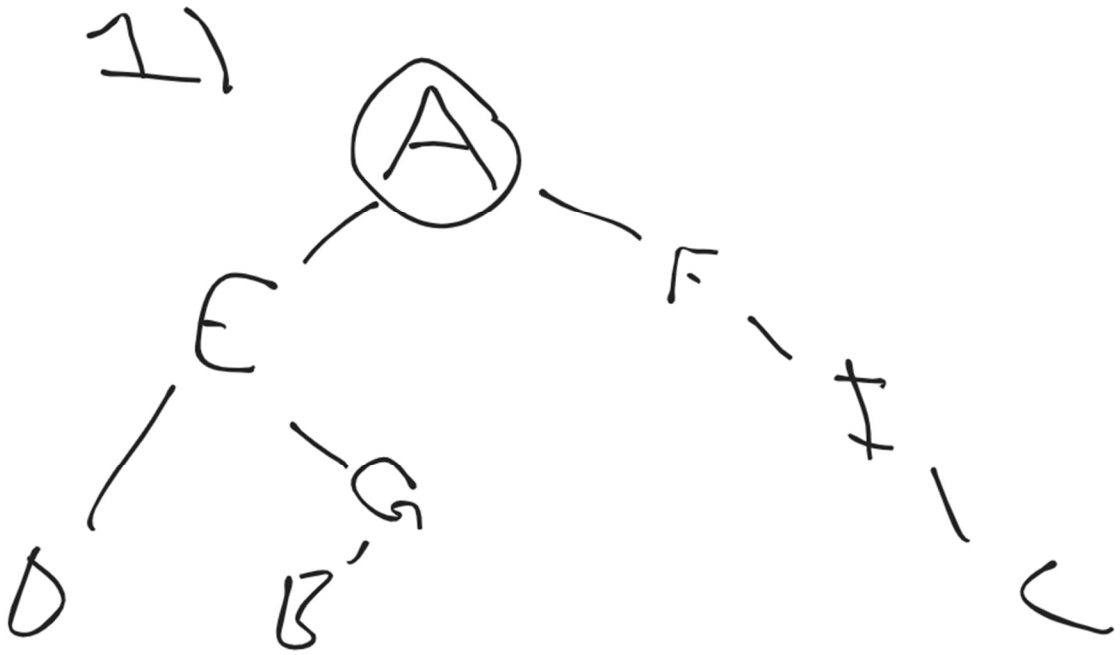
Chase Conaway

10/20/2020

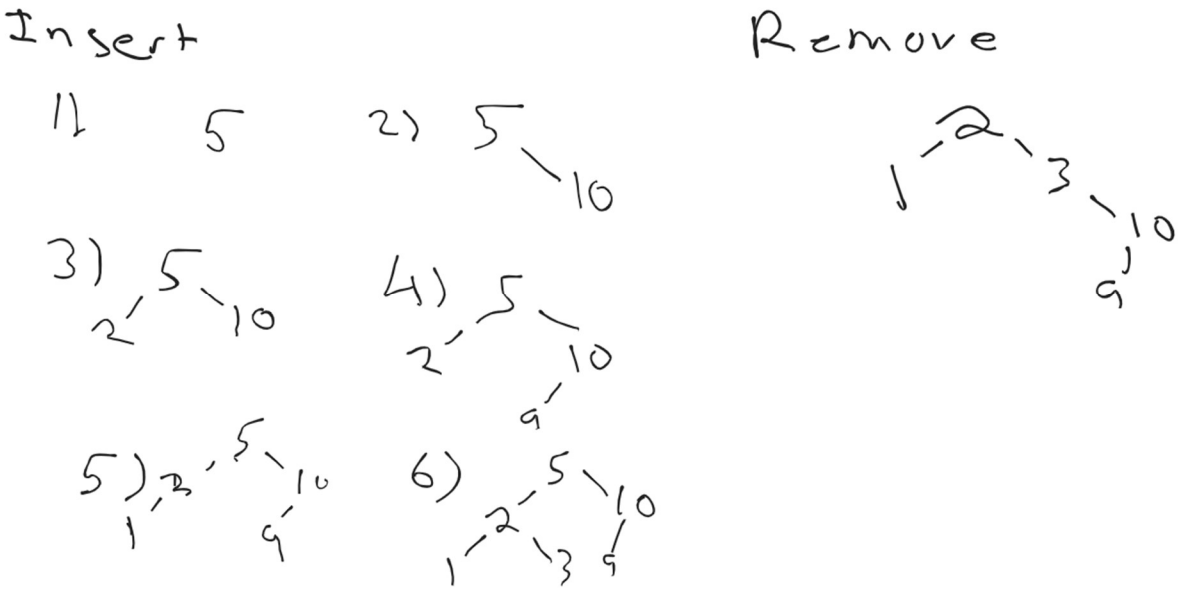
