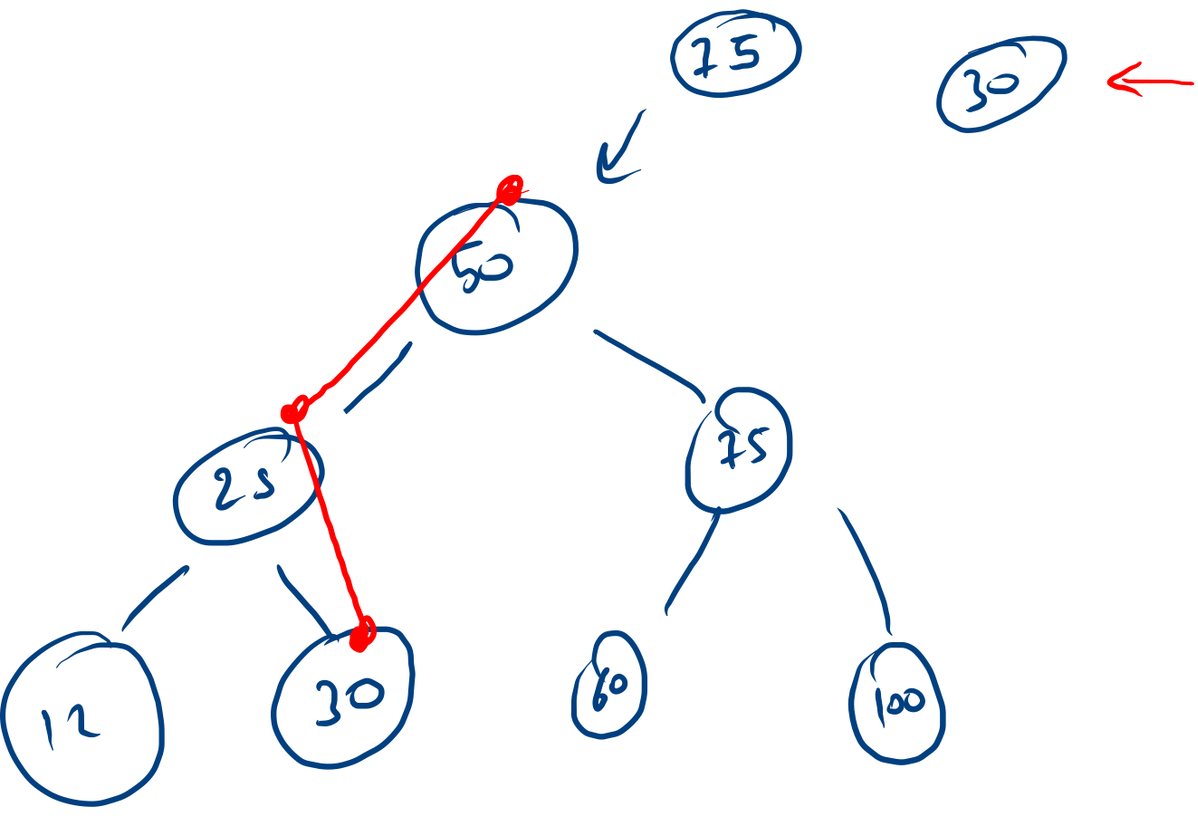
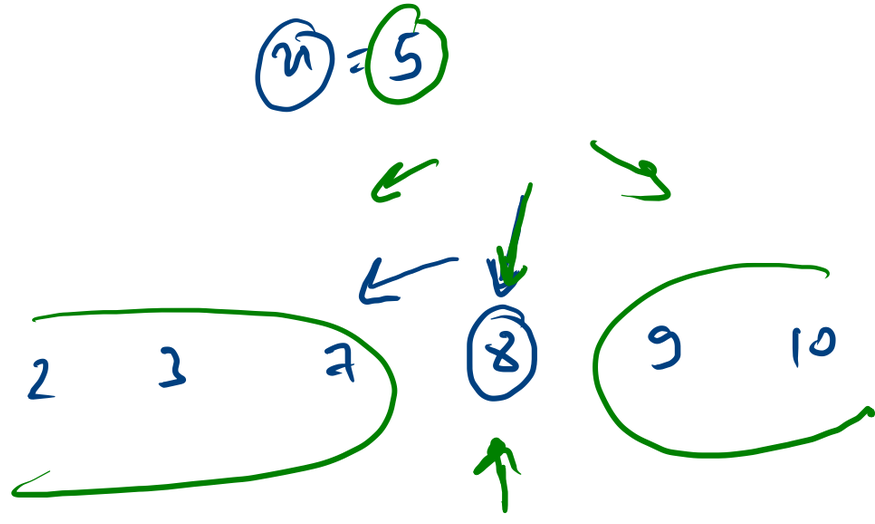
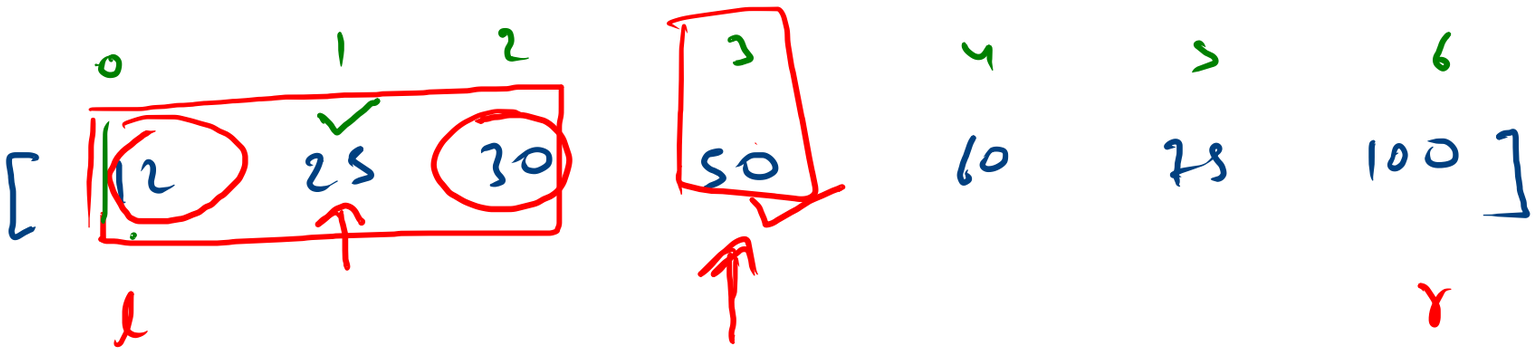
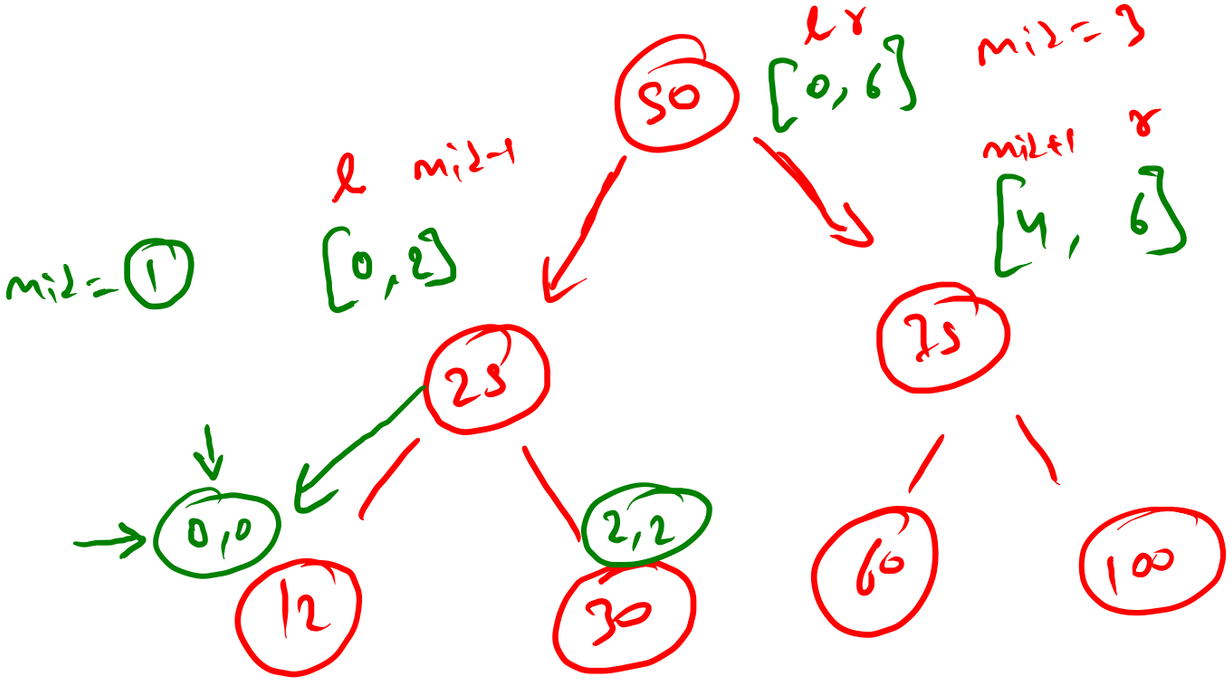

75
30
50
25
75
12
30
60
100


n = 5
2
3
7
8
9
10



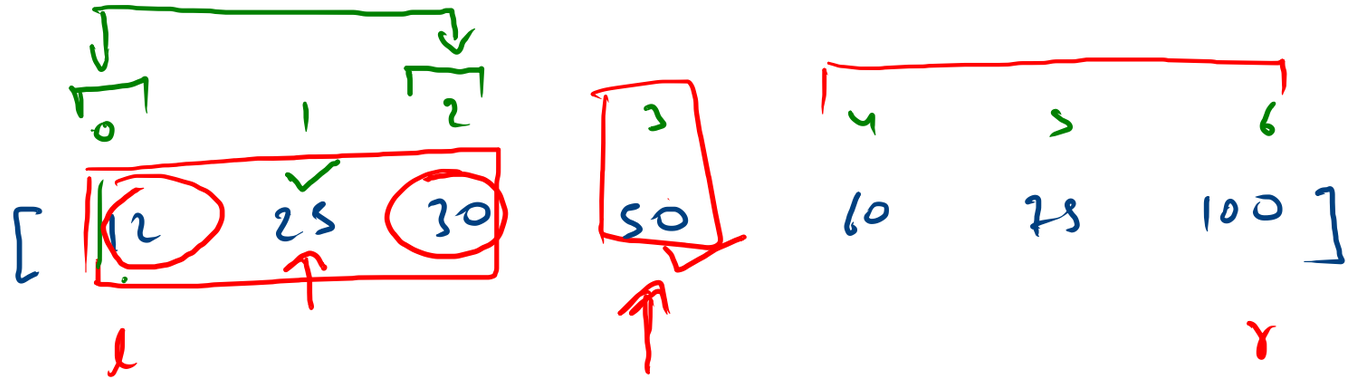


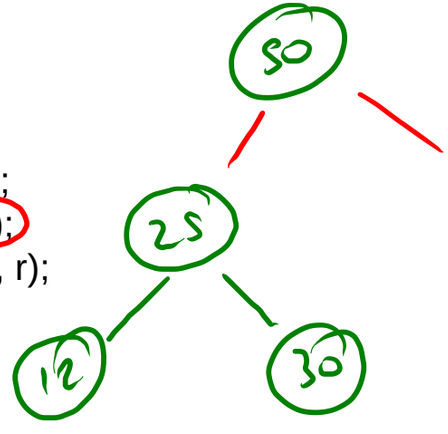
l , r

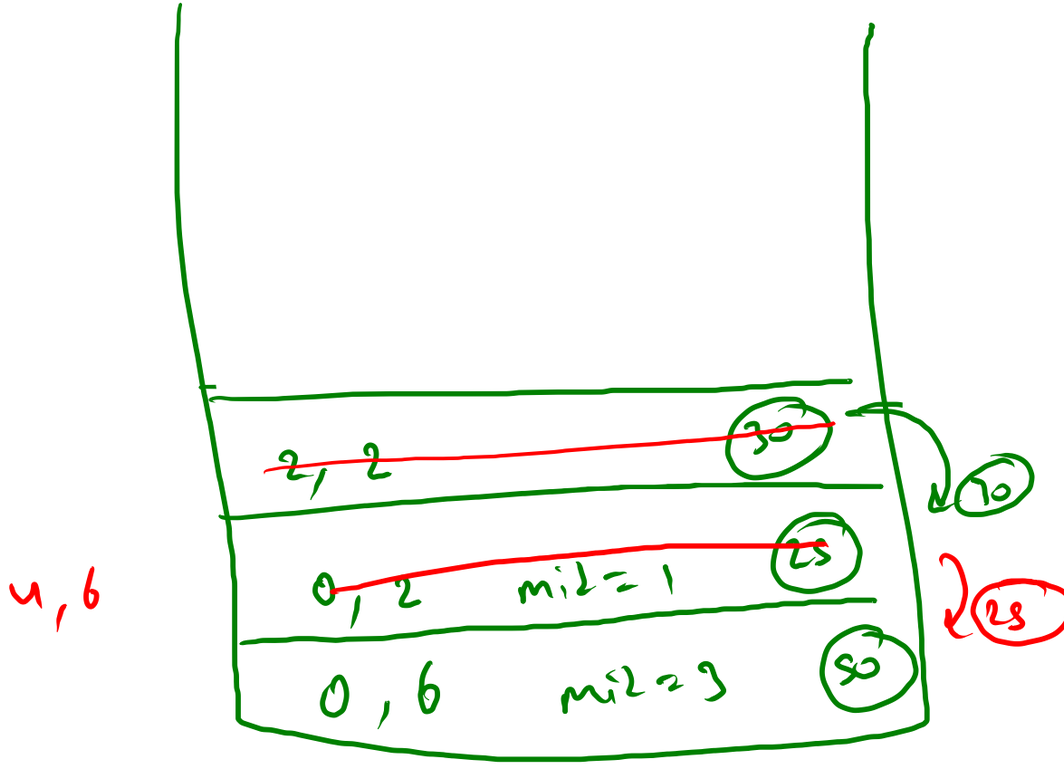
mid = (l + r) / 2

hold of arr[mid]

```
public static Node construct(int arr[], int l, int r){
        if(l==r){
          return new Node(arr[l]);
        }

        int mid = (l+r)/2;

        Node root = new Node(arr[mid]);
        root.left = construct(arr, l, mid-1);
        root.right = construct(arr, mid+1, r);

        return root;
    }
```







