i {} ⊖ ⊙ ∷

 Contest
 Discuss
 Store

 Interview
 Contest
 Discuss
 □ Store

 Image: Description
 □ Discuss
 □ Submissions

Medium ௴ 787 ♀ 129 ♡ Add to List ௴ Share

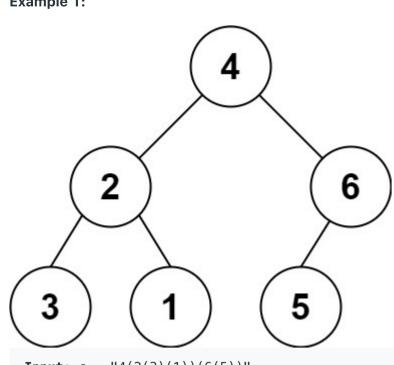
You need to construct a binary tree from a string consisting of parenthesis and integers.

536. Construct Binary Tree from String

The whole input represents a binary tree. It contains an integer followed by zero, one or two pairs of parenthesis. The integer represents the root's value and a pair of parenthesis contains a child binary tree with the same structure.

You always start to construct the **left** child node of the parent first if it exists.

Example 1:



Input: s = "4(2(3)(1))(6(5))"
Output: [4,2,6,3,1,5]

Example 2:

Input: s = "4(2(3)(1))(6(5)(7))"
Output: [4,2,6,3,1,5,7]

Example 3:

Input: s = "-4(2(3)(1))(6(5)(7))"
Output: [-4,2,6,3,1,5,7]

Constraints:

0 <= s.length <= 3 * 10⁴
 s consists of digits, '(', ')', and '-' only.

Accepted 63,120 | Submissions 115,759

Seen this question in a real interview before? Yes No

Companies 🚡 i

0 ~ 6 months 6 months ~ 1 year 1 year ~ 2 years

Facebook | 12

Related Topics

String Tree Depth-First Search Binary Tree

Similar Questions

Construct String from Binary Tree

1 v class Solution { public TreeNode str2tree(String s) {
 return this.str2treeInternal(s, 0).getKey(); public Pair<Integer, Integer> getNumber(String s, int index) { boolean isNegative = false; // A negative number 11 ▼ if (s.charAt(index) == '-') { isNegative = true; 13 index++; 14 int number = 0; while (index < s.length() && Character.isDigit(s.charAt(index))) {</pre> 17 ▼ 18 number = number * 10 + (s.charAt(index) - '0'); 19 20 index++; 21 22 23 24 25 **v** 26 return new Pair<Integer, Integer>(isNegative ? -number : number, index); public Pair<TreeNode, Integer> str2treeInternal(String s, int index) { 27 ▼ if (index == s.length()) { 28 29 return new Pair<TreeNode, Integer>(null, index); // Start of the tree will always contain a number representing // the root of the tree. So we calculate that first. Pair<Integer, Integer> numberData = this.getNumber(s, index); int value = numberData.getKey();
index = numberData.getValue(); TreeNode node = new TreeNode(value); Pair<TreeNode, Integer> data; // Next, if there is any data left, we check for the first subtree // which according to the problem statement will always be the left child. if (index < s.length() && s.charAt(index) == '(') {
 data = this.str2treeInternal(s, index + 1);</pre> 42 ▼ 43 node.left = data.getKey();
index = data.getValue(); 45 48 // Indicates a right child 49 50 ▼ if (node.left != null && index < s.length() && s.charAt(index) == '(') {</pre> data = this.str2treeInternal(s, index + 1); node.right = data.getKey();
index = data.getValue();

i Java

◆ Autocomplete

return new Pair<TreeNode, Integer>(node, index < s.length() && s.charAt(index) == ')' ? index + 1 : index);</pre>