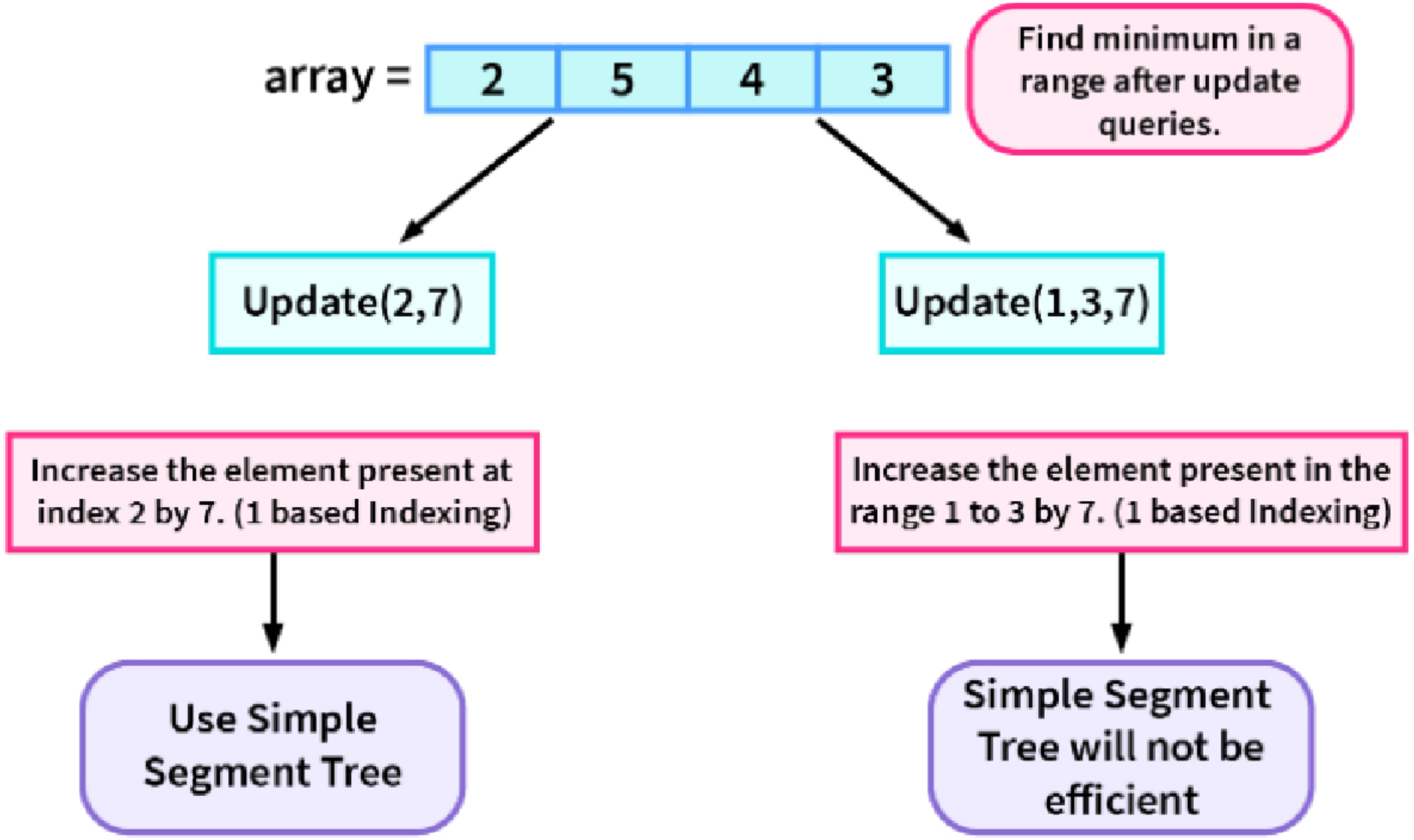
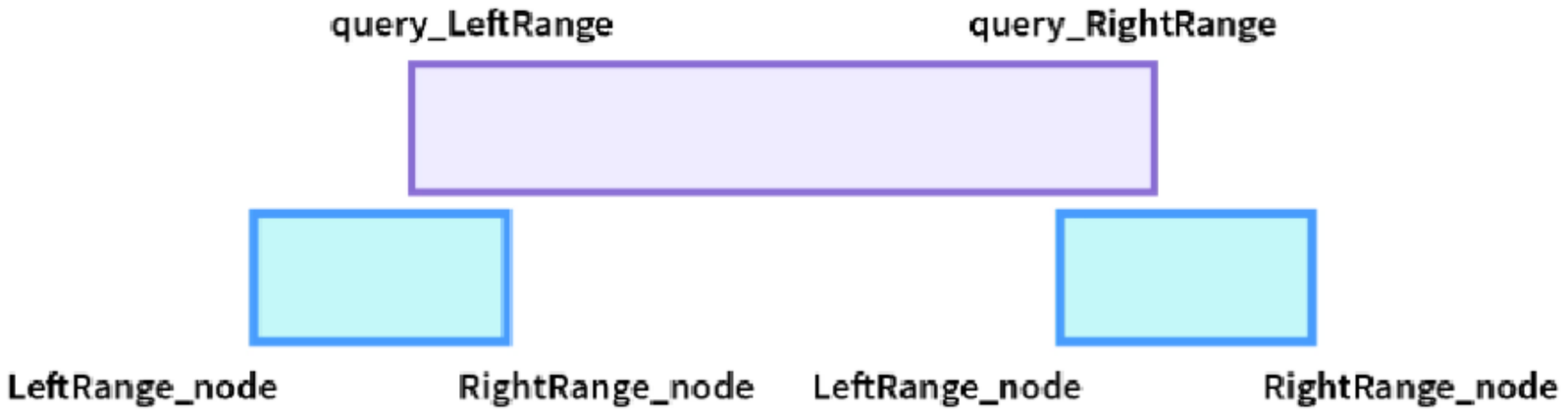
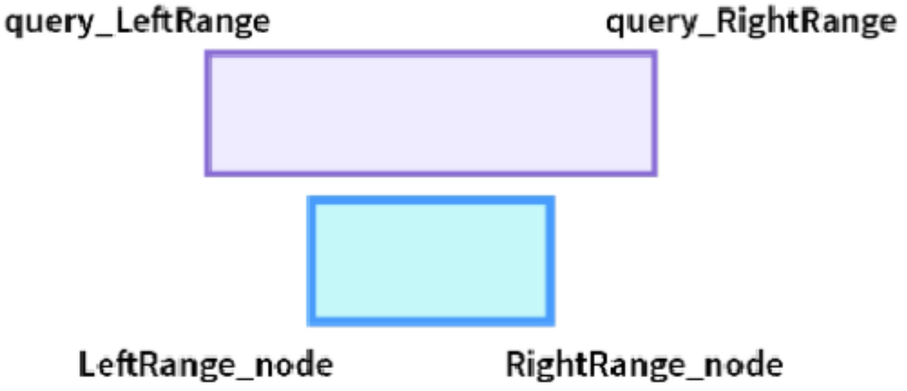
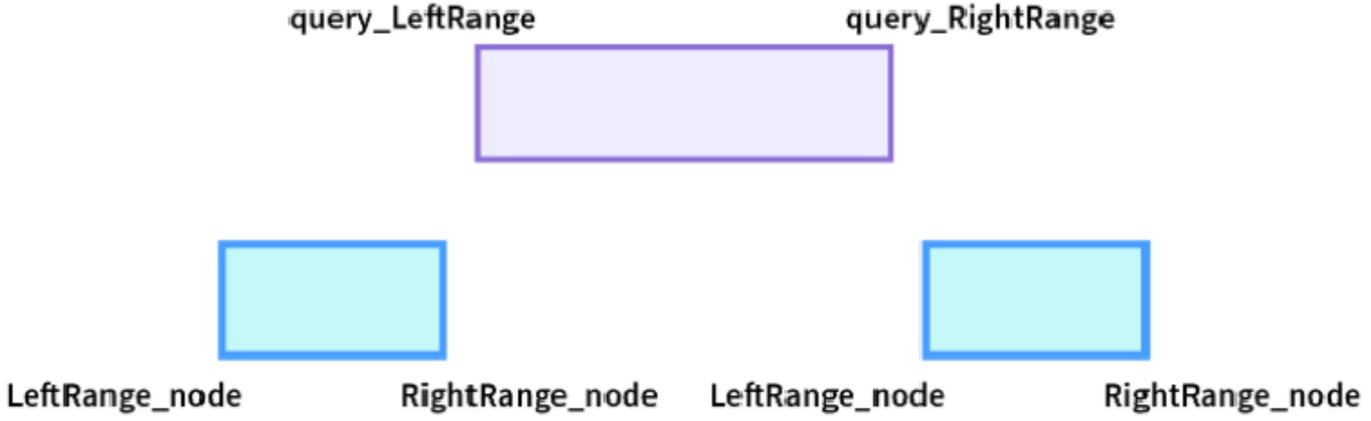
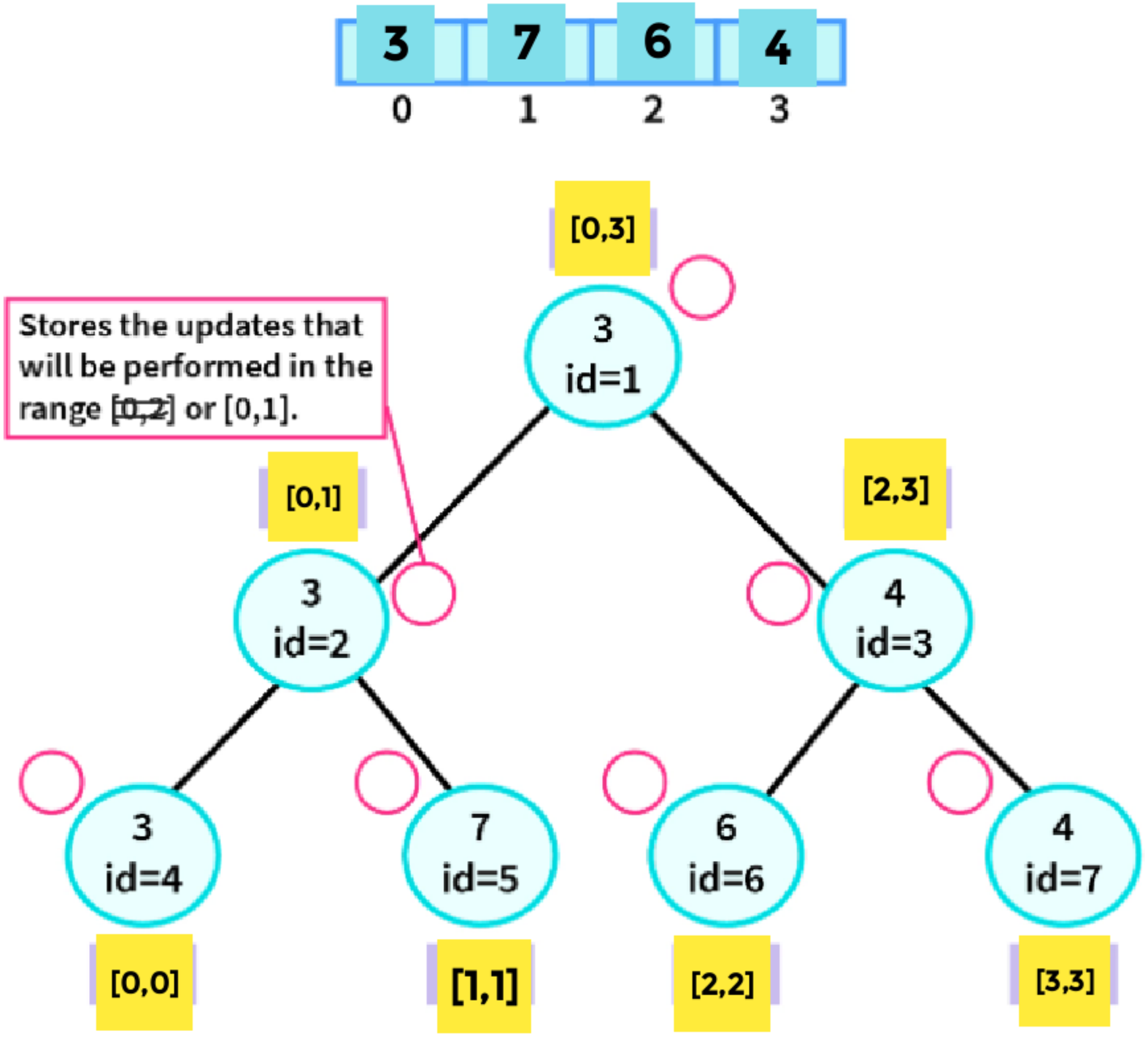
Segment Tree Lazy Propagation


array = 2 5 4 3
Find minimum in a range after update queries.