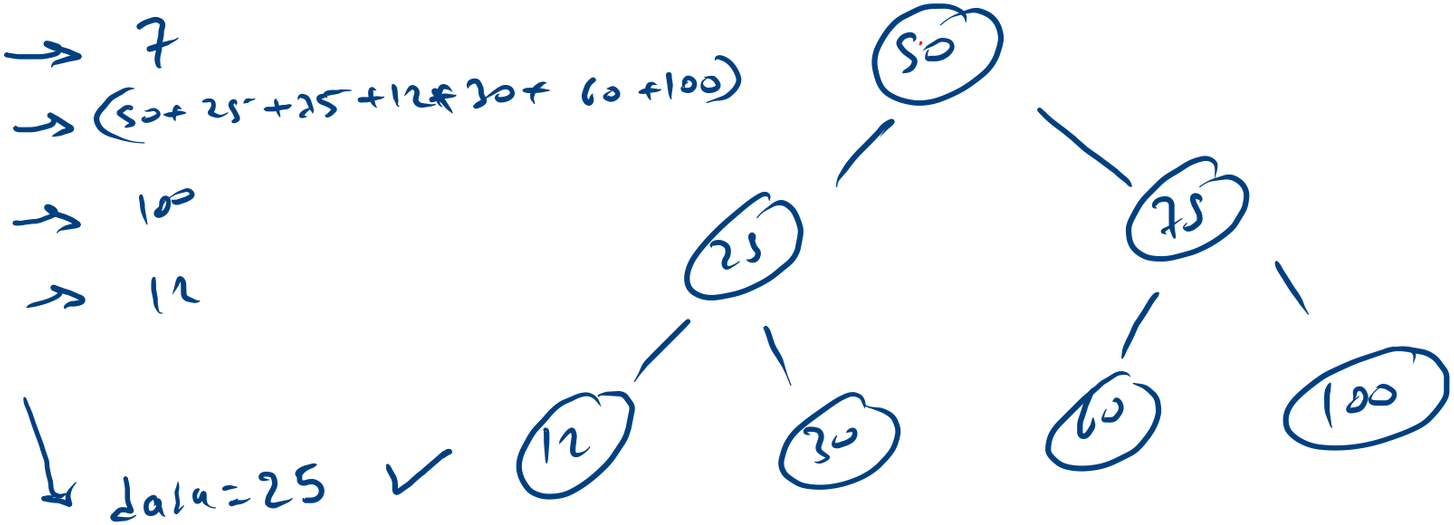
size → 7

sum → (50+25+75+12+30+60+100)

