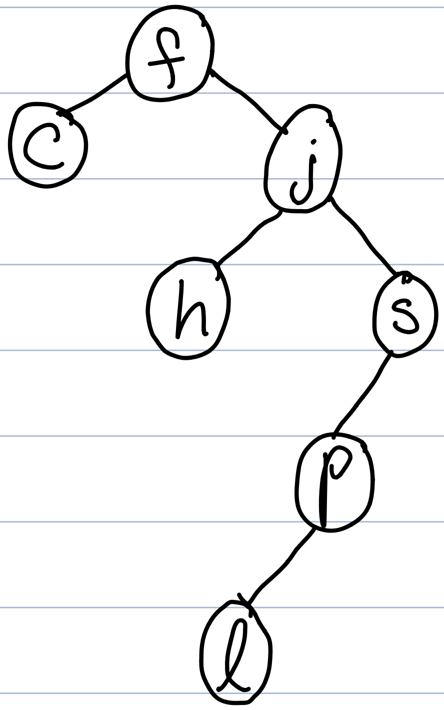
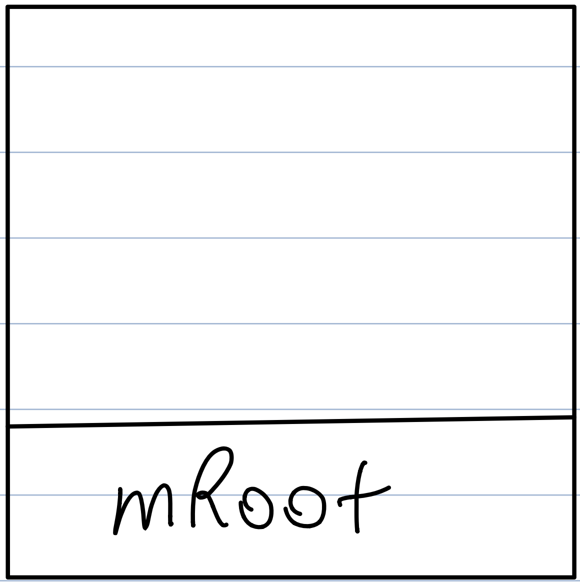
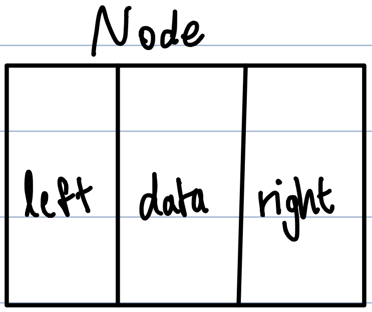
f, j, h, s, p, c, l







