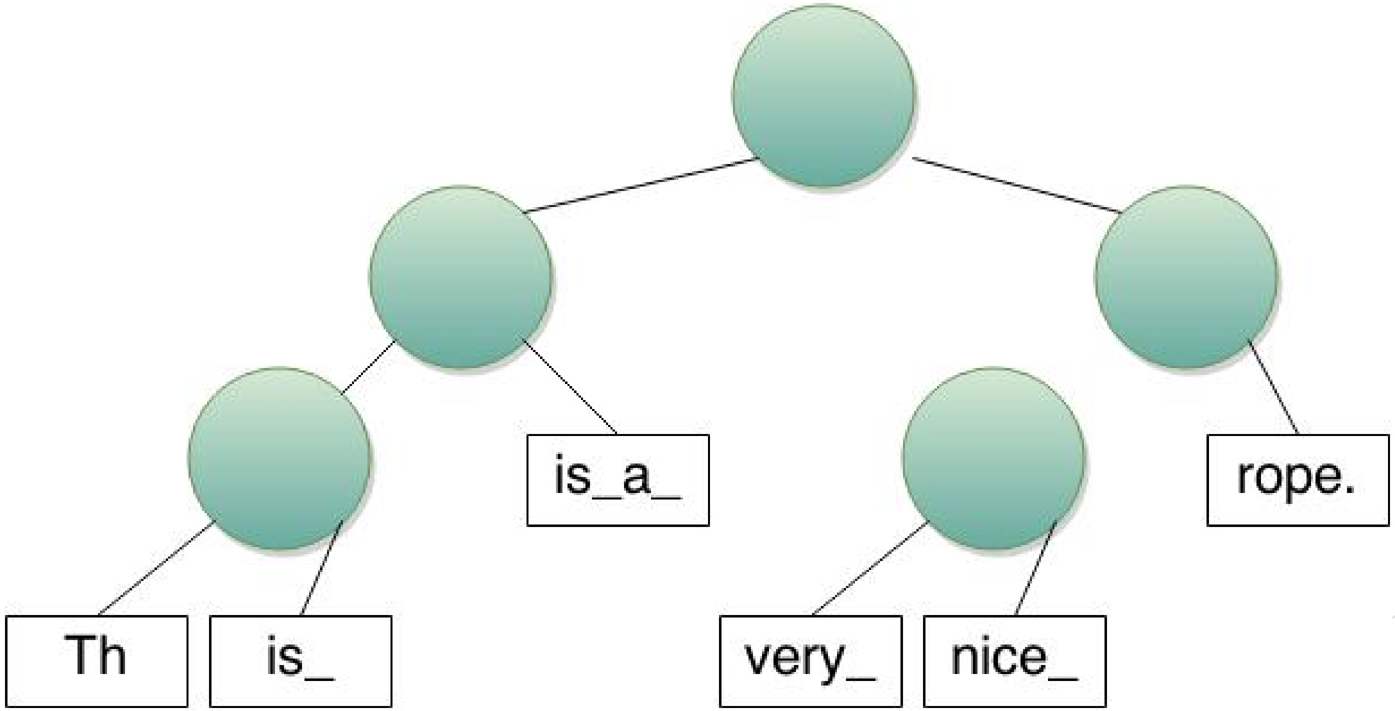
Rope - Data Structure Exercises

Nicholas Elliot

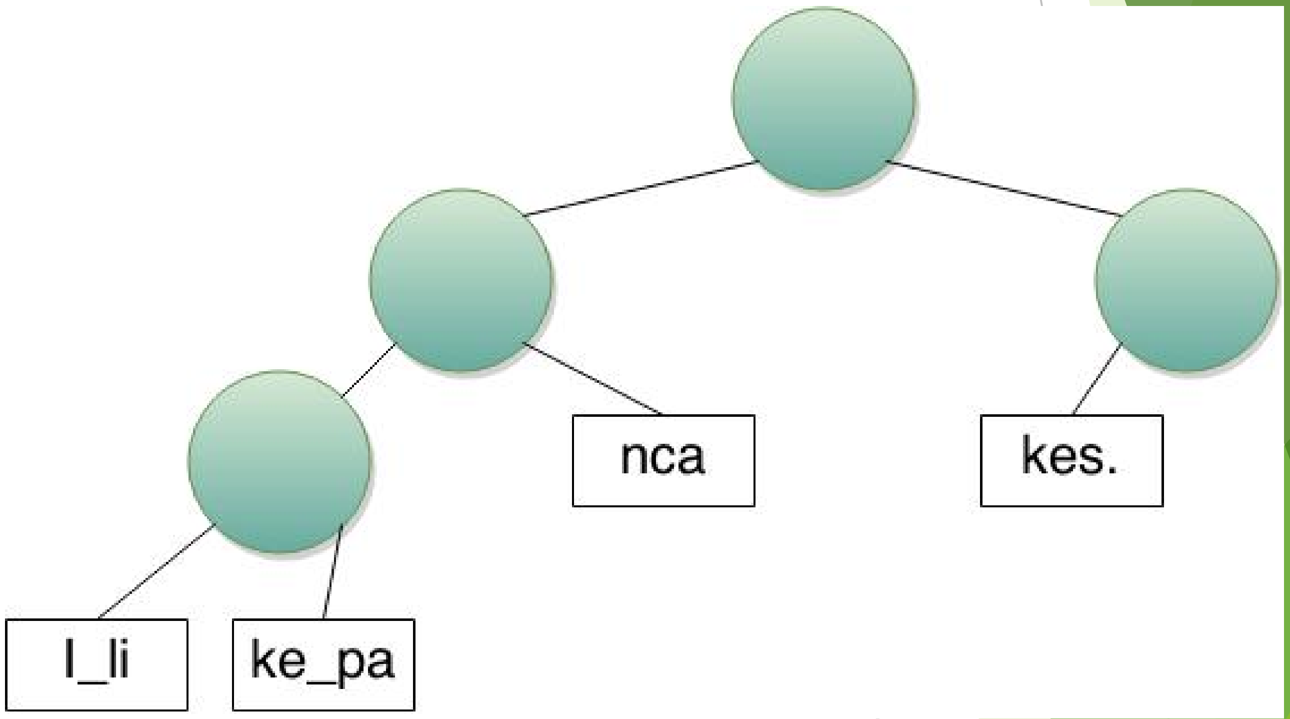
Rope

- A rope is a data structure that is meant to store a large string.
- When a large string is stored in a rope, it is divided into smaller “fragment” strings.
- Storing a large string in this way makes storage and manipulation more efficient.



Weight

- Each node has a weight value equal to the length of its string plus the sum of all leaf nodes' weight in its left subtree.
- **YOU: Label each node with its weight**