max → 100

min → 12

find

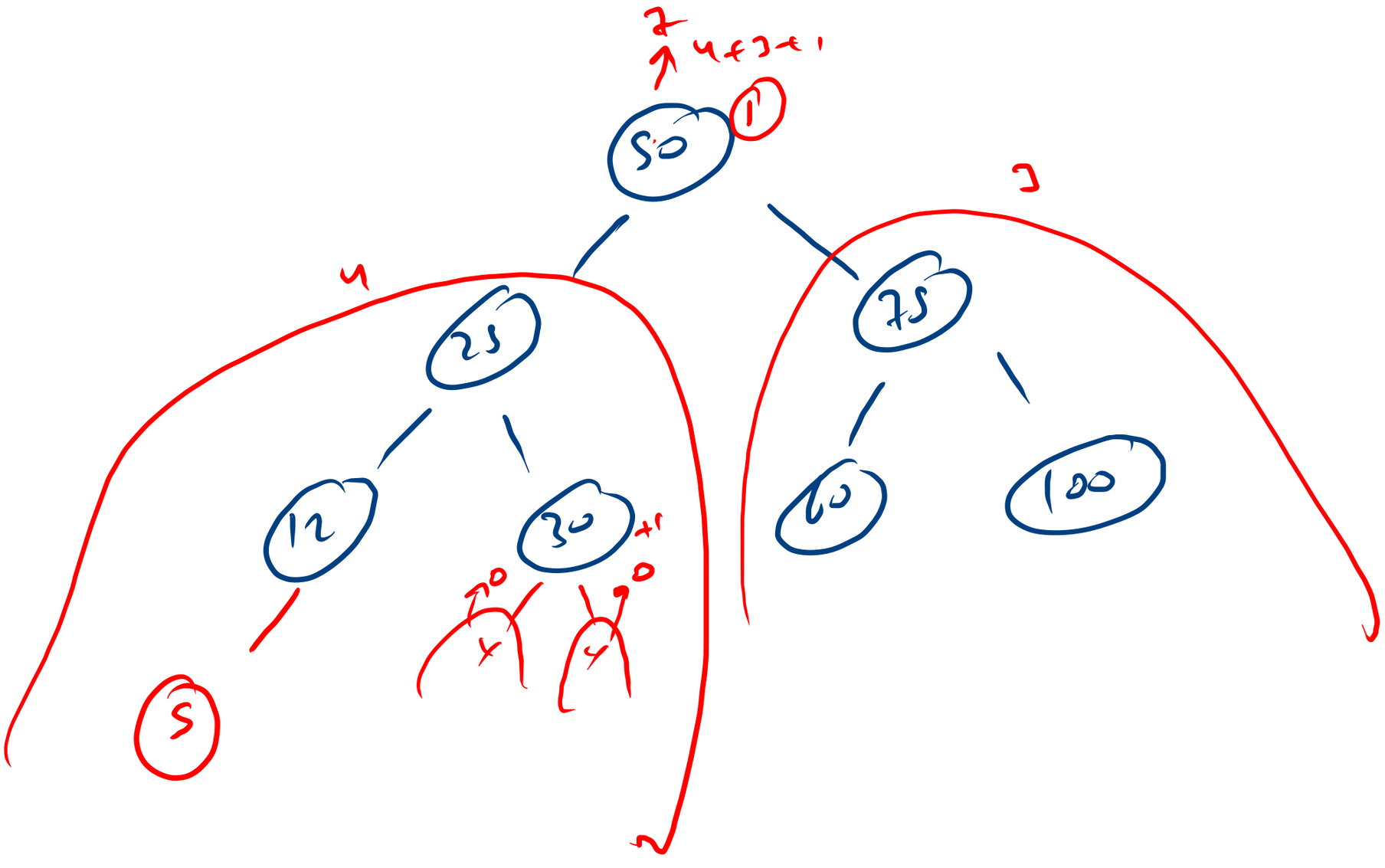


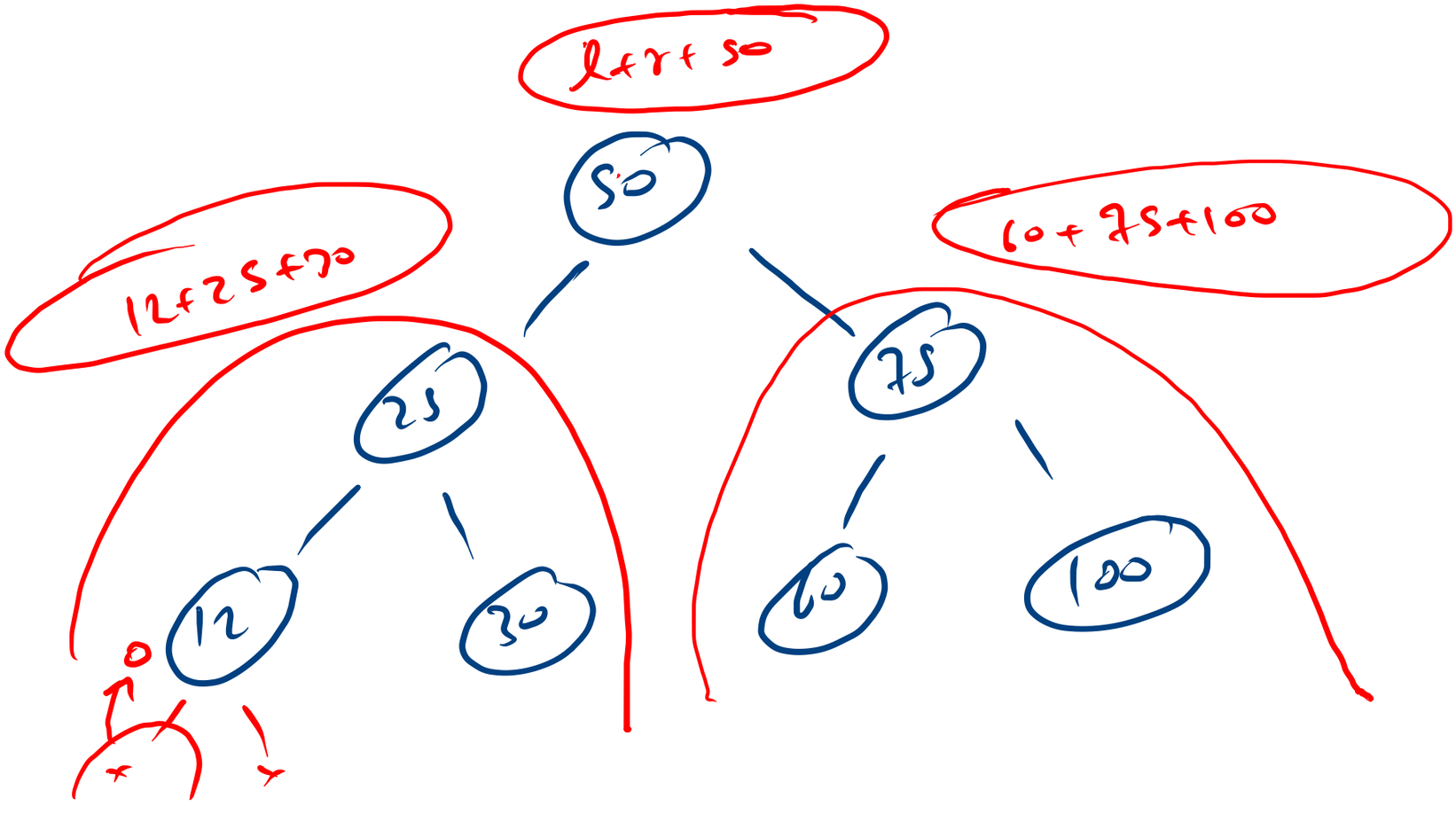
data=25 ✓

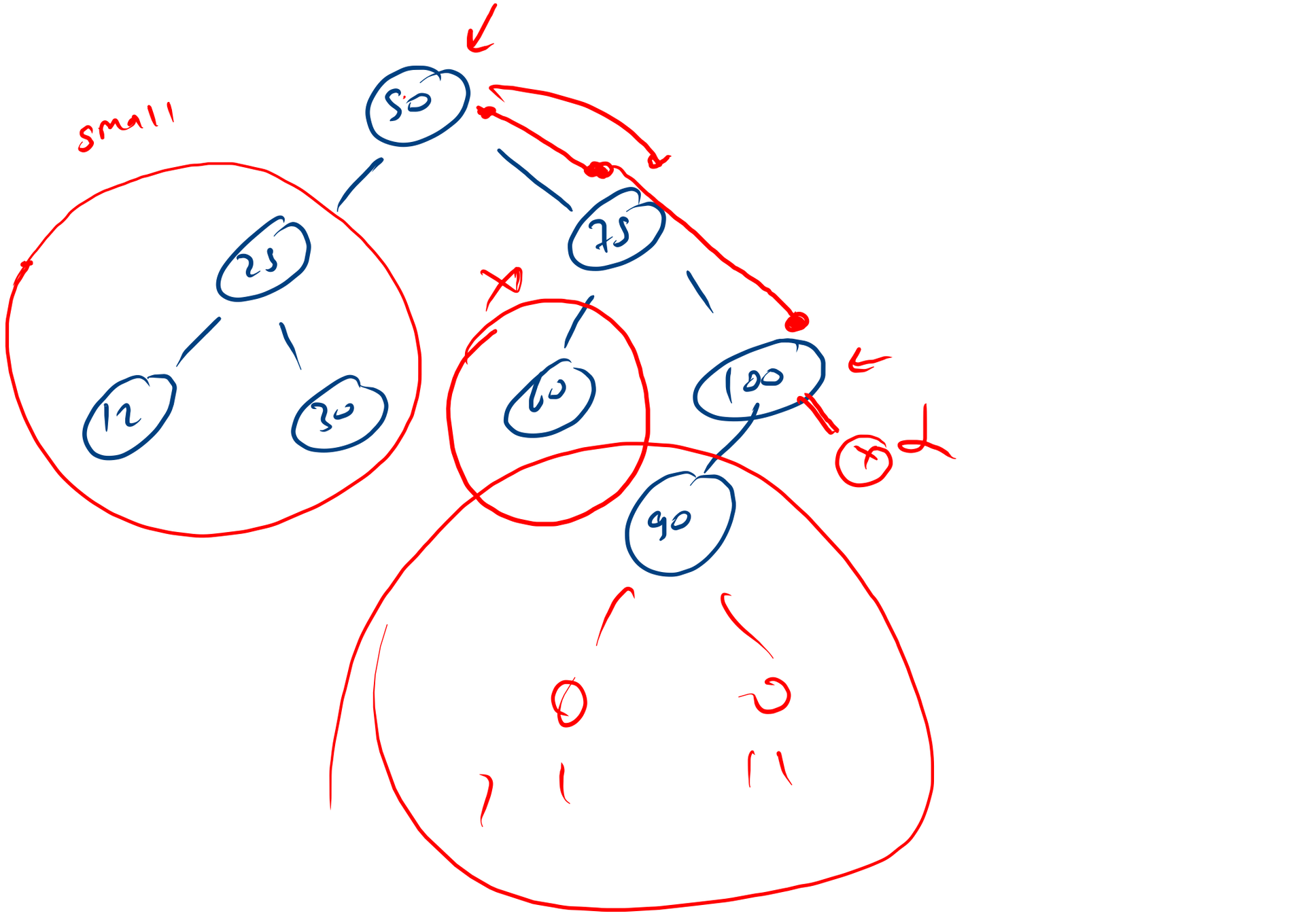
24 q

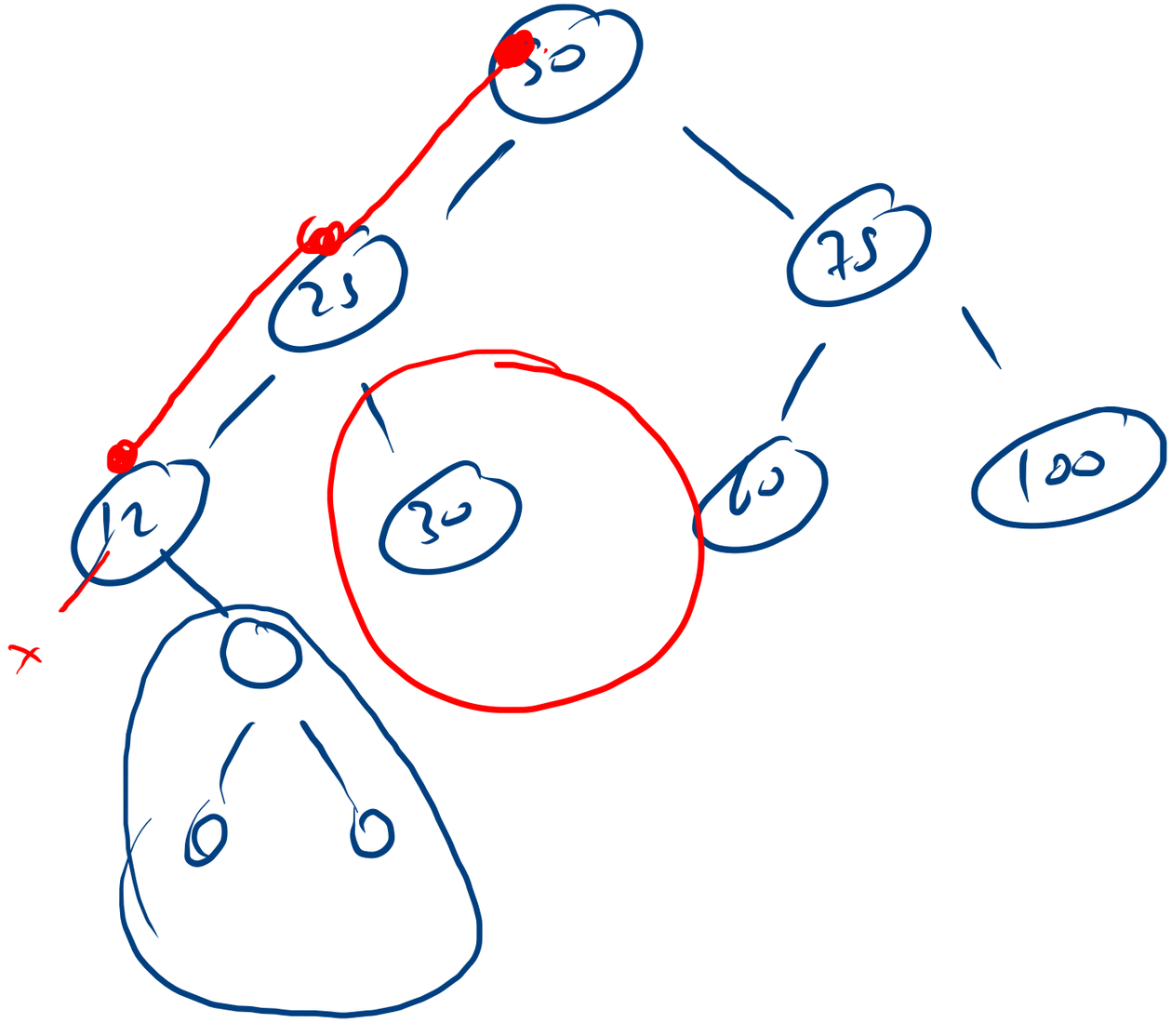
BT

BST ← order


7
4+3+1
50
1
3
4
25
75
12
30
+1
60
100
5
0
0
4
4

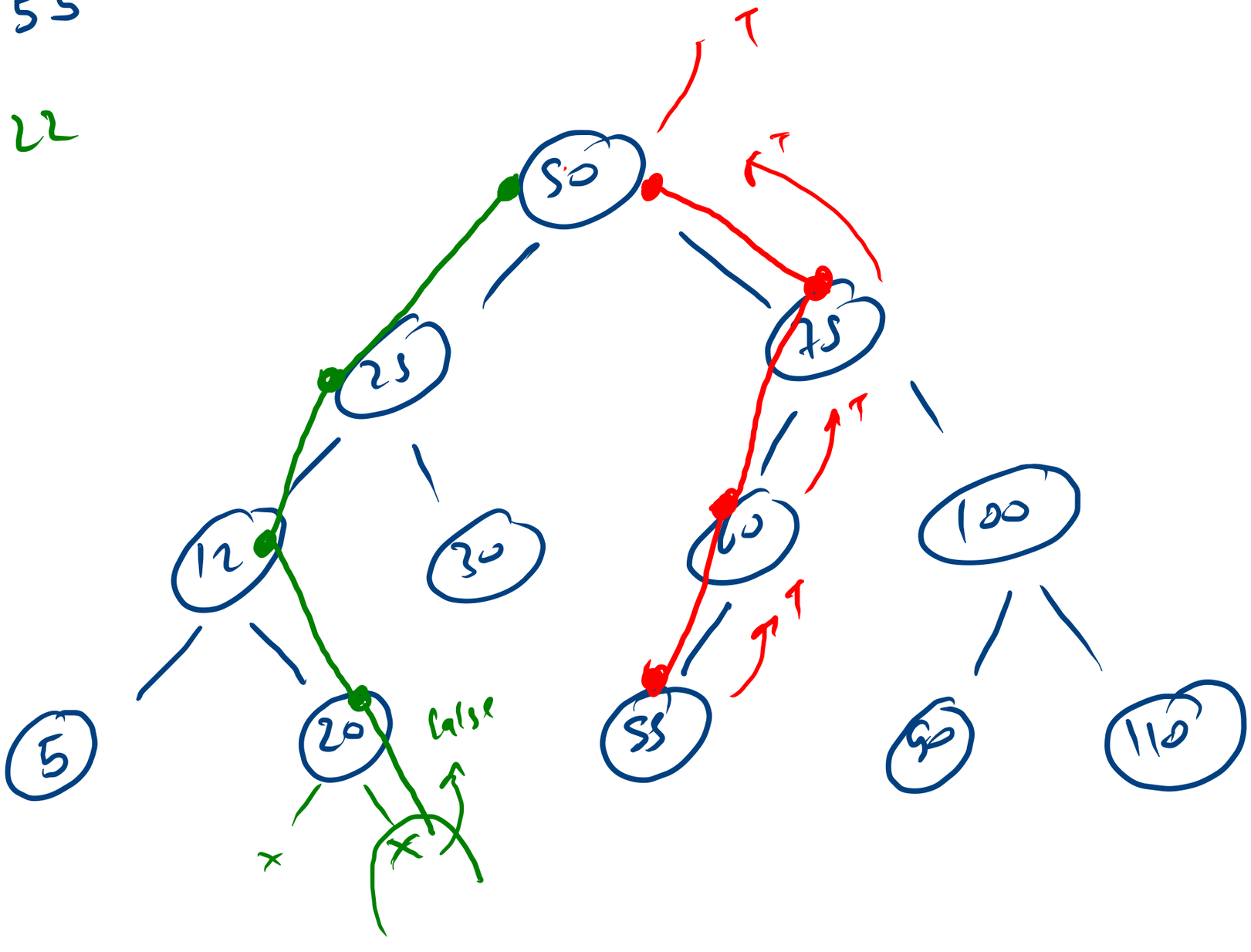

l+r+50
50
12+25+30
60+75+100
25
75
12
30
60
100
0
x
y


node = 50 75 100
small
50
25
75
12
30
60
100
90
0
3
1
1
11


50
25
75
12
30
60
100
x

data = 55

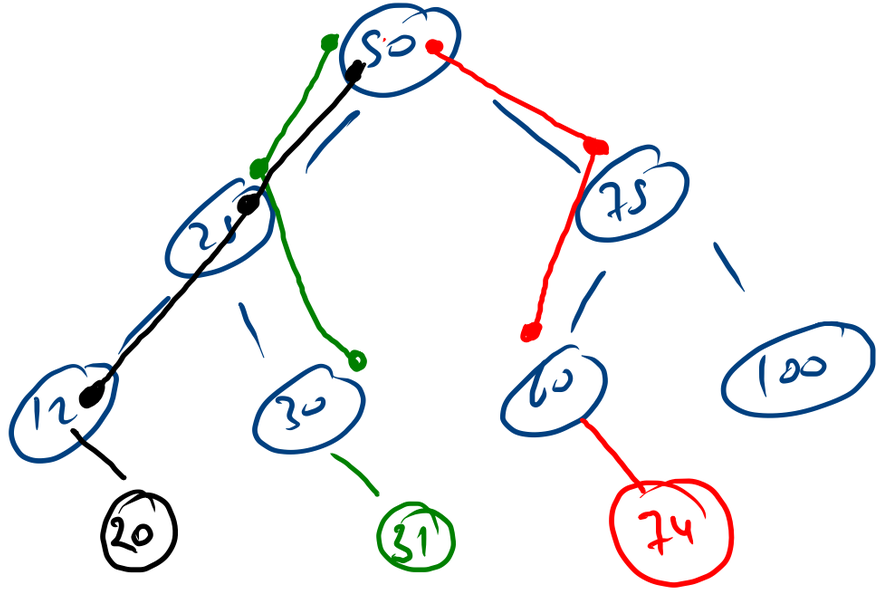
data = 22



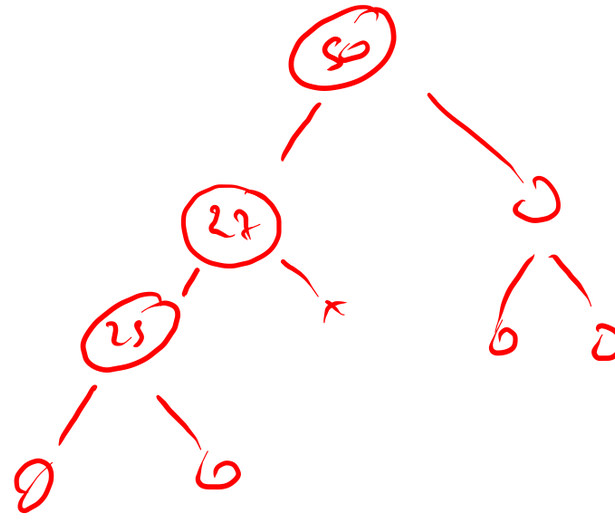
data → 31

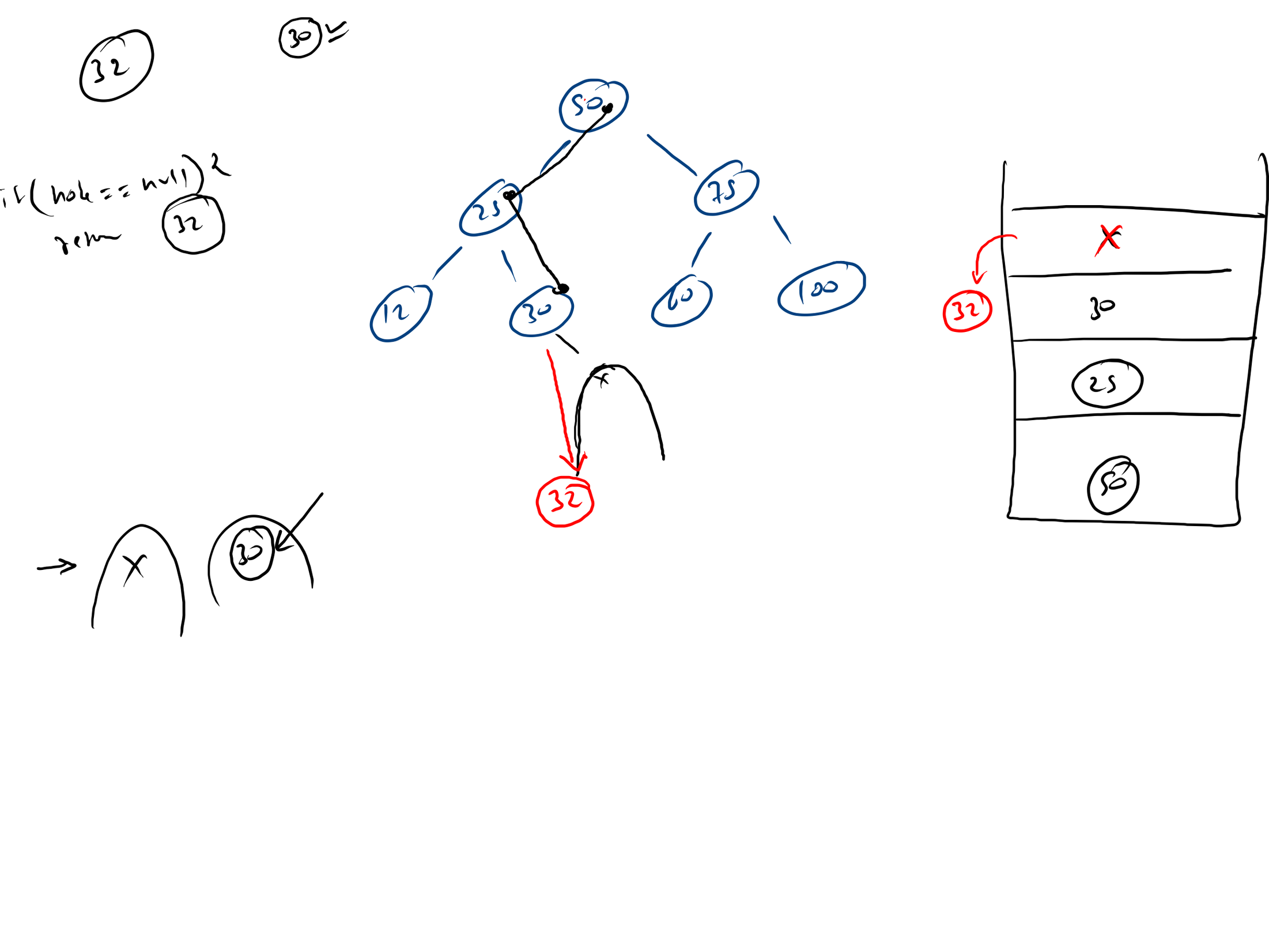
data = 74

