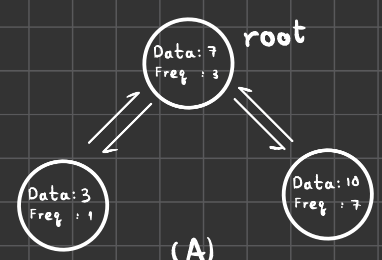
2018.

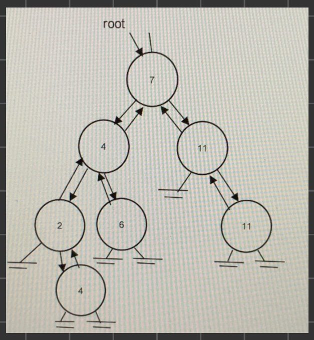
6) Implementation of BST with duplicate data

Solution : Make the nodes store the frequencies of the data

Ex.



(A)
sol^n



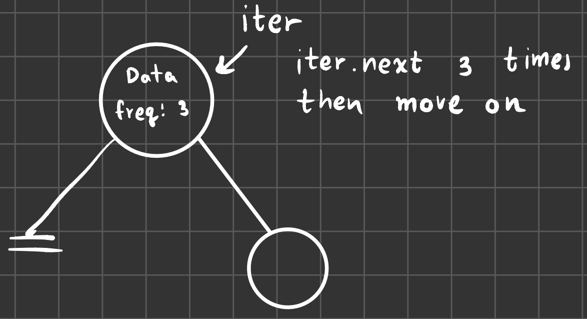
(B)
Given

1.) Iteration

A.) While an iterator is on a node it must iterate through the number of frequencies

Pseudo code:

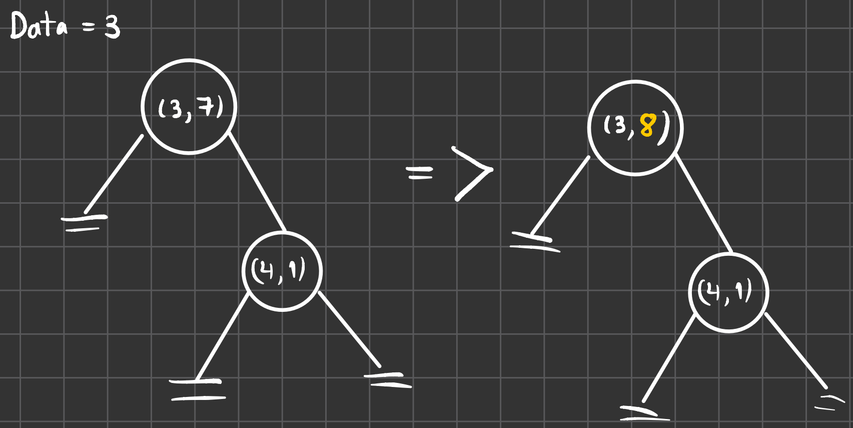
```
BST Node ;   int data  = ...
             int freq  = ....

Tree Iterator;  next( ) {
                   while (freq ≠0) { freq --}
                   if (freq = 0) {
                   // Implement same as non-duplicate next()
                   }
                }

BST ;   Tree Iterator  t  = new  Tree Iterator (root);
        print (root.next().data)
      * output : 7
```



B.) Same as non-duplicate but will need to check for duplicate nodes so a second iterator might be needed to implement iteration

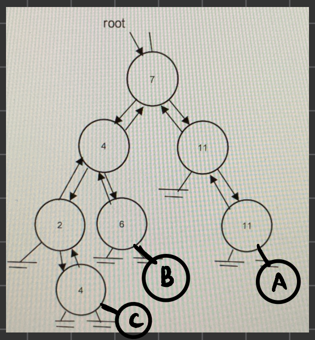
2.) Insert

A.) Find the data first if the index is returned iterate to it and add 1 frequency else do regular insert



B.) Do regular insert but connections can be ambiguous. It depends on implementation.

Insert 11
can be at
A, B or C



3.) Removing

A.) Better for duplicate cases just decrement the frequency
Else do regular removing

B.) Slower since no special cases. The process is done by modifying pointers