Update(2,7)
Update(1,3,7)
Increase the element present at index 2 by 7. (1 based Indexing)
Increase the element present in the range 1 to 3 by 7. (1 based Indexing)
Use Simple Segment Tree
Simple Segment Tree will not be efficient


query_LeftRange
query_RightRange
LeftRange_node
RightRange_node
LeftRange_node
RightRange_node

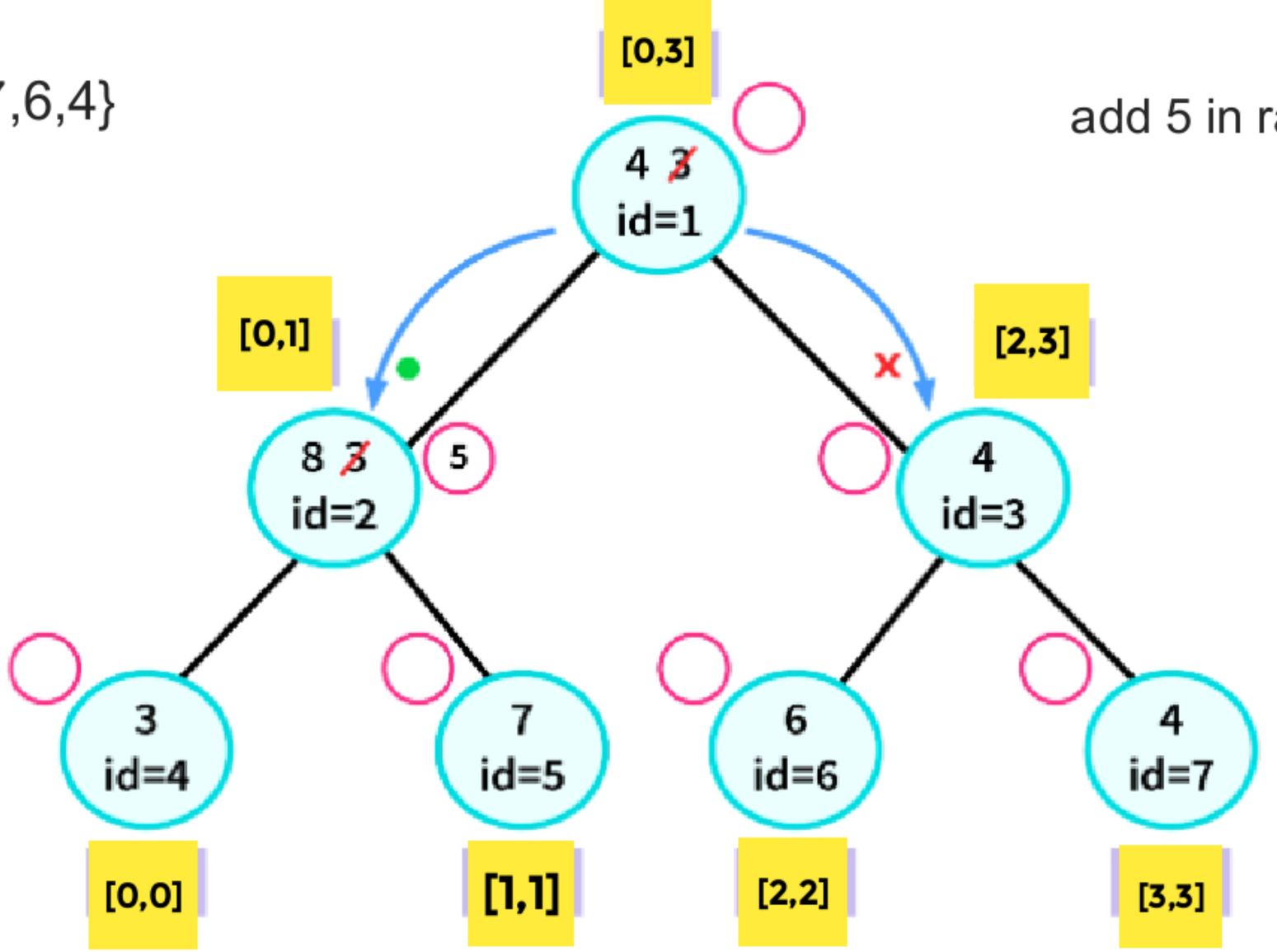

query_LeftRange
query_RightRange
LeftRange_node
RightRange_node


query_LeftRange
query_RightRange
LeftRange_node
RightRange_node
LeftRange_node
RightRange_node


3
7
6
4
0
1
2
3
[0,3]
3