Nadra Guizani

Homework 2

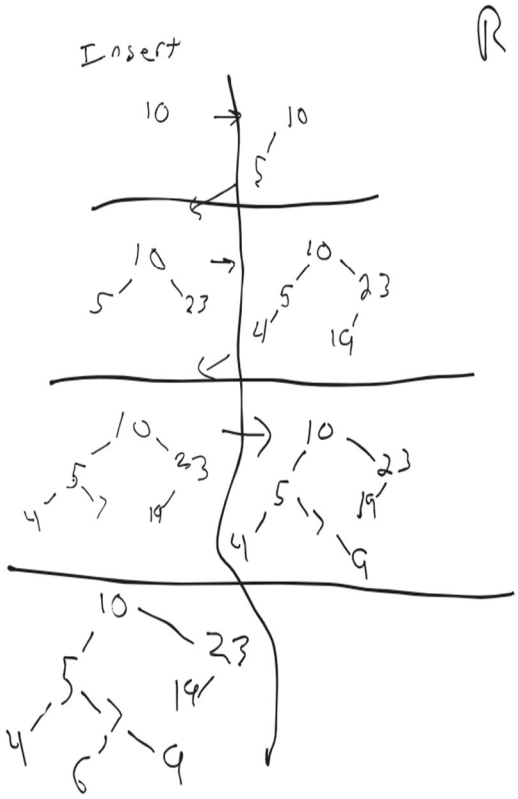
1.

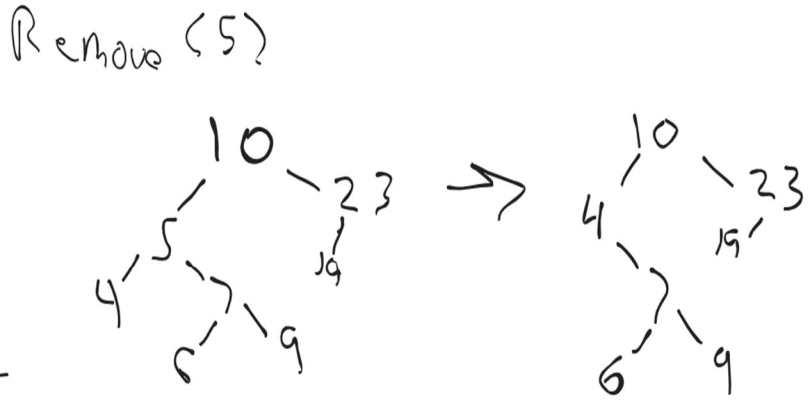


2.