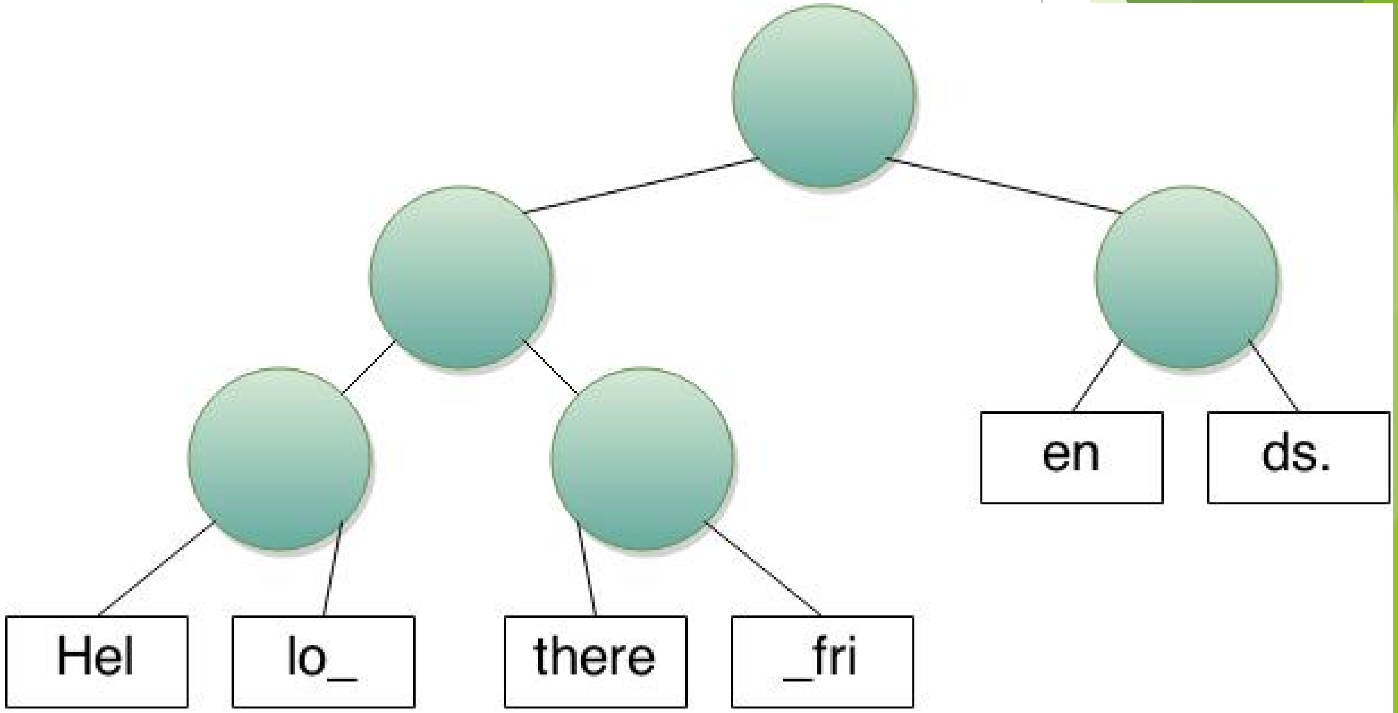


Index – O(log n)

- Here is code for index:

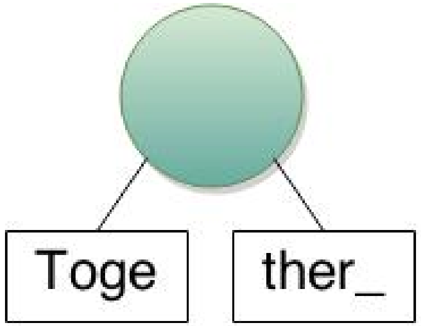
```
def index(root, i):
    if weight(root) <= i:
        return index(root.right, i - weight(root))
    else:
        if root.left != None:
            return index(root.left, i)
        else:
            return root.data[i]
```

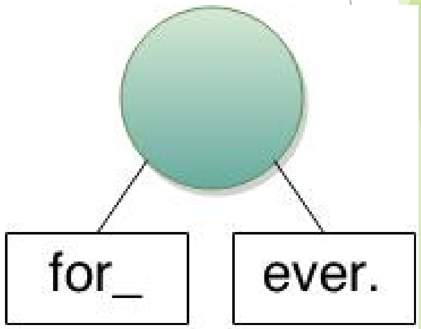