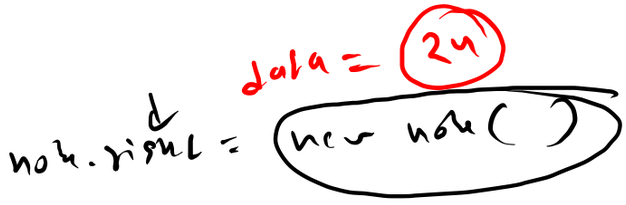
data = 20

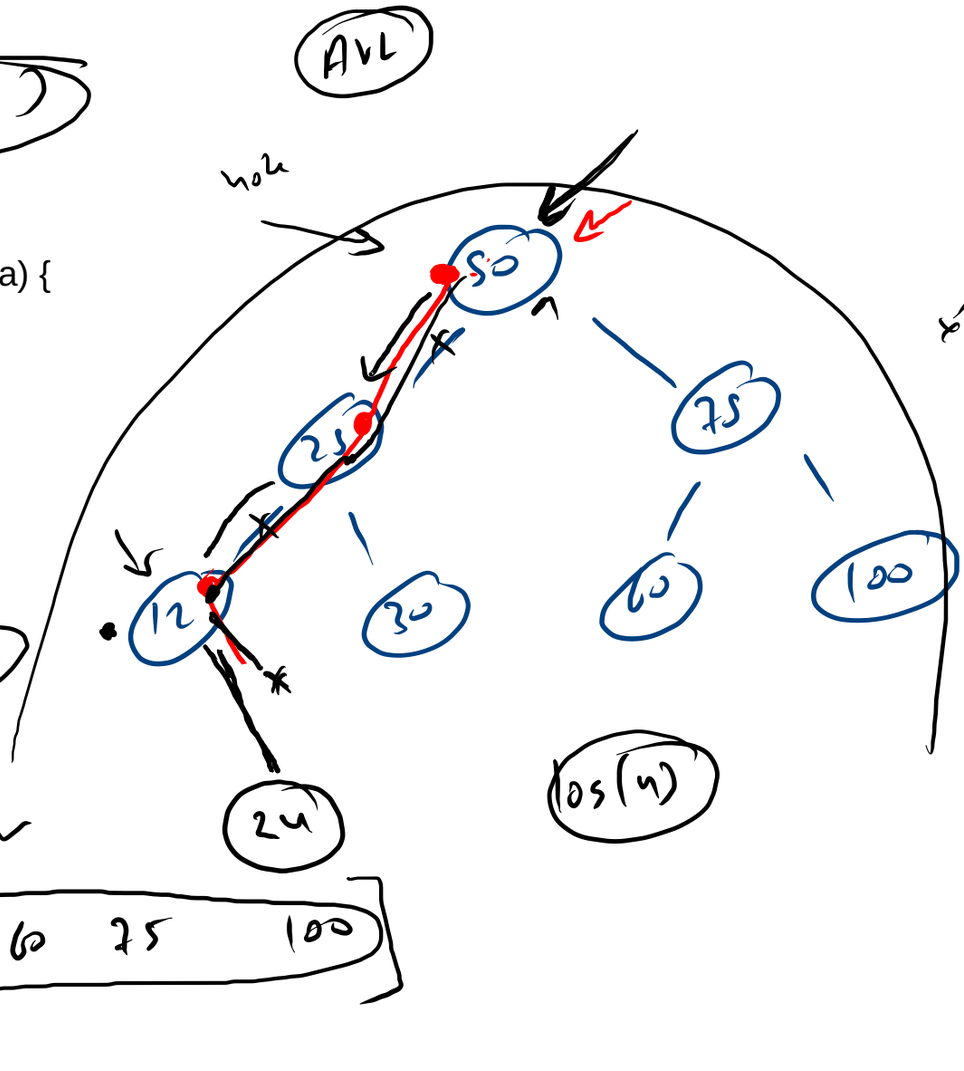


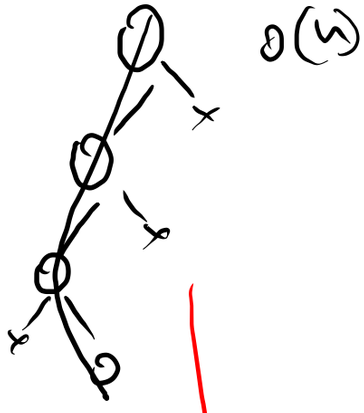
27



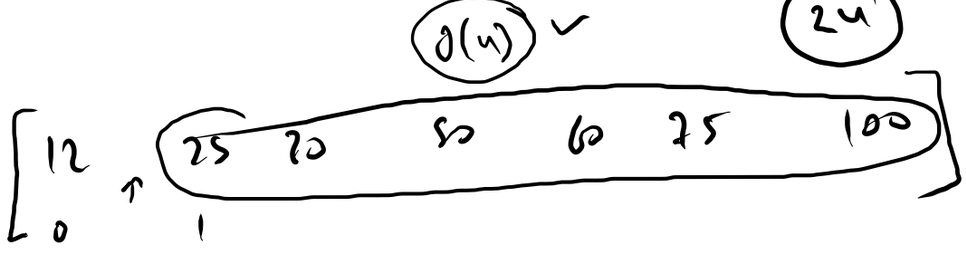

32
30
if(node == null)
return
32
50
25
75
12
30
60
100
32
X
30
32
X
30
25
50

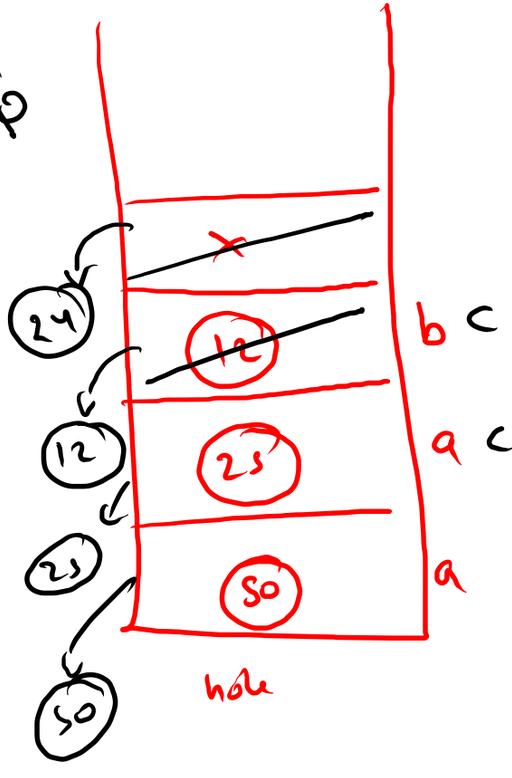




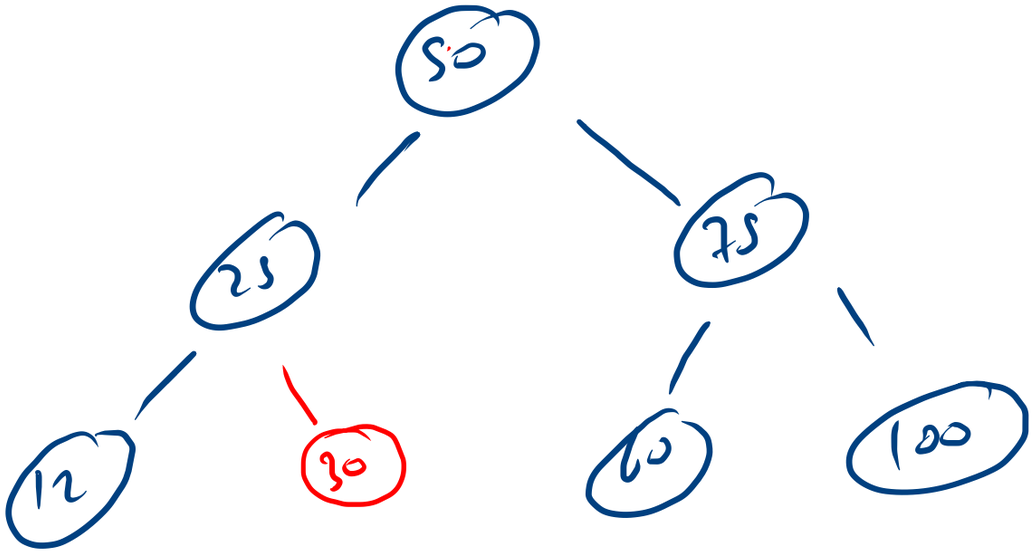



```
public static Node add(Node node, int data) {

    if(node == null){
        return new Node(data, null, null);
    }
    24 < 12
    if(data < node.data){
a       node.left = add(node.left, data);
    }else if(data > node.data){   24>12
b       node.right = add(node.right, data);
    }

c   return node;
}
```

