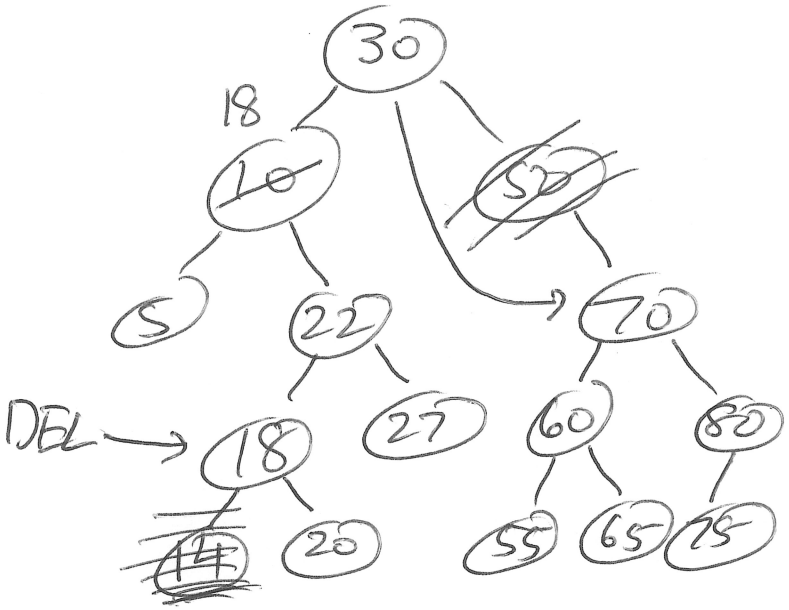
COP 3502 2/29/27

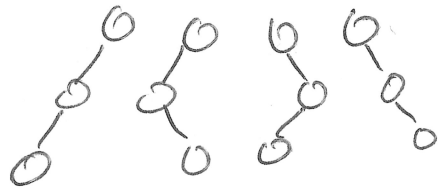
Binary Search Tree Delete

BST Problems



Delete 14 (leaf node)

- easy case
  set appropriate ptr (par→left or par→right) NULL, free memory return value.
- If tree 1 → 0 special case ptr root => NULL



2 child case

Delete 50 (one child)

patch appropriate parent ptr appropriate child ptr free memory, return value.

- 4 cases

Step 1: Copy either max left or min on right into the node storing the value to be deleted.

Step 2: ID node that used to have that value + physically delete it. This node has ≤ 1 child OLD code can handle it!
min on right has no left child
max of left has no right child.

Delete 10 (two child)

- Can't patch parent to both children!

Storing Extra Data in a Node

default BST

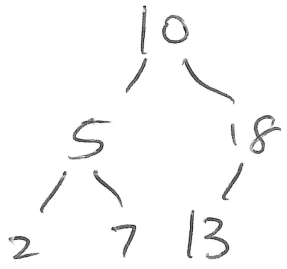
data
left
right

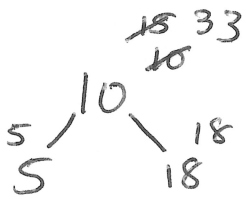
More Stuff

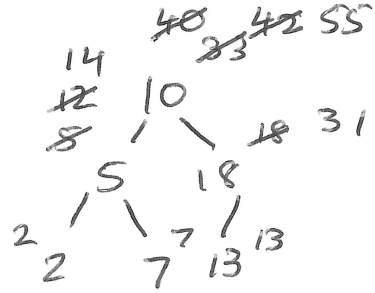
data
sum // sum of all values in subtree
height // the height of this subtree
left
right

What changes?

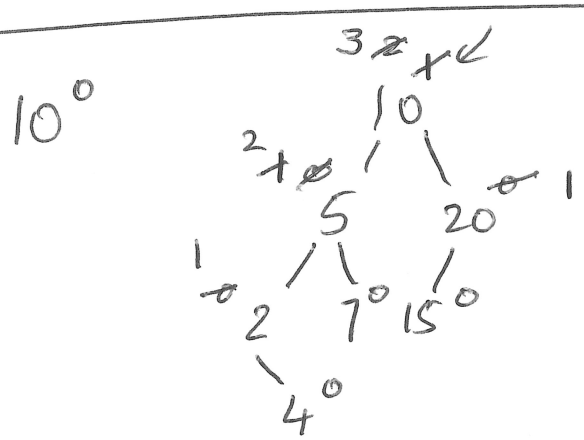
In implementation you **have to** update all the extra data as you insert/delete!