- **YOU: Trace the rope to find index(root, 12)**



Concatenate – O(log n)

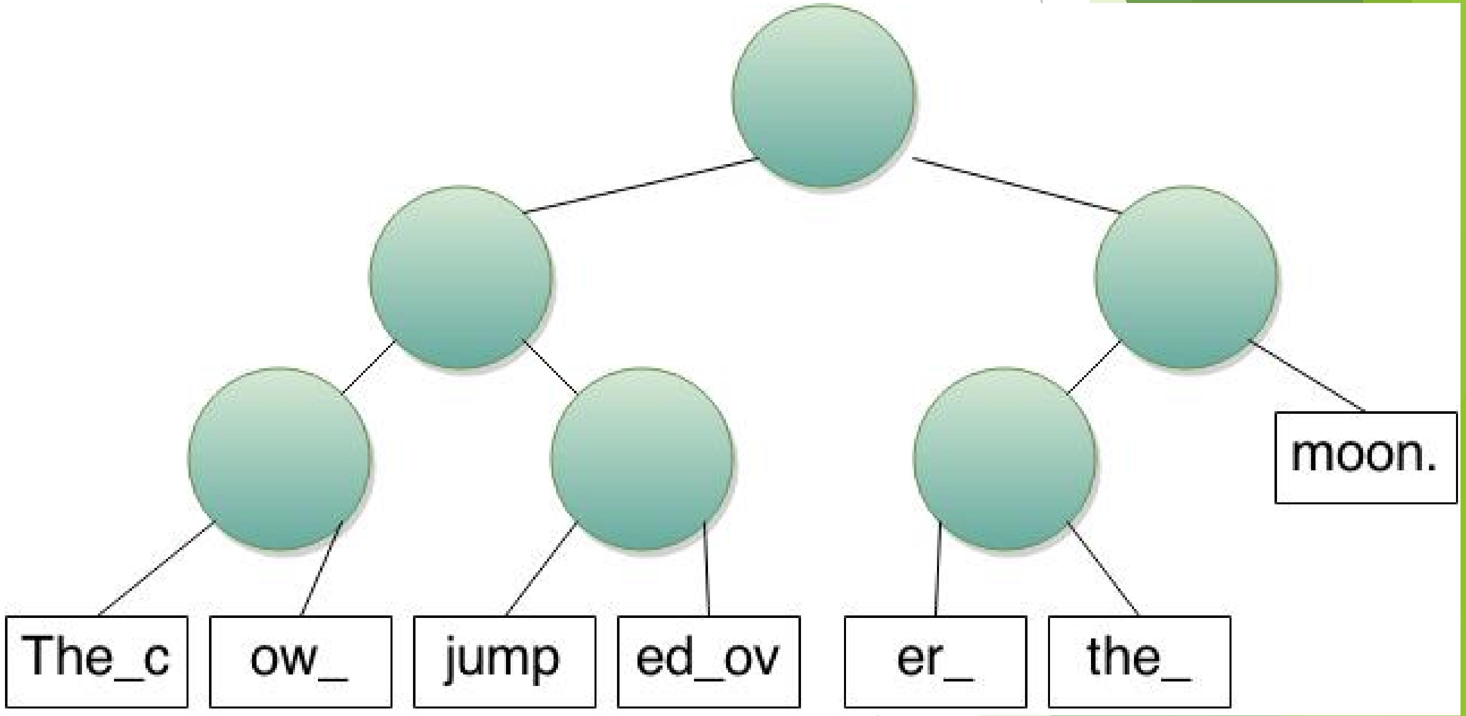
- Concat(S1, S2): concatenate two ropes, S1 and S2, into a single rope.
- A concatenation can be performed simply by creating a new root node with *left* = S_1 and *right* = S_2
- **YOU: Concat these two separate ropes into a single rope.**