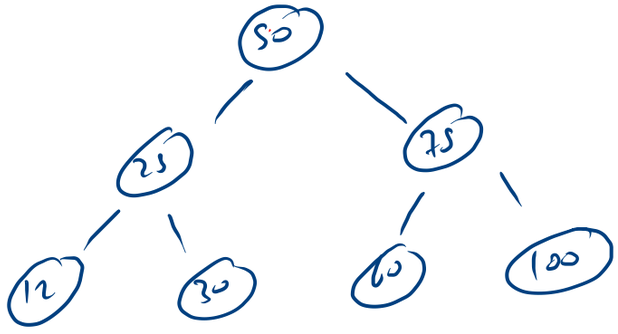


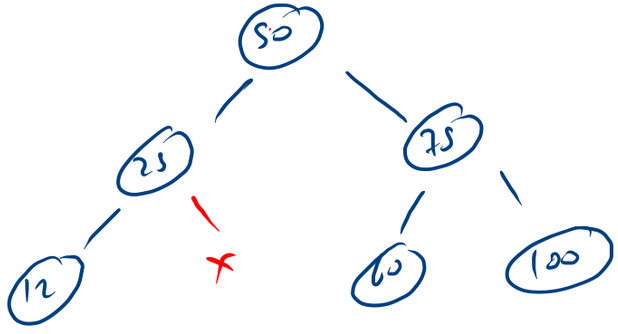


dala = 30

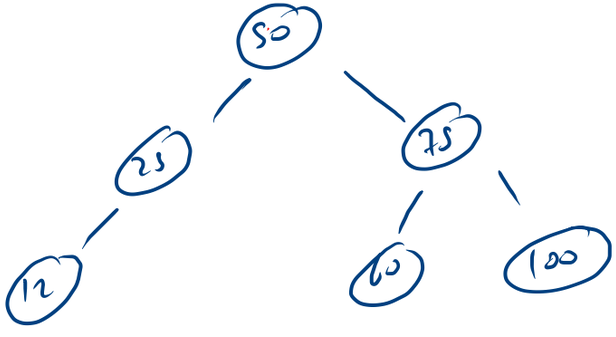


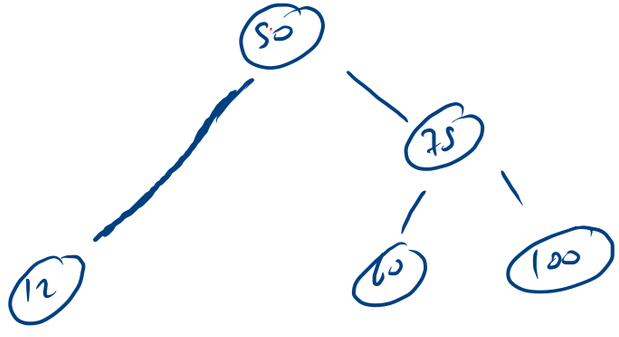
data → 30



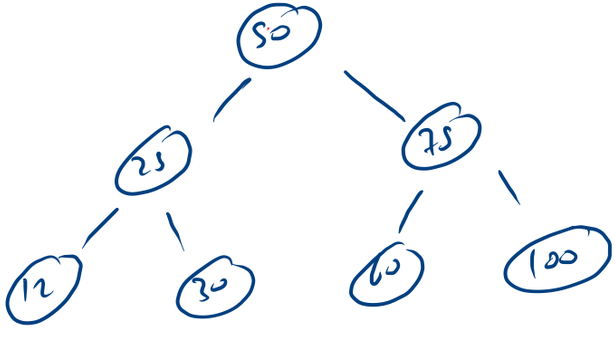


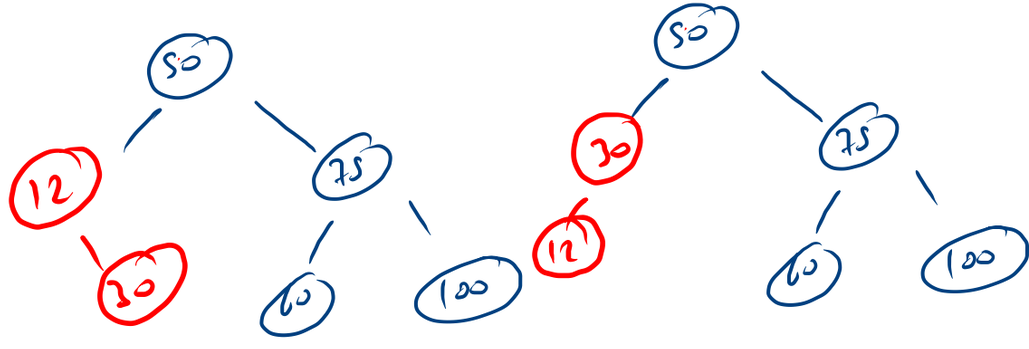
data = 25



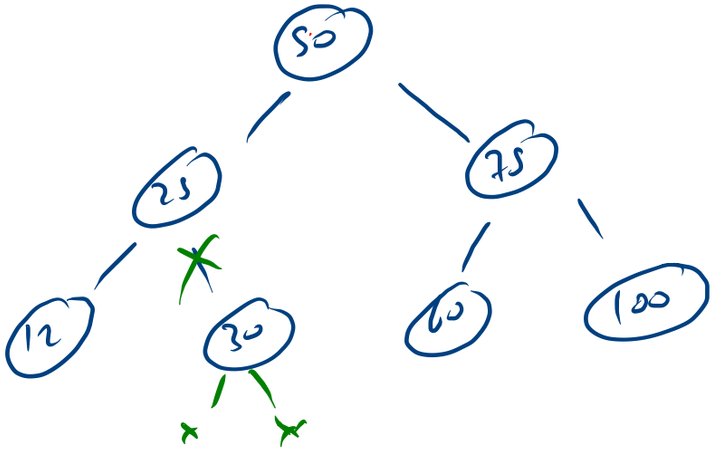


data = 25

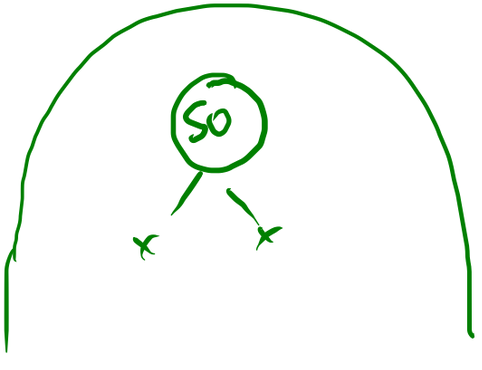




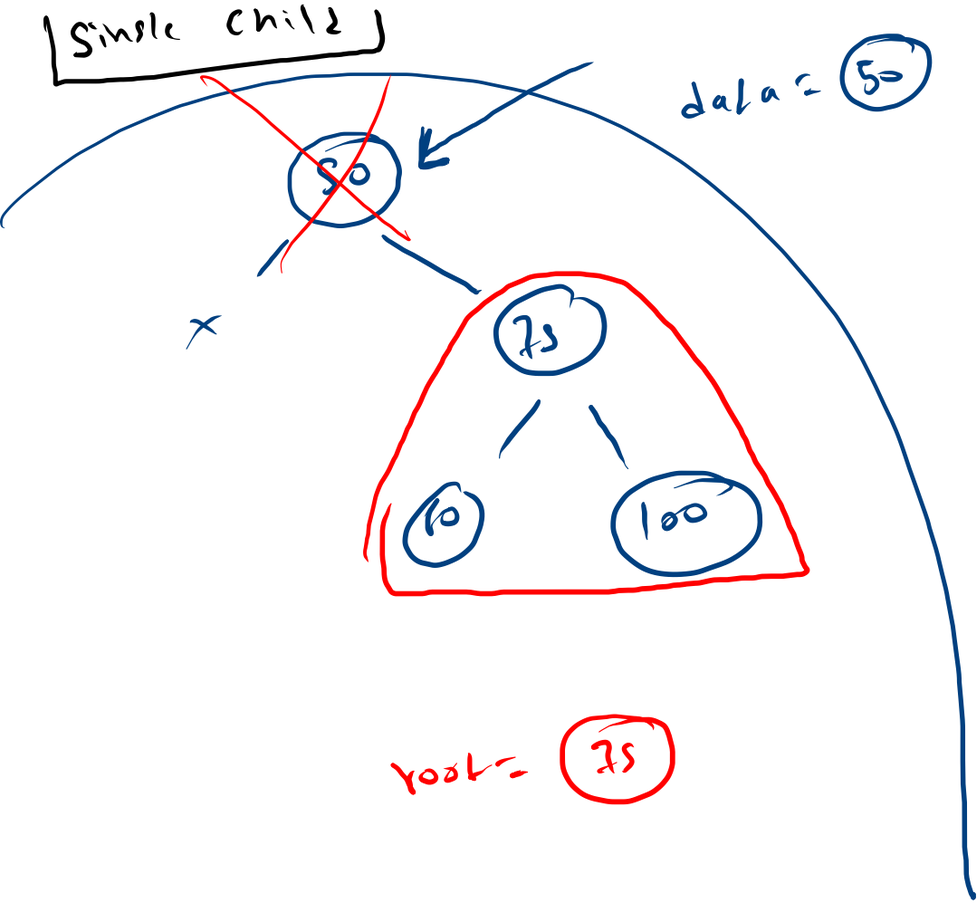
delete 30

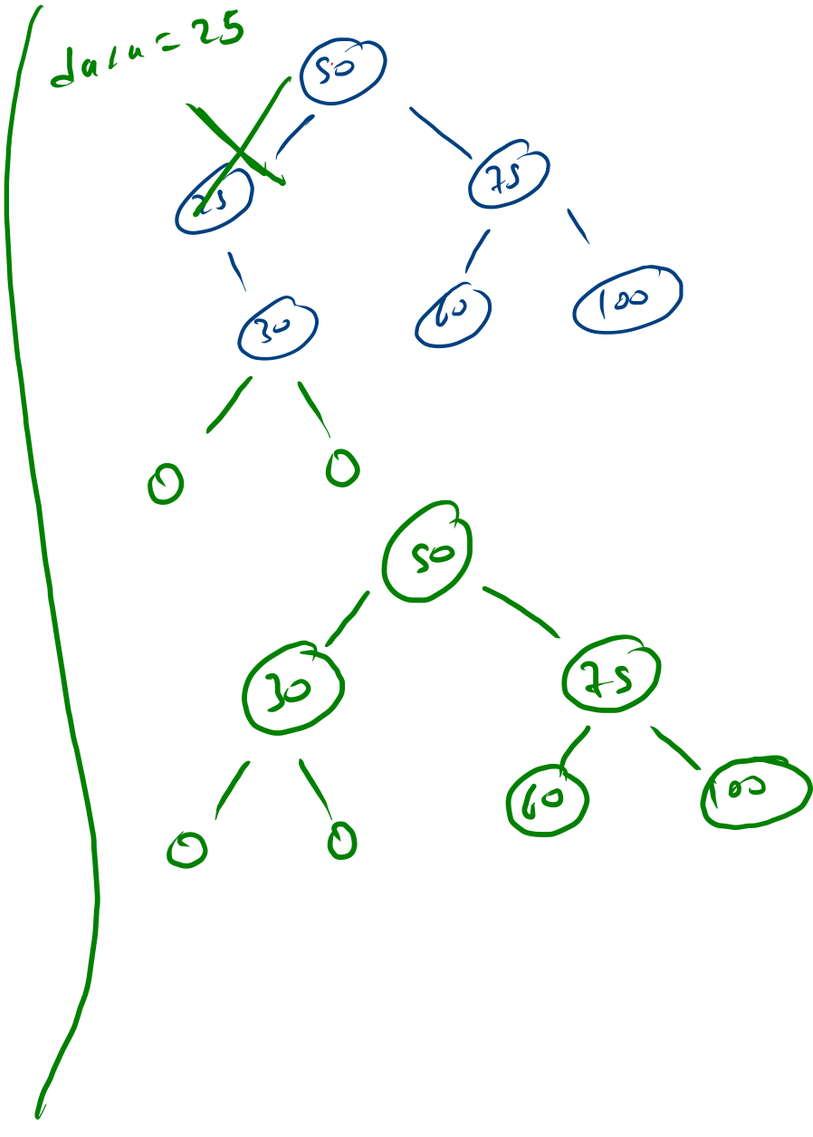


delete 50

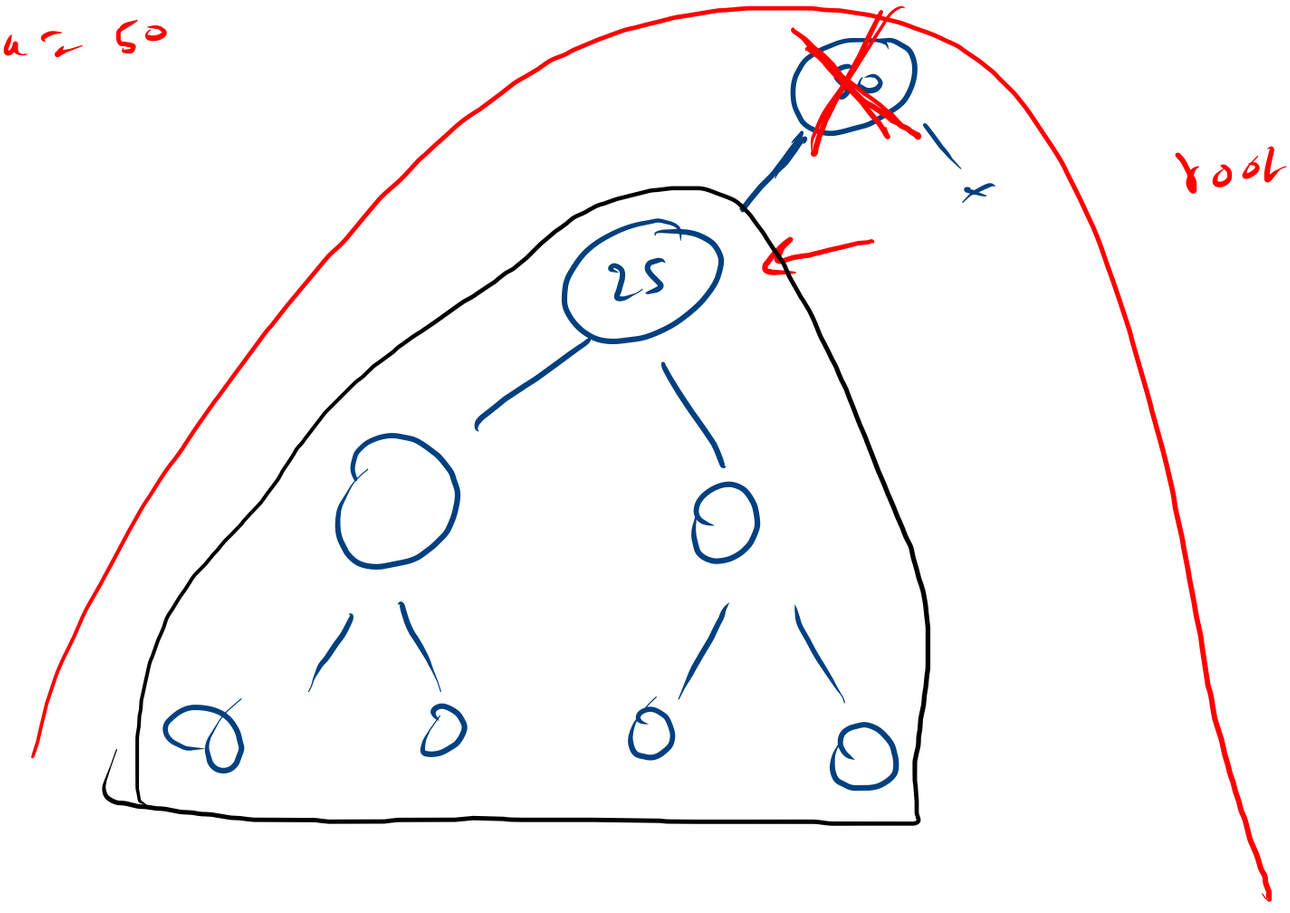


root = null


Single child
data = 50
50
x
75
60
100
root = 75

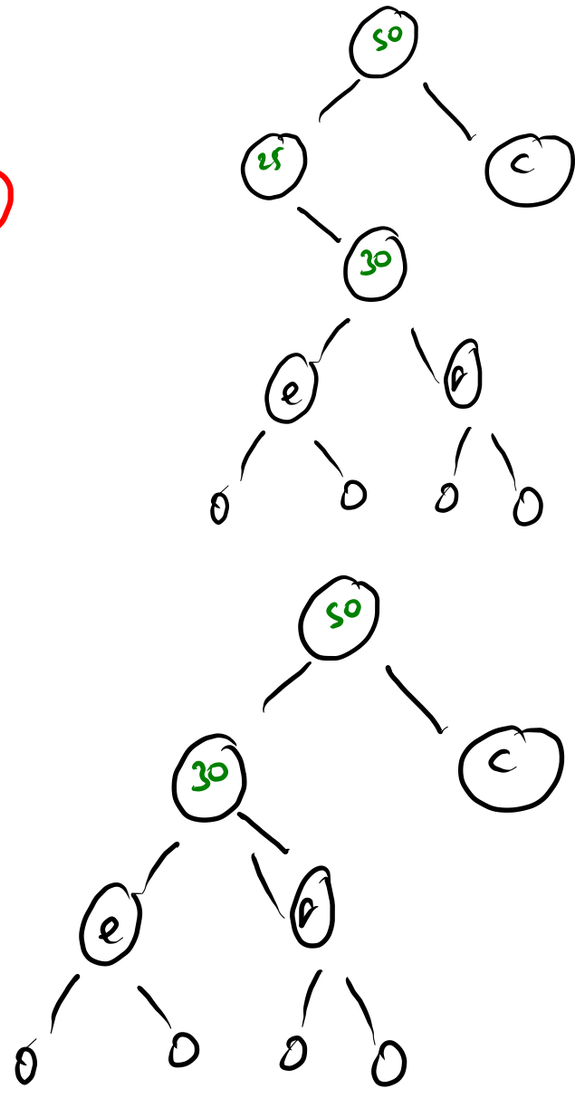

data = 25
50
25
75
30
60
100
0
0
50
30
75
0
0
60
100

data = 50

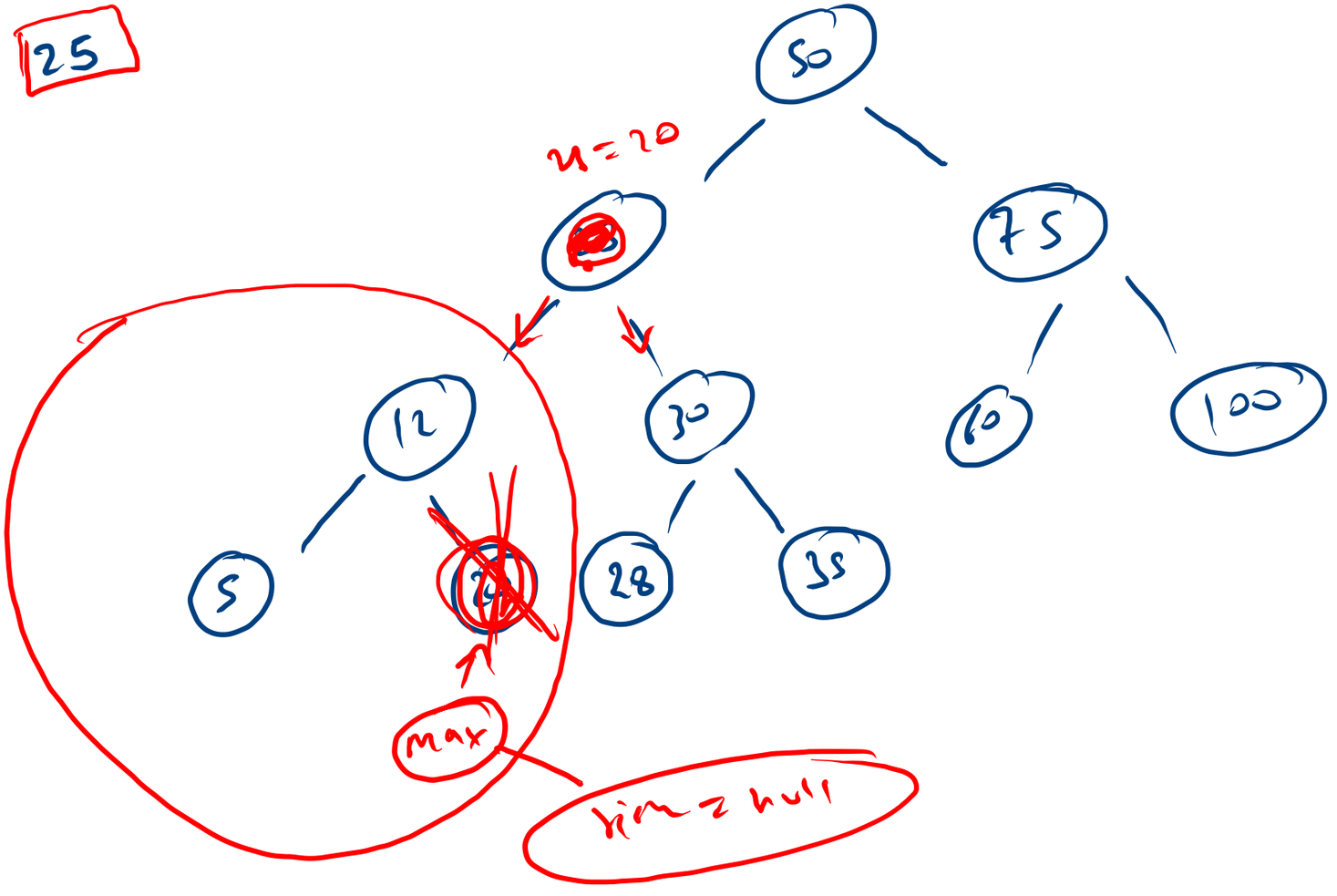


root → 25

data 225



data = 25

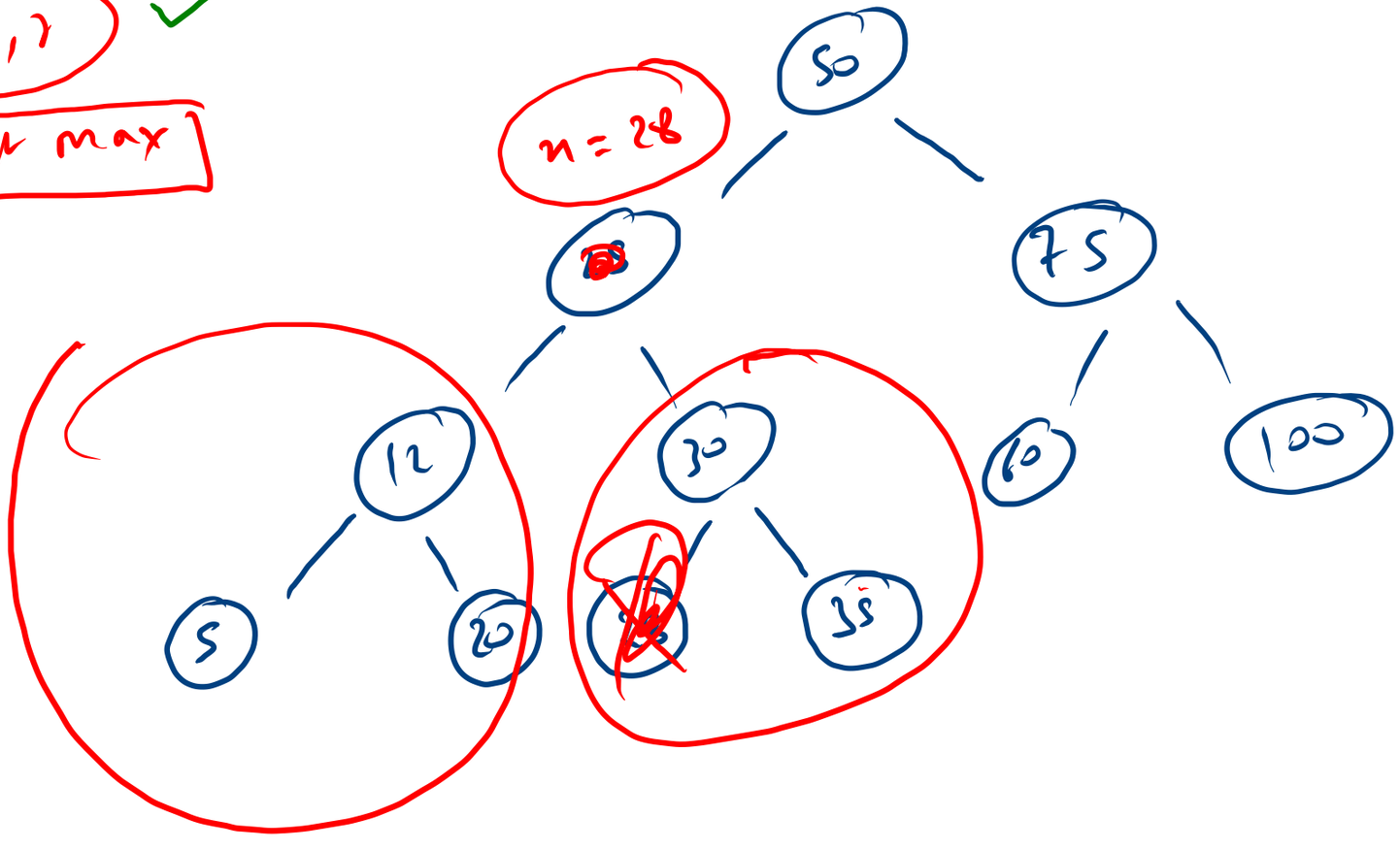

50
x = 20
75
12
30
60
100