id=1
Stores the updates that will be performed in the range [0,2] or [0,1].
[0,1]
[2,3]
3
id=2
4
id=3
3
id=4
7
id=5
6
id=6
4
id=7
[0,0]
[1,1]
[2,2]
[3,3]

Array [] = {3,7,6,4}

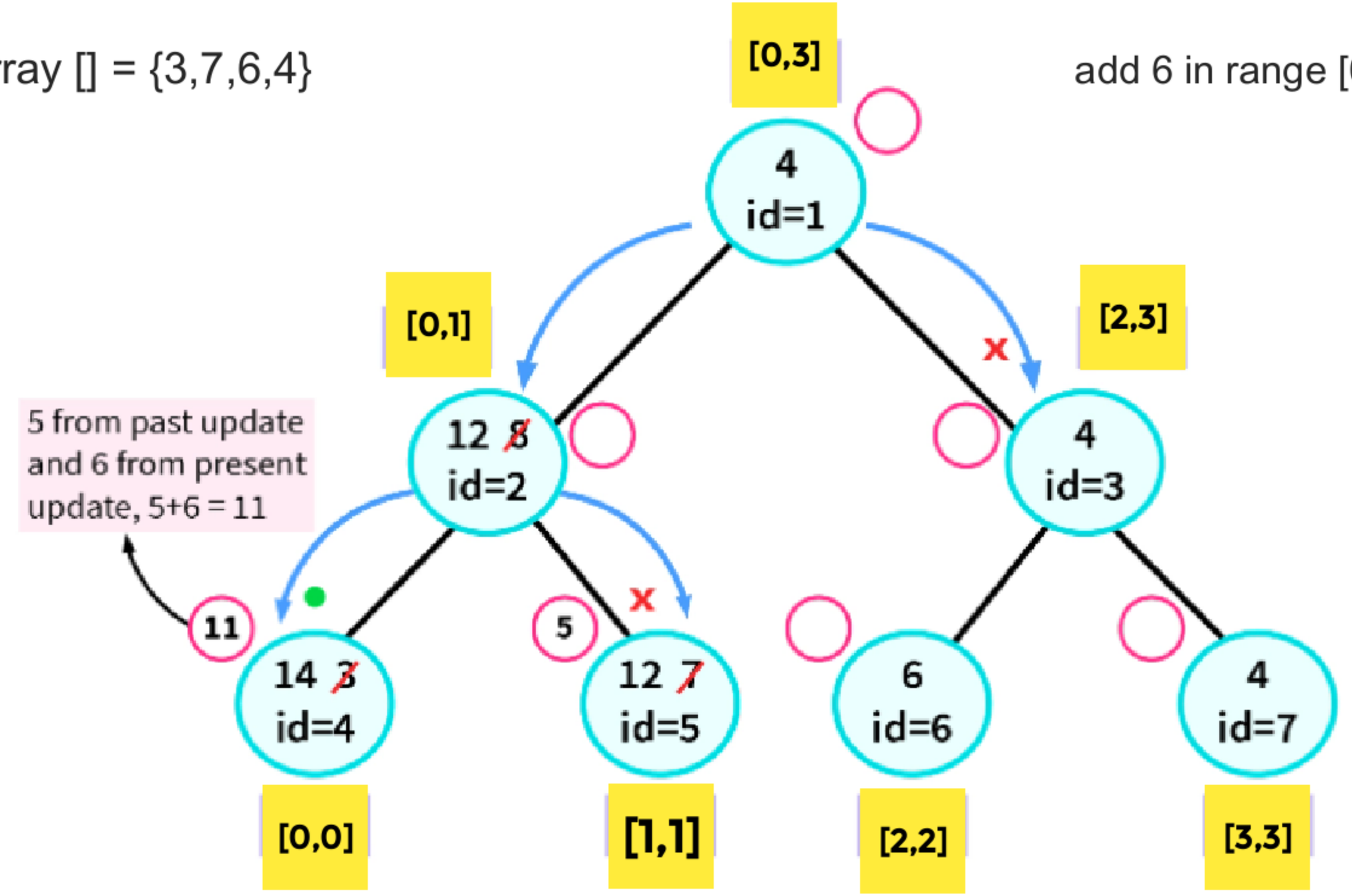
add 5 in range [0 to 1]



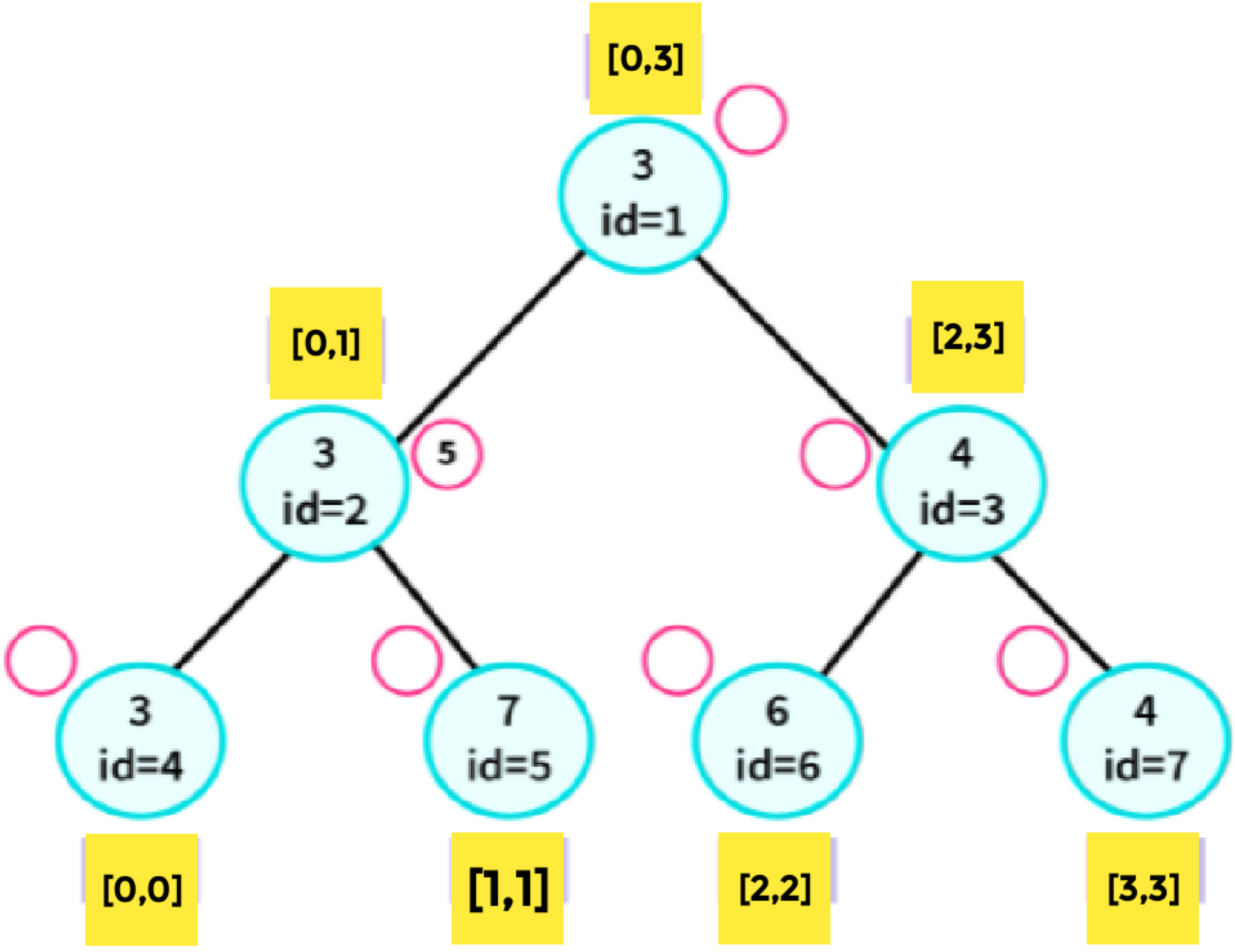
