Back

Python, Straightforward with Explanation

awice ★ 4269 Last Edit: October 13, 2018 8:27 AM 2.5K VIEWS

36

We perform a recursive solution. There are four cases for what the string might look like:

1. empty
2. [integer]
3. [integer] ([tree])
4. [integer] ([tree]) ([tree])

When there is no '(', we are in one of the first two cases and proceed appropriately.
Else, we find the index "jx" of the ')' character that marks the end of the first tree. We do this by keeping a tally of how many left brackets minus right brackets we've seen. When we've seen 0, we must be at the end of the first tree. The second tree is going to be the expression S[jx + 2: -1], which might be empty if we are in case #3.

```
def str2tree(self, S):
    ix = S.find('(')
    if ix < 0:
        return TreeNode(int(S)) if S else None

    bal = 0
    for jx, u in enumerate(S):
        if u == '(': bal += 1
        if u == ')': bal -= 1
        if jx > ix and bal == 0:
            break

    root = TreeNode(int(S[:ix]))
    root.left = self.str2tree(S[ix+1:jx])
    root.right = self.str2tree(S[jx+2:-1])
    return root
```