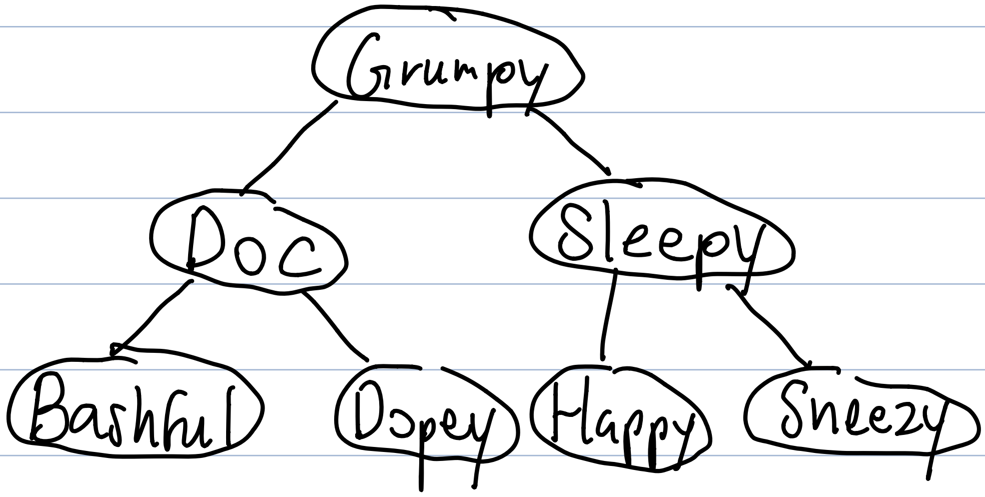
BST

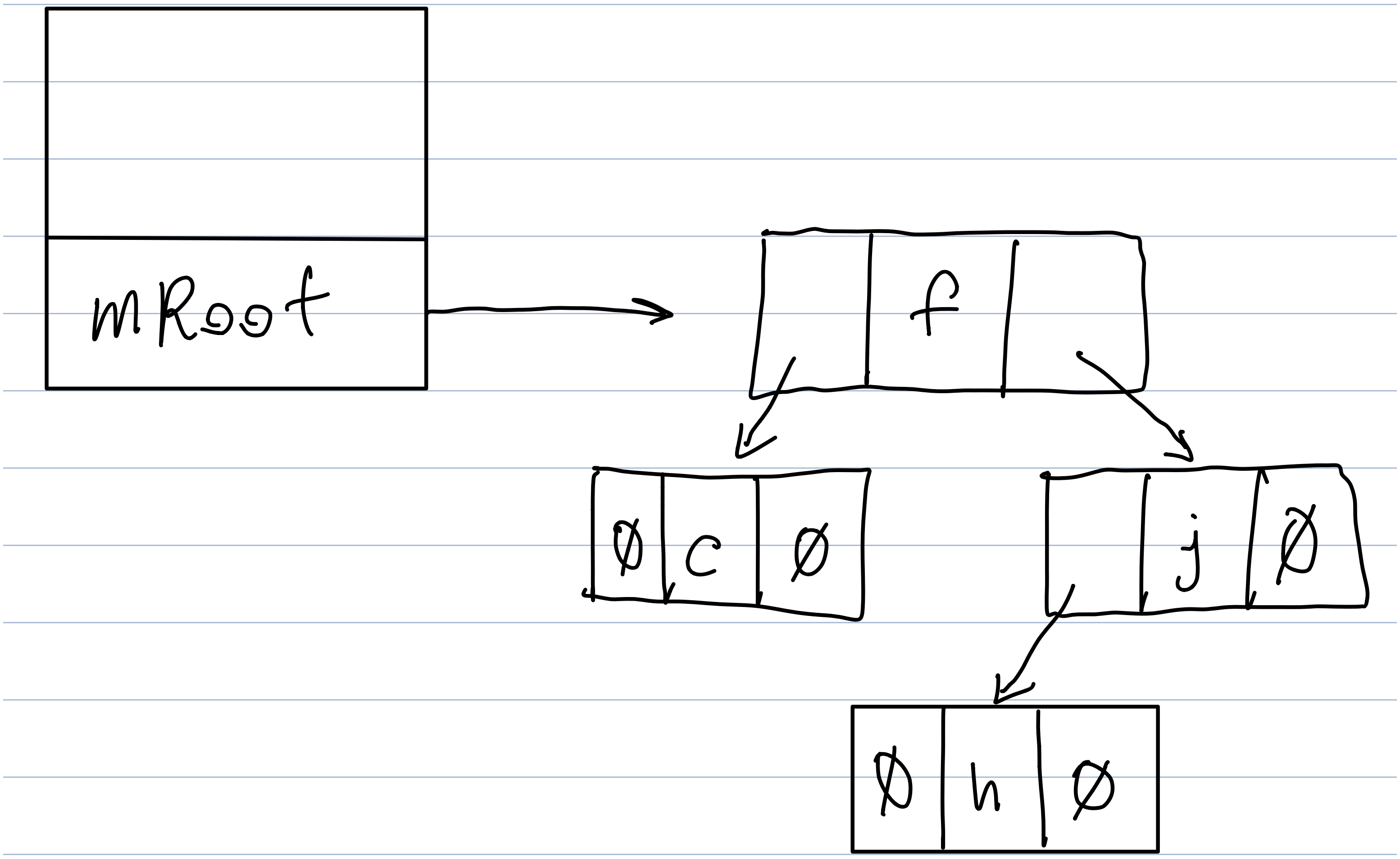
```
insert(String key)
{
  if (mRoot == null)
  {
```

Node

```
insert(key)
{
if(key.compareTo(mData)<0)
{
    mRoot = new Node(key);
  }
  else
  {
    mRoot.insert(key);
  }
}
```

```
      if (mleft == null)
      {
        mLeft = new Node(key);
      }
      else
      {
        mLeft.Insert(key);
      }
    }
    else
    {
      if (mRight == null)
      {
        mRight = new Node(key);
      }
      else
      {
        mRight.insert(key);
      }
    }
  }
```