5
28
35
max
right child
leaf
sibling child

max

leaf → null ✓

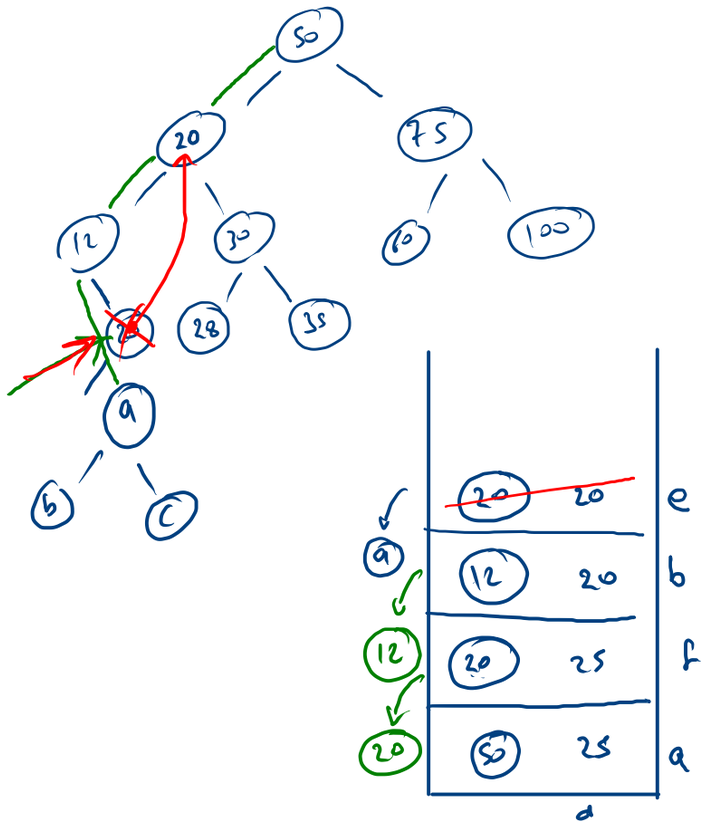
single child → l, r ✓

double → left max



data = 25

```
public static Node remove(Node node, int data) {

    if(data < node.data){
a       node.left = remove(node.left, data);
    }else if(data > node.data){
b       node.right = remove(node.right, data);
    }else{
        if(node.left == null && node.right == null){// left
c           return null;
        }else if(node.left == null && node.right != null){// single right child
d           return node.right;
        }else if(node.left != null && node.right == null){// single left child
e           return node.left;
        }else{

            int max = max(node.left);
            node.data = max;
f           node.left = remove(node.left, max);
        }
    }

    return node;
```