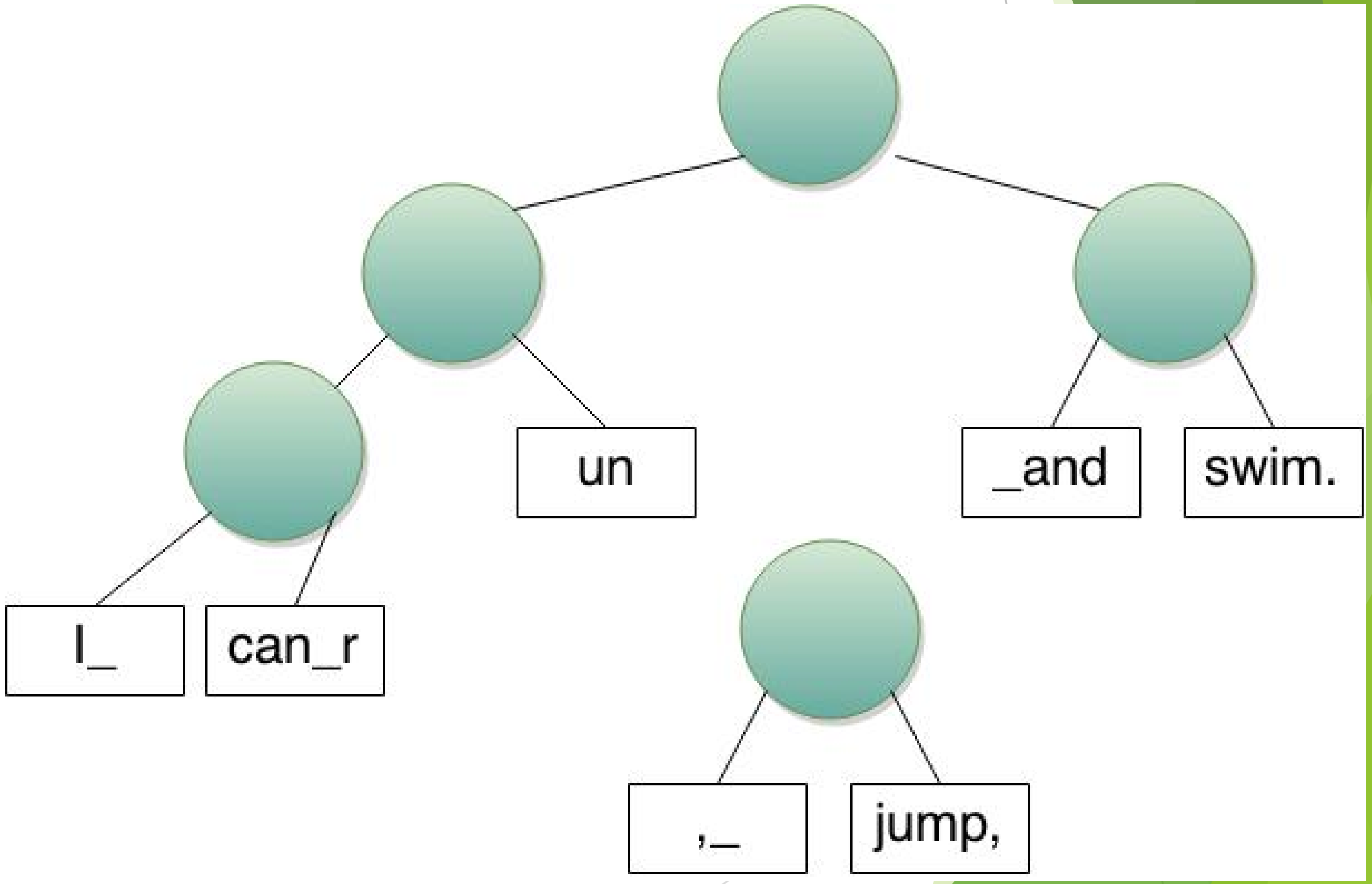
Split – O(log n)