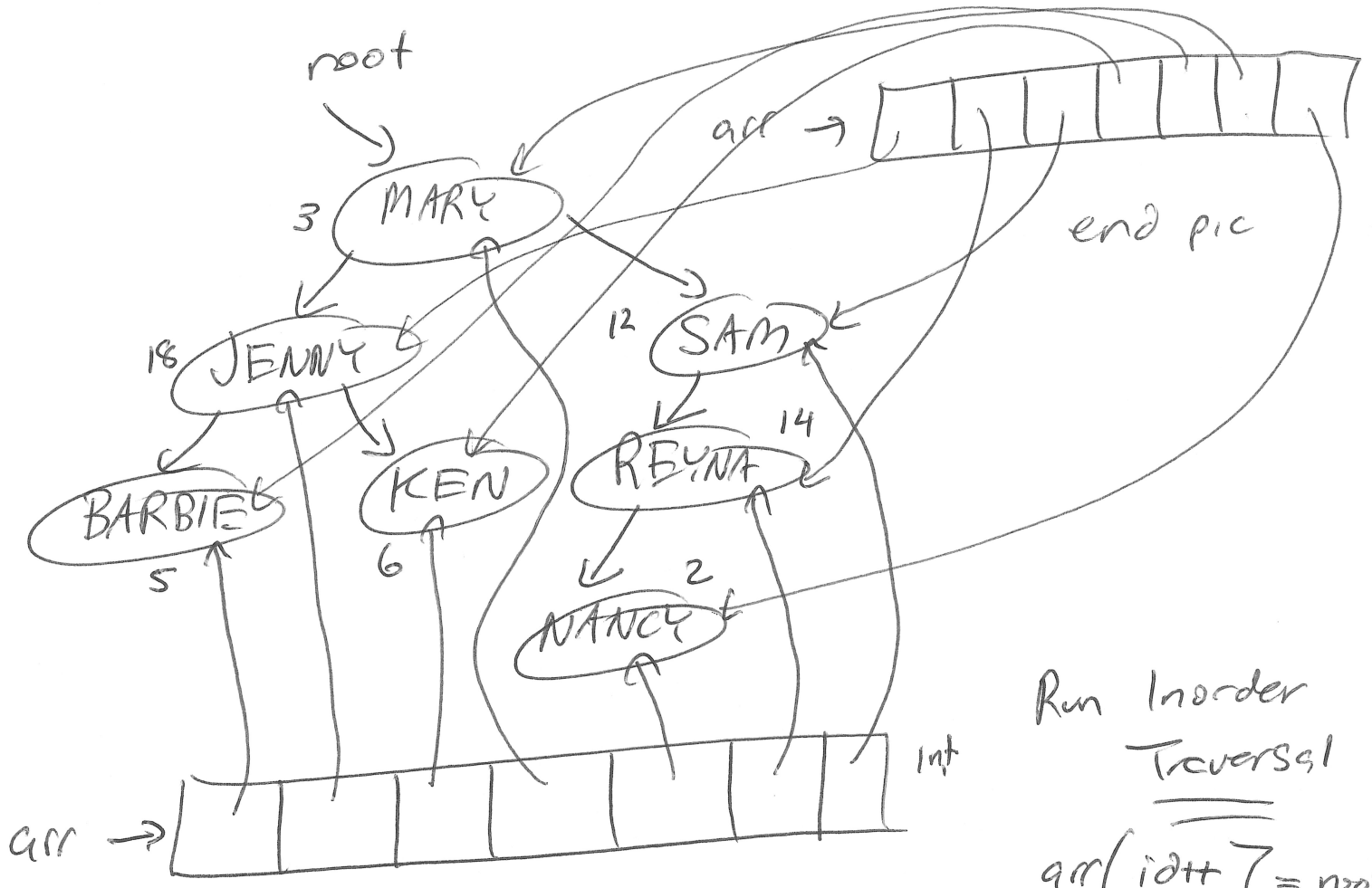




Maintaing Sum



Maintain height



FND Q1 Notes

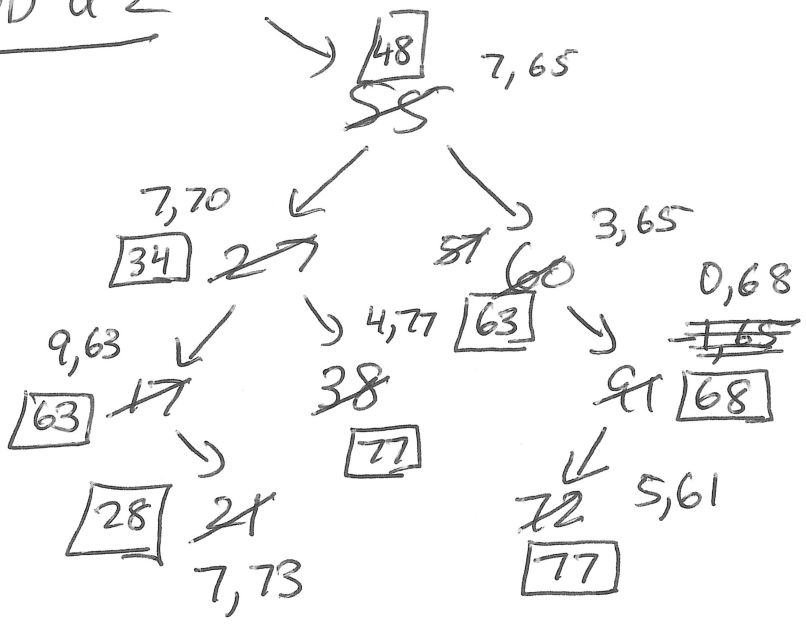
0 mod 3	print data go left
1 mod 3	print data+2 go right
2 mod 3	do L, R no print

10 16 30 70 90 85

↓ ↓ ↓ ↓ ↓ ↓

12, 18, 30 72, 90, 87

FND Q2



FND Q3

```
int check(treenode * A, treenode * B) {
    if (A == NULL && B == NULL) return 1;
    if (A == NULL || B == NULL) return 0;
    if (A->data != B->data) return 0;
    return check(A->left, B->left) && check(A->right, B->right);
}
```