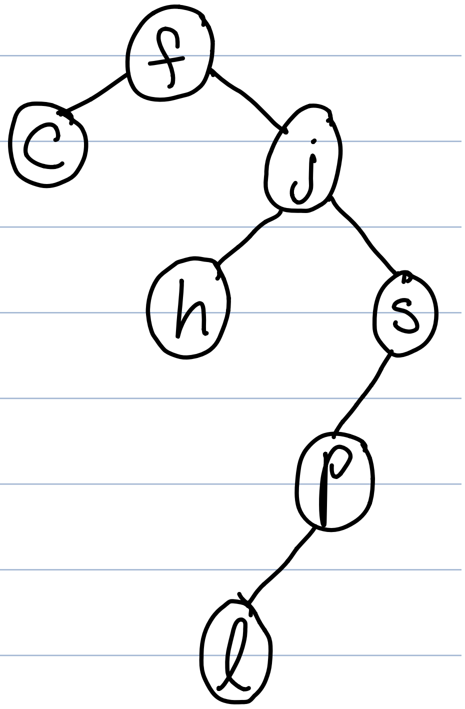




BST

```
traverse (TraversalType order)
{
   System.out.println( order + " tree traversal");
   if (mRoot == null) return;
   mRoot.traverse(order);
```

}



PREORDER (visit, left, right)
f, c, j, h, s, p, l

INORDER (left, visit, right)
c, f, h, j, l, p, S

POSTORDER (left, right, visit)
c, h, l, p, s, j, f

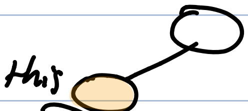
getCount (AUDIO)

getCount → POSTORDER,

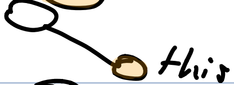
REMOVE

① Leaf + Root

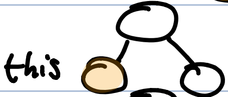
② Leaf + no sibling + left



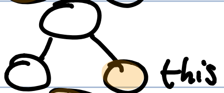
③ Leaf + no sibling + right



④ Leaf + Left + Sibling

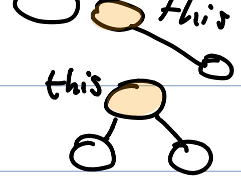


⑤ Leaf + Right + Sibling



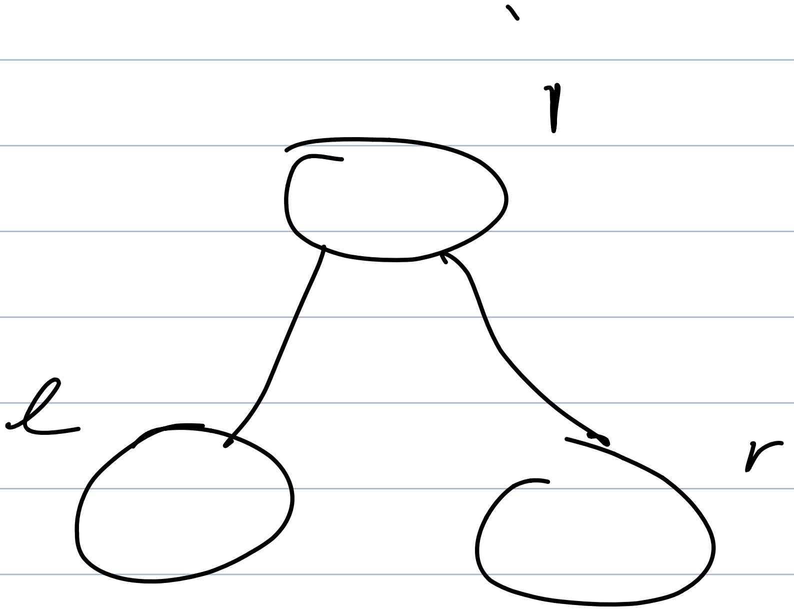
⑥ Int. Node (I.N.) Root + Left Only

7. I.N. Root + Right Only
8. I.N. Root, Left + Right



9. Rpt. 6-8 except w/ non-root + left
10. Rpt. 6-8 except w/ non-root + right

NO O-NOTATION ON FINAL!



1 + l.count() + r.count()