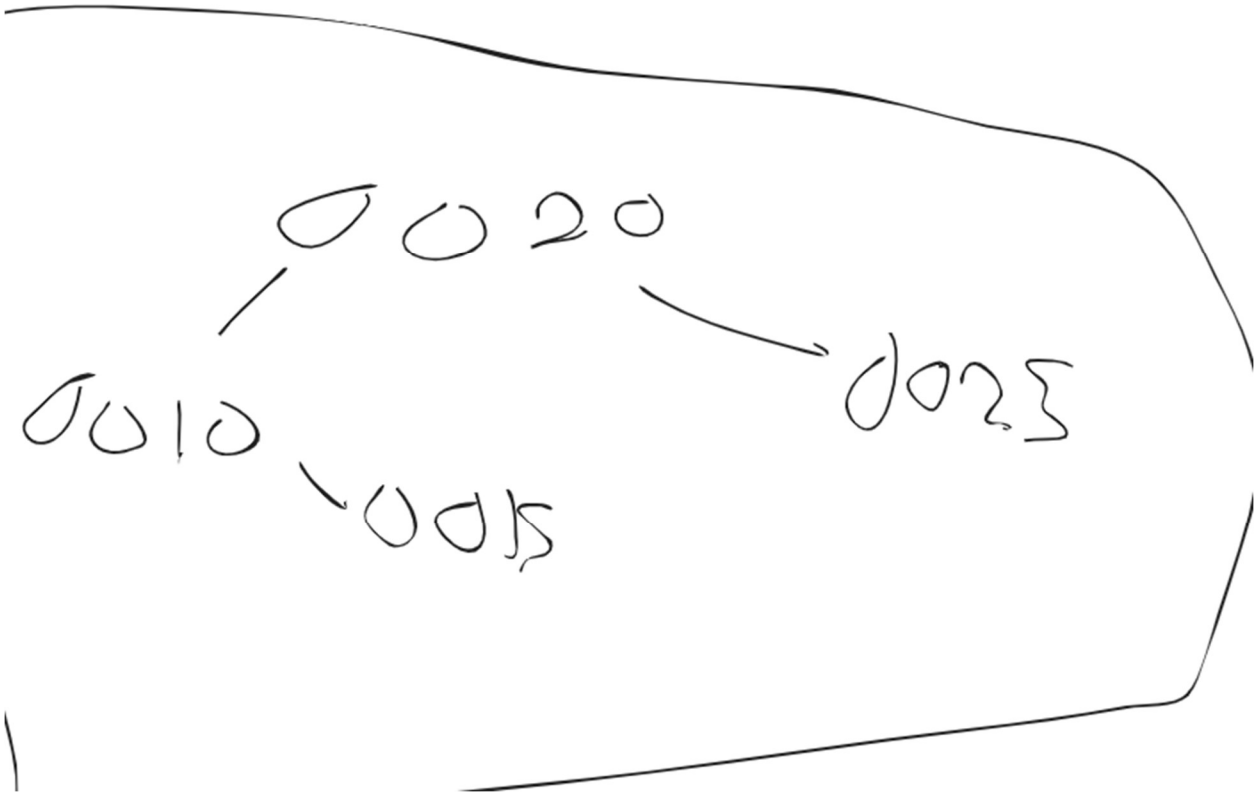

Insert
1) 5
2) 5 10
3) 5 2 10
4) 5 2 10 9
5) 2 5 10 1 9
6) 5 2 10 1 3 9
Remove
2 1 3 10 9

3.


Insert
10
10
5
10
5
23
10
5
23
4
19
10
23
5
19
4
9
10
23
5
19
4
6
9

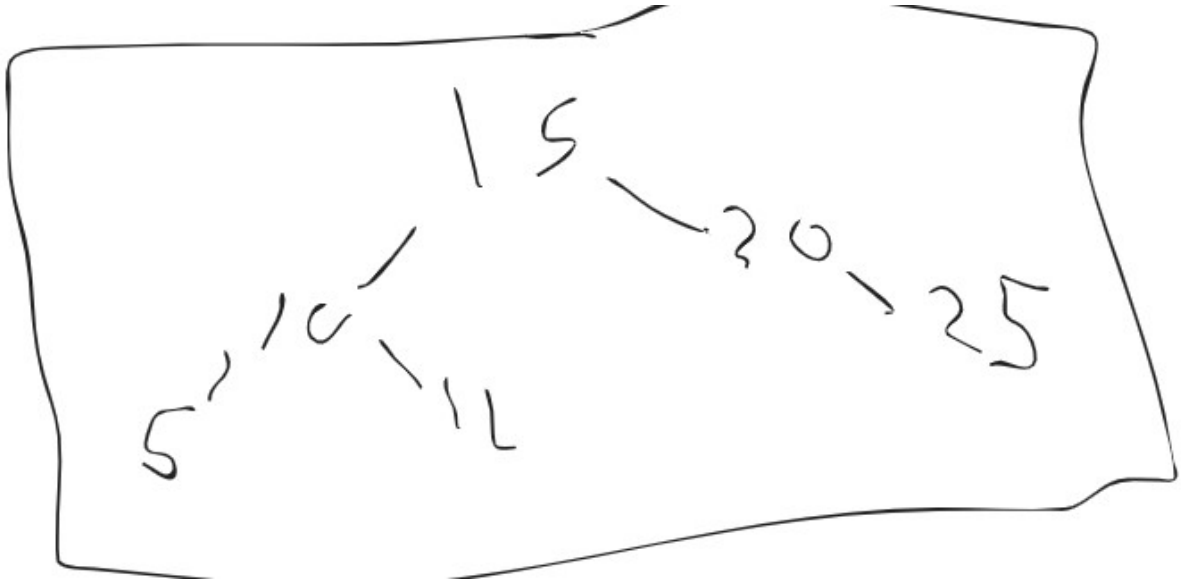

Remove (5)
10
23
5
19
4
6
9
10
23
4
19
6
9

4. Given the following binary tree (where nullptr height == -1):
    a. The tree has a height of 4.
    b. The depth of 90 is 3.
    c. The height of node 90 is 1.
    d. Traversals
        i. Preorder: 0100, 0050, 0003, 0001, 0020, 0080, 0052, 0090, 0083, 0099, 0150, 0125, 0152
        ii. Inorder: 0001, 0003, 0020, 0050, 0052, 0080, 0083, 0090, 0099, 0100, 0125, 0150, 0152
        iii. Post-order: 0001, 0020, 0003, 0052, 0083, 0099, 0090, 0080, 0050, 0125, 0152, 0150, 0100
5.