- Split (i): split the rope into two new strings S1 and S2
- There are two cases that must be dealt with:
  1. The split point is at the end of a string (i.e. after the last character of a leaf node)
  2. The split point is in the middle of a string.
- **YOU: Perform split(15)**
  - Redraw the rope to compensate for the case 2.
  - Then, draw a line through the rope, across the branches to be removed.



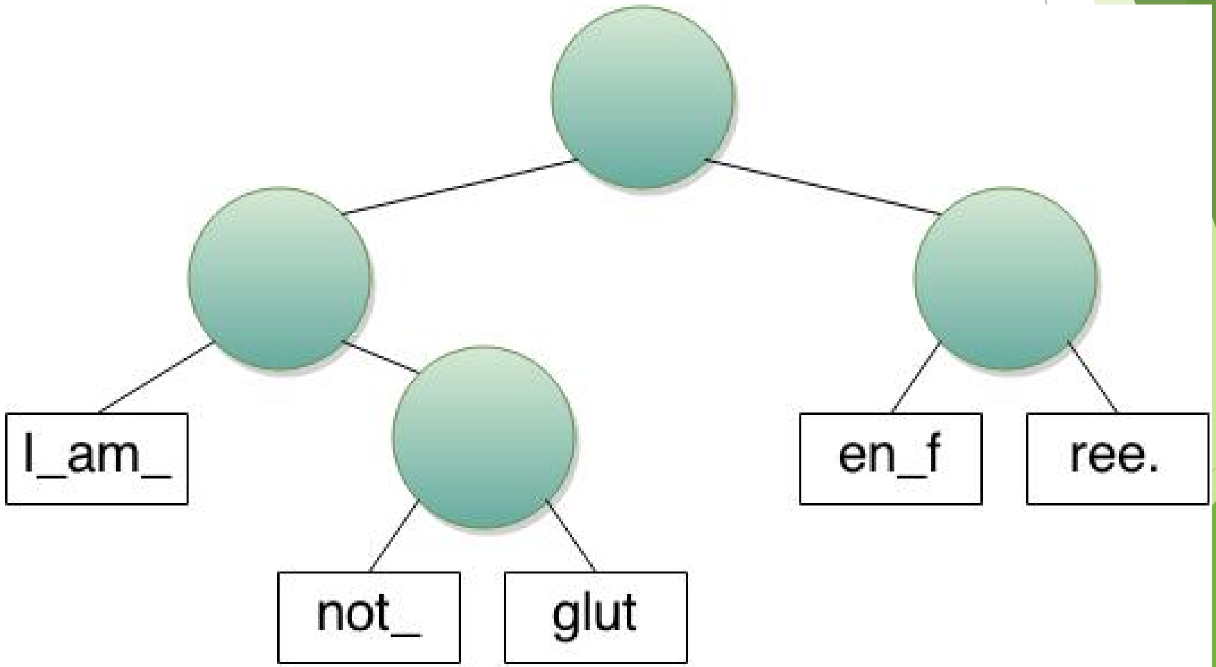
Insert – O(log n)

- Insert(i, S): instert string S at position i.
- Operation can be completed by:
  - Split()
  - Concat()
  - Concat()
- **YOU: Insert(9, “,_jump,”)**



Delete – O(log n)

- Delete(i, j): delete the substring at indices i to j Rope to form a new Rope.
- Operation can be complete by:
  - Split()
  - Split()
  - Concat()
- **YOU: Delete(5, 9)**



Performance

Operation	Rope	String
Build	O(n)	O(n)
Iterate over e/ character	O(n)	O(n)
Index	O(log n)	O(1)
Concatenate	O(log n)	O(n)
Split	O(log n)	O(1)
Insert	O(log n)	O(n)
Delete	O(log n)	O(n)