a

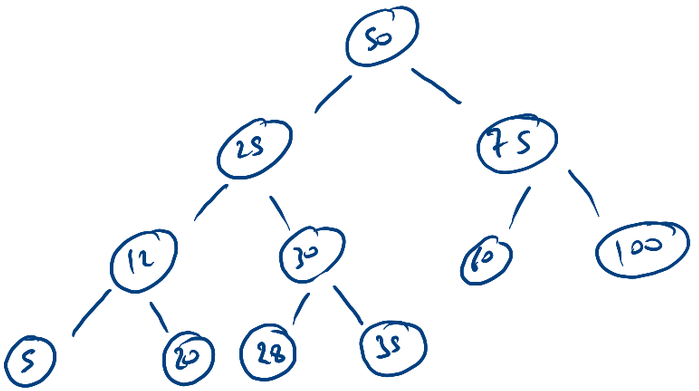
b

c

d

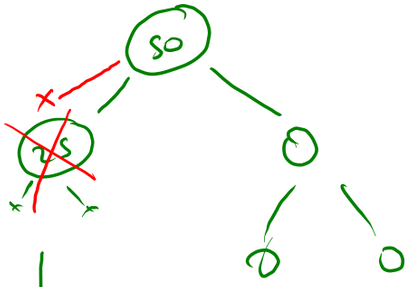
e

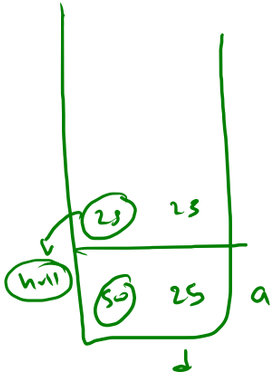
f

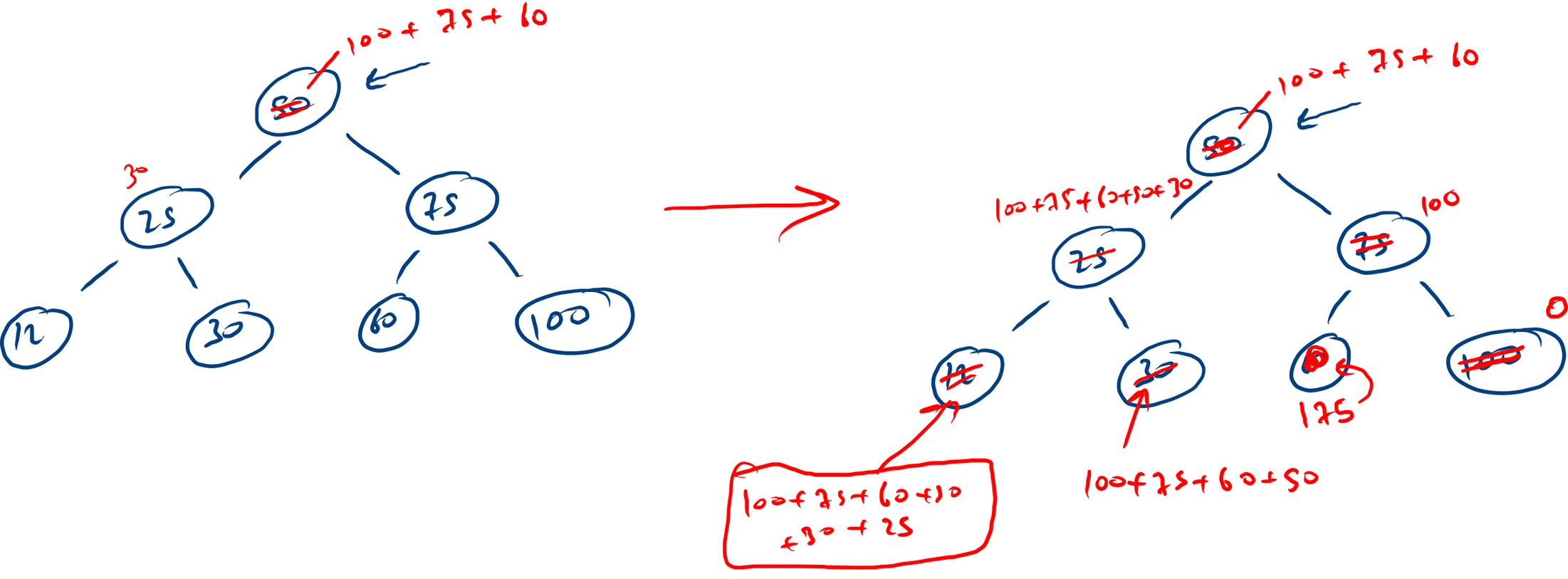


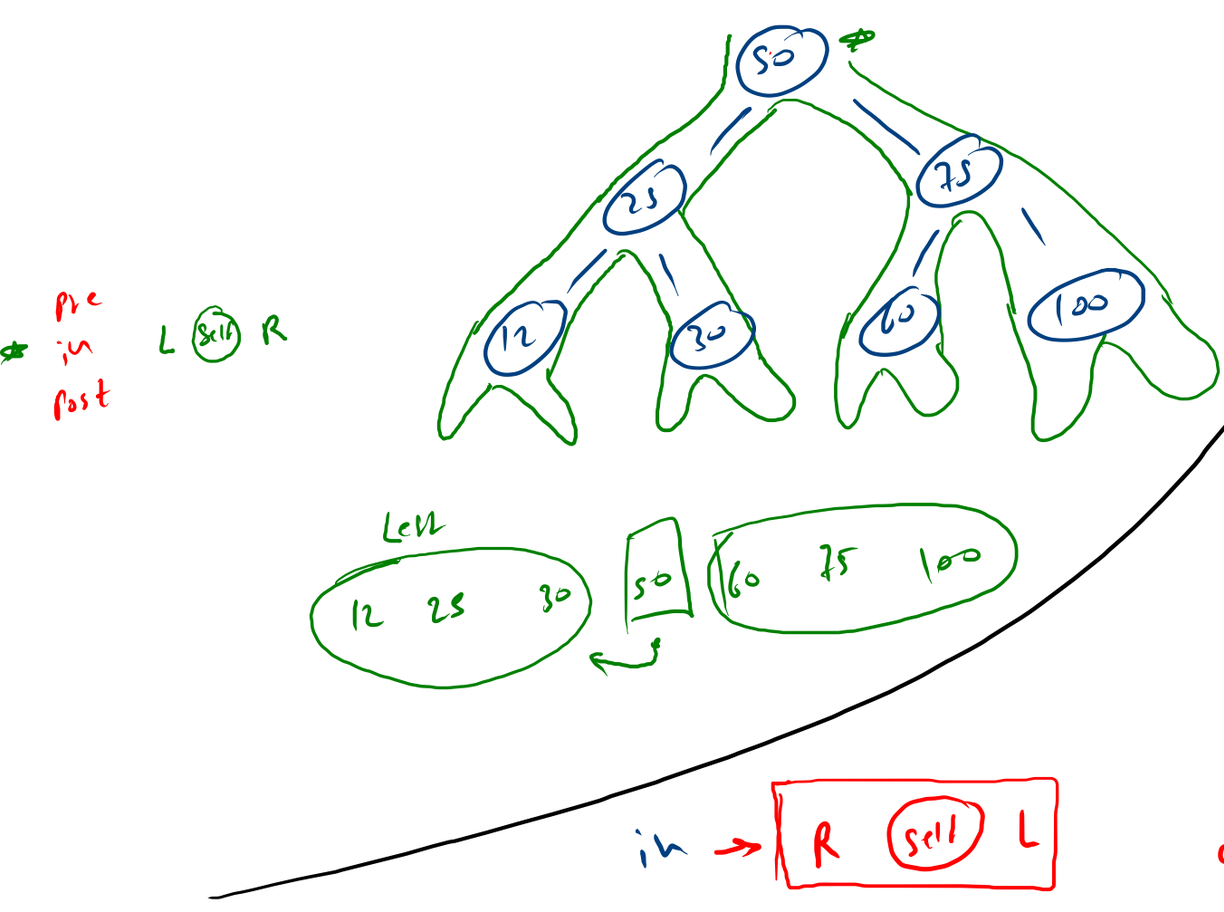

```
public static Node remove(Node node, int data) {

    if(data < node.data){
a       node.left = remove(node.left, data);
    }else if(data > node.data){
b       node.right = remove(node.right, data);
    }else{
        if(node.left == null && node.right == null){// left
c           return null;
        }else if(node.left == null && node.right != null){// single right child
d           return node.right;
        }else if(node.left != null && node.right == null){// single left child
e           return node.left;
        }else{

f           int max = max(node.left);
            node.data = max;
            node.left = remove(node.left, max);
        }
    }

    return node;
```

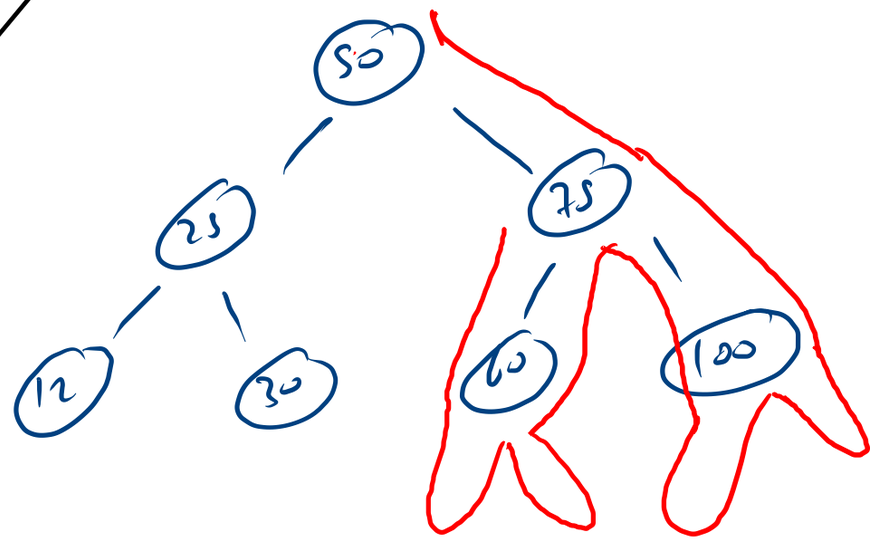
data = 25



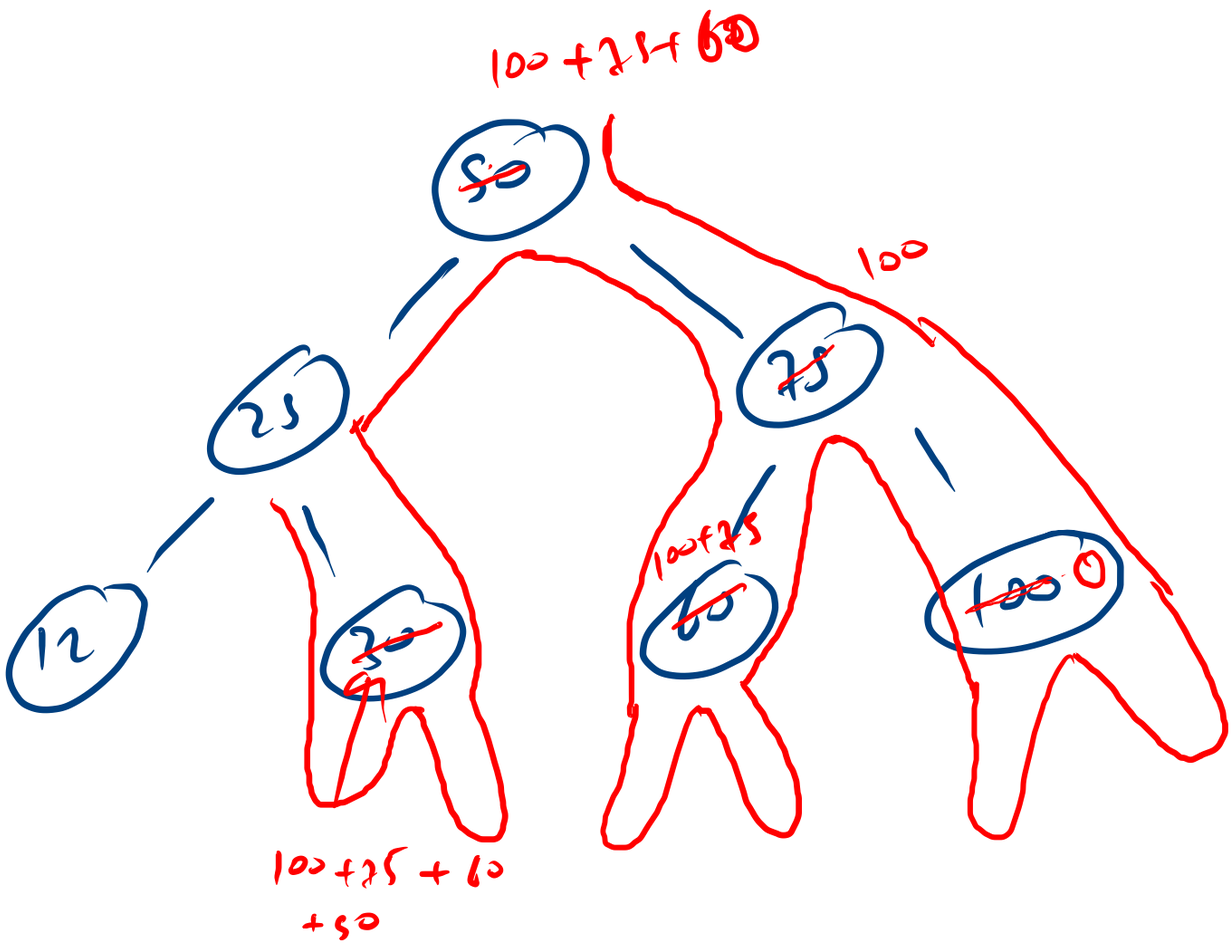



100 + 75 + 60
50
25
75
12
30
60
100
100 + 75 + 60
50
100 + 75 + 60 + 50 + 30
25
75
100
0
12
30
0
100
175
100 + 75 + 60 + 10 + 30 + 25
100 + 75 + 60 + 50


50
25
75
12
30
60
100
pre
in
post
L
self
R
Left
12 25 30
50
60 75 100
in →
R
self
L


50
25
75
12
30
60
100


100
75
60



sum = 0+100+75+60
+50 +30

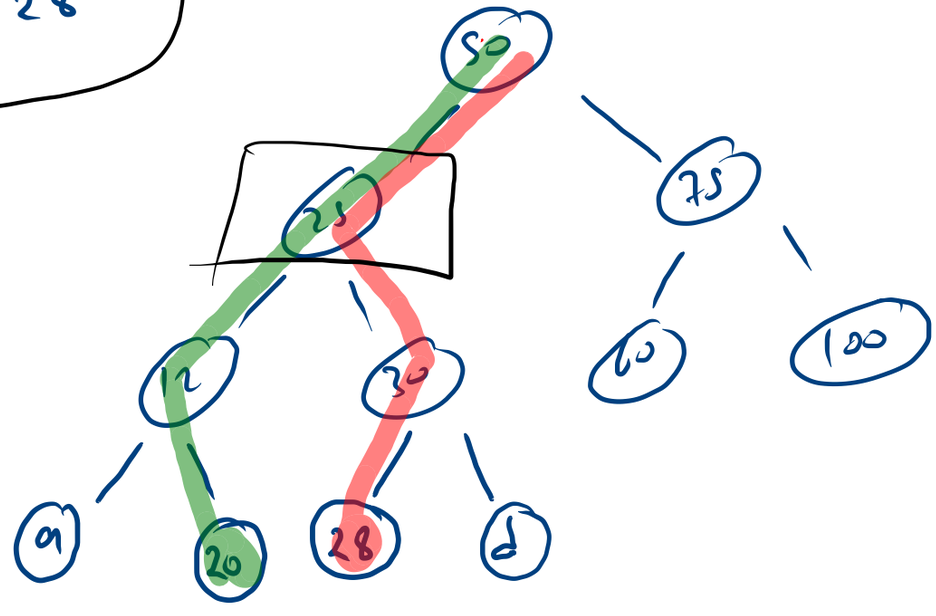
```
int val = hold.data ; //30

hold.data = sum
sum += val;
```