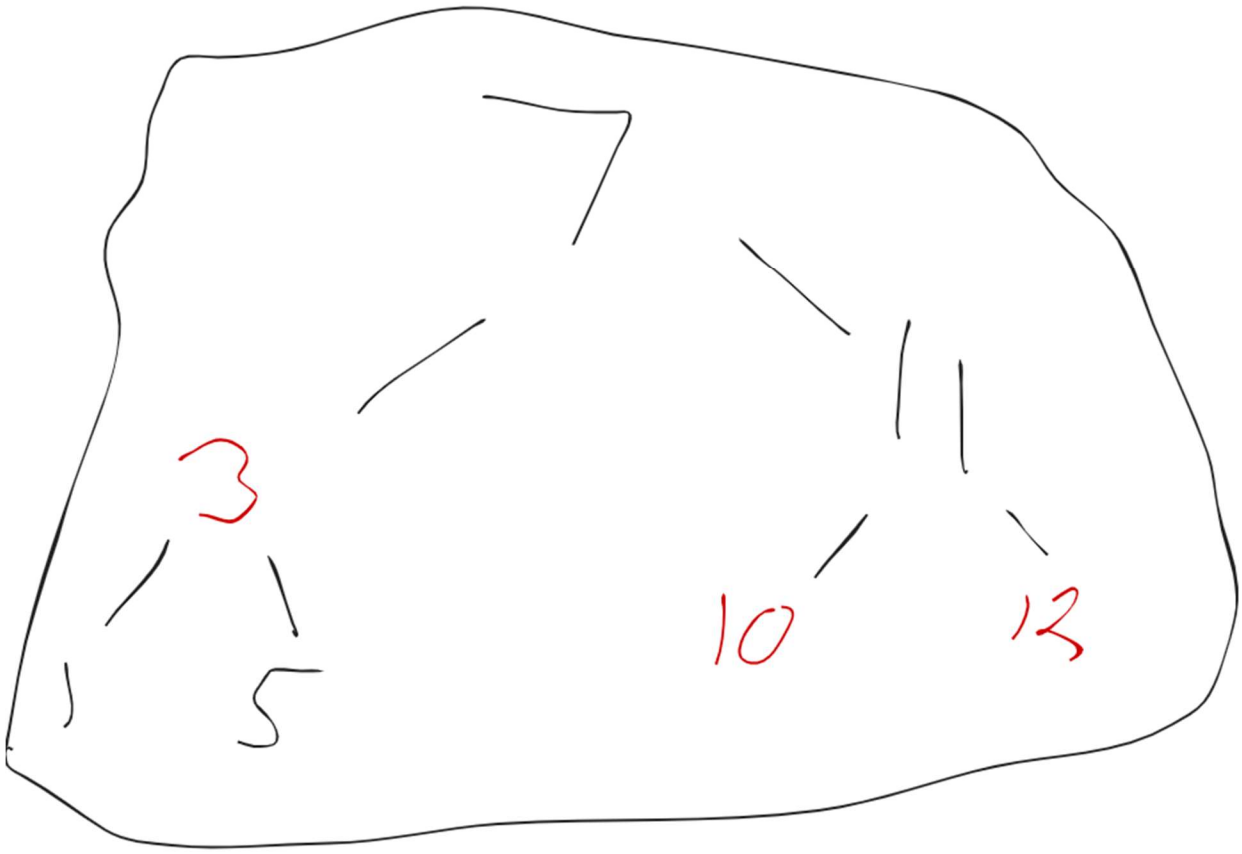
6.

7.