SCALER Topics

Array [] = {3,7,6,4}

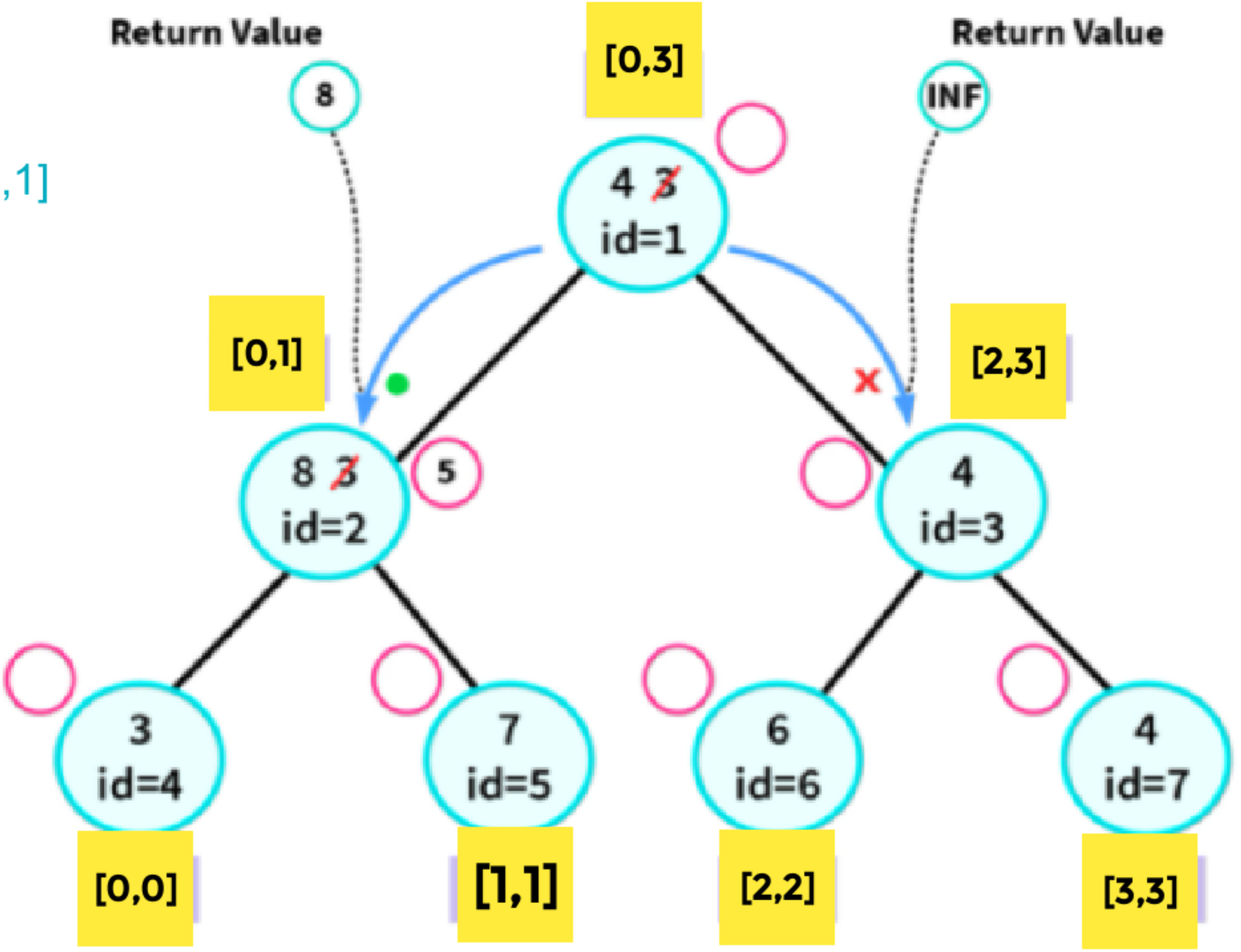
add 6 in range [0 to 0]



```
void LazySegmentTreeRangeUpdate(updateValue,query_LeftRange,query_RightRange)
{
    if(no overlap condition)
    {
         return from the dfs call.
    }
    else if(total overlap condition)
    {
        update the node of the segment tree with updateValue.
        update the copy node associated with it with updateValue.
        return from the dfs call.
    }

    //Here we will handle the condition of partial overlap.
    if(value present in the copy node is not empty)
    {
        propagate its value to the left and right child.
        update the copy nodes associated with the child nodes.
    }

    Move to the left and right child of the current node.

    While backtracking update the value of the parent node
    with value present in its child node after the update.
}
```


[0,3]
3
id=1
[0,1]
[2,3]
3
id=2
5
4
id=3
3
id=4
7
id=5
6
id=6
4
id=7
[0,0]
[1,1]
[2,2]
[3,3]

Query min [0,1]


Return Value
8
[0,3]
Return Value
INF
4 3
id=1
[0,1]
[2,3]
8 3
id=2
5
4
id=3
3
id=4
7
id=5
6
id=6
4
id=7
[0,0]
[1,1]
[2,2]
[3,3]