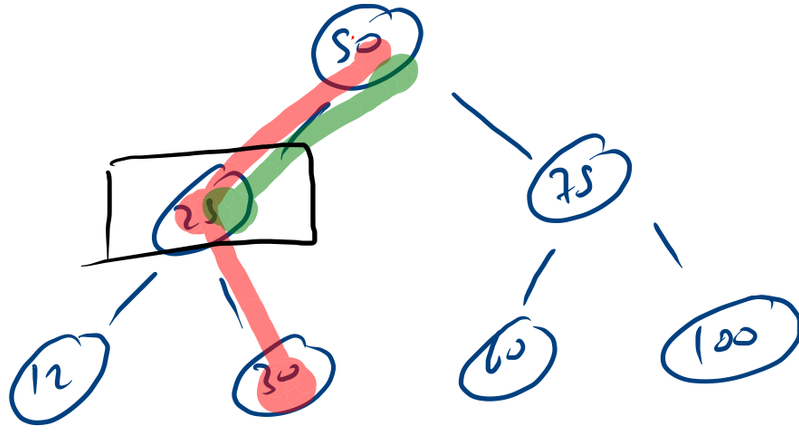
LCA

exit2

d1 = 20

d2 = 28

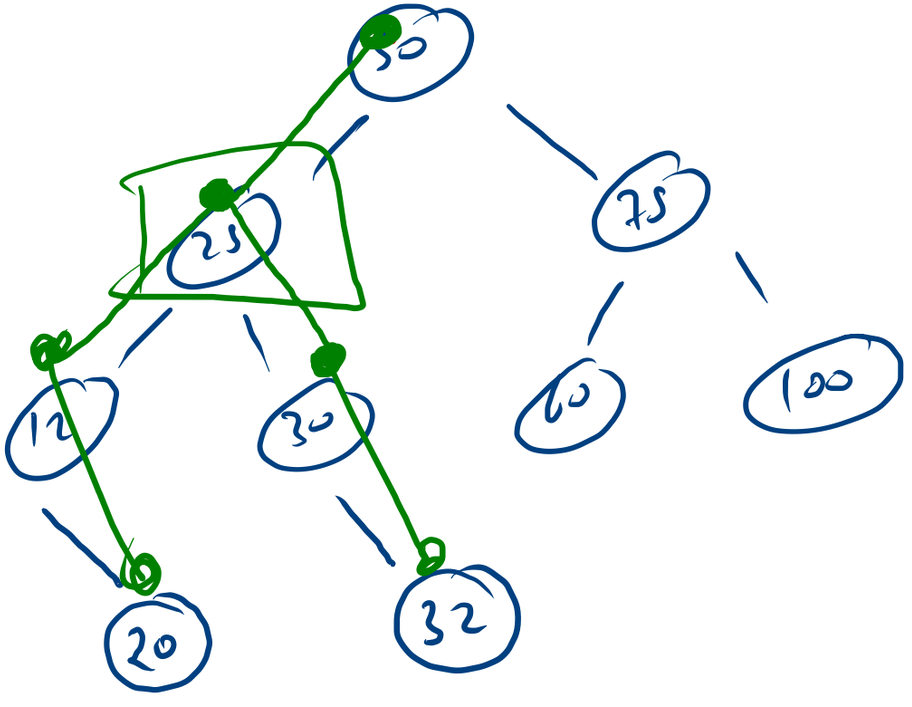




d1 = 25

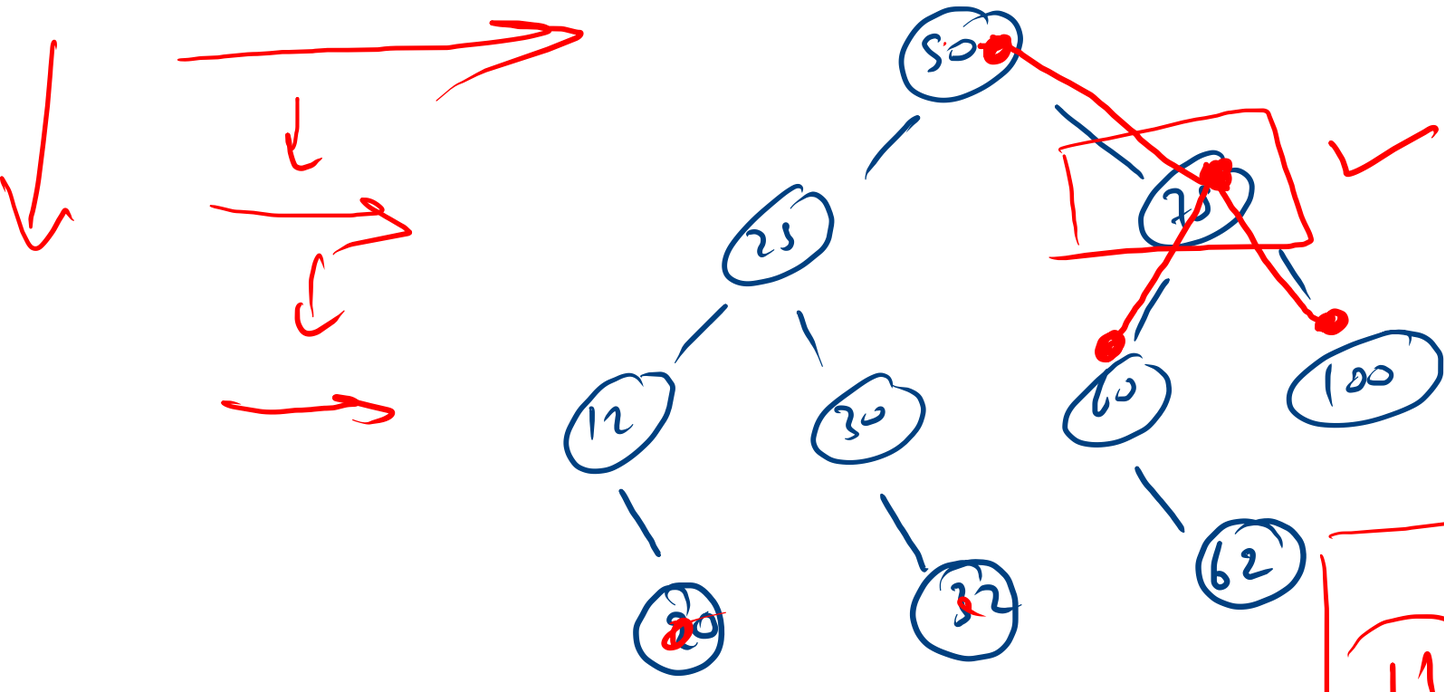
d2 = 30

ans = 25



dar = 20

d2 = 32



62

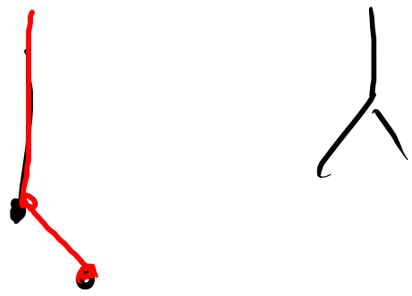
100

d1 → small

d2 → big

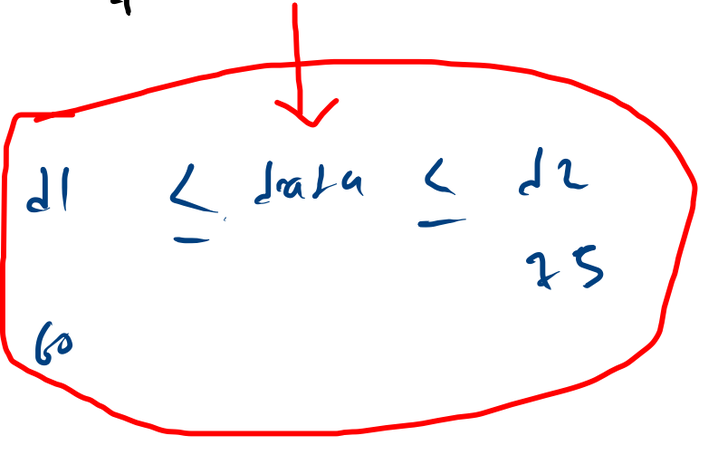
75

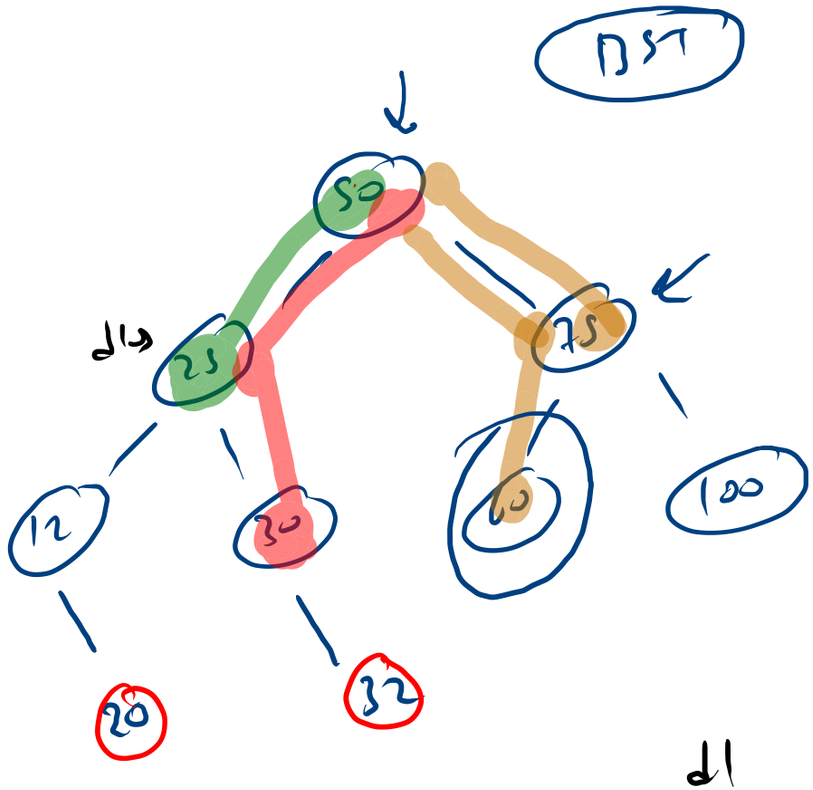
d1 < node.data < d2



60

75



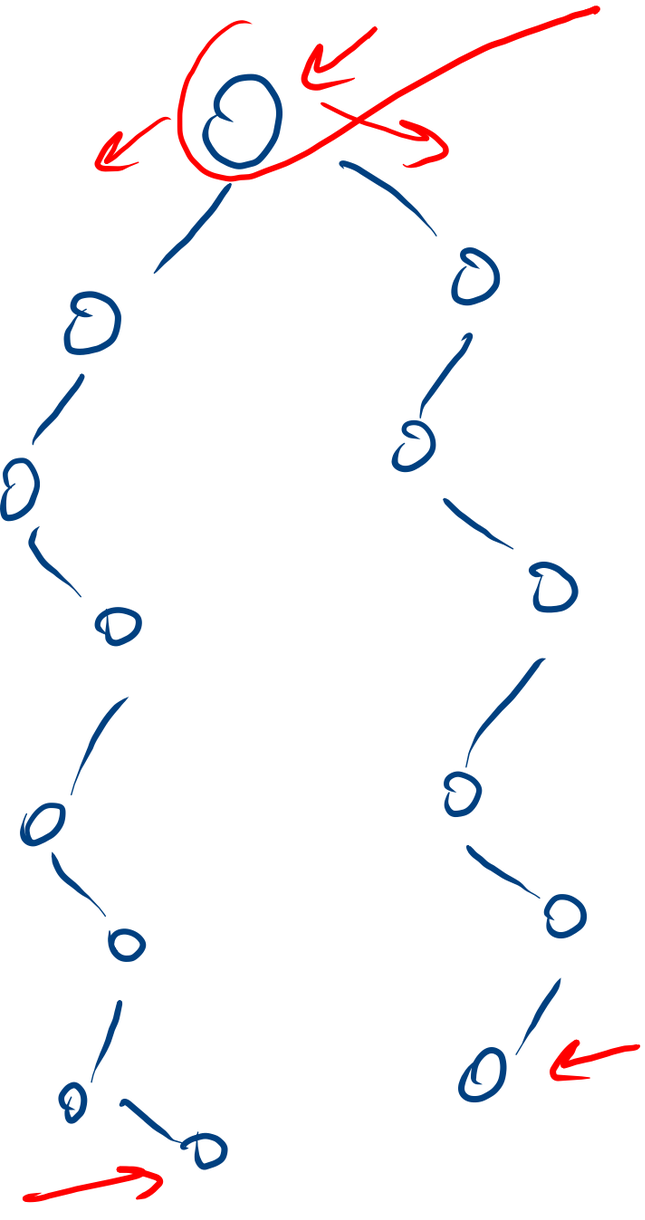


d1 = 20
d2 = 100

d1 → small
d2 → big

d1 ≤ data < d2

B.T.



D.ST