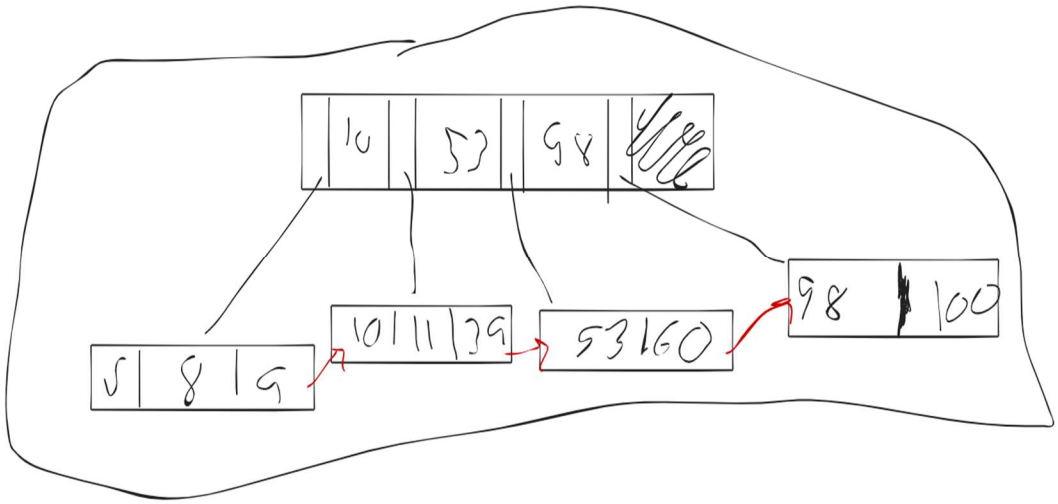
8.

9.

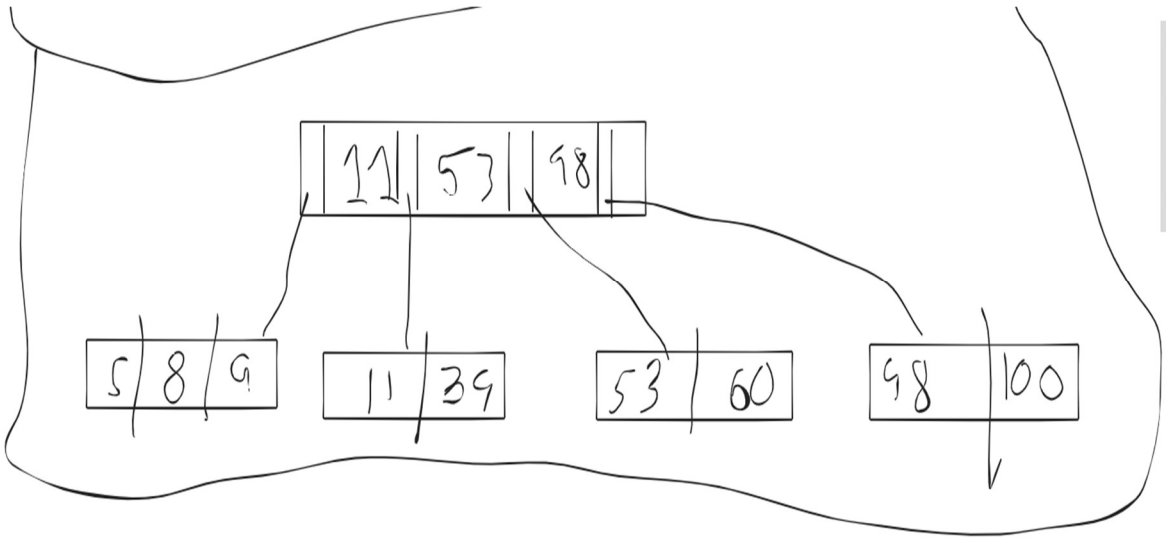


10.

a.



b.



11. We are going to design our B+ Tree to be as optimal as possible for our old hard drives (since the management won't buy new ones, those cheapskates!). We want to keep the tree as short as we can and pack each disk block in the filesystem as tightly as possible. We also want to access our data in sorted order for printing out reports, so each leaf node will have a pointer to the next one. See figure #1 on next page for a visualization of our tree.
    a. The size of the internal node in bytes is 104. In each node we have 4 keys, each with 128 bits of data and 5 pointers with 64 bits each, so the total number of bits is 832 bits, or 104 bytes.
    b. When we calculate the size of the leaf node, we must also account for the pointer to the next leaf node, so, just another 64 bits, which is 896 bits, or 112 bytes.
    c. The B+ tree will end up being about $1 + \log_{M/2}(\frac{N+1}{2})$ levels high on average.
    d. The tree should have a height of 12.
    e. The tree with 2.5 million customers will have a height of 17.