Problem List

Description | Editorial | Solutions | Submissions | Testca

2196. Create Binary Tree From Descriptions

Solved

Medium | Topics | Companies | Hint

You are given a 2D integer array `descriptions` where `descriptions[i] = [parent_i, child_i, isLeft_i]` indicates that `parent_i` is the **parent** of `child_i` in a **binary** tree of **unique** values. Furthermore,

- If `isLeft_i == 1`, then `child_i` is the left child of `parent_i`.
- If `isLeft_i == 0`, then `child_i` is the right child of `parent_i`.

Construct the binary tree described by `descriptions` and return *its* ***root***.

The test cases will be generated such that the binary tree is **valid**.

Example 1:

```
/**
* Definition for a binary tree node.
* public class TreeNode {
*     public var val: Int
*     public var left: TreeNode?
*     public var right: TreeNode?
*     public init() { self.val = 0; self.left = nil; self.right =
nil; }
*     public init(_ val: Int) { self.val = val; self.left = nil;
self.right = nil; }
*     public init(_ val: Int, _ left: TreeNode?, _ right:
TreeNode?) {
*         self.val = val
*         self.left = left
*         self.right = right
*     }
* }
*/
class Solution {
```

```swift
func createBinaryTree(_ descriptions: [[Int]]) -> TreeNode? {

    var hash:[Int:TreeNode] = [:]
    var noparent = Set<Int>()
    var parentset = Set<Int>()

    for i in descriptions {
        var parent = hash[i[0]]
        if (parent == nil) {
            parent = TreeNode(i[0])
        }

        var child = hash[i[1]]
        if (child == nil) {
            child = TreeNode(i[1])
        }

        if (i[2] == 1) {
            parent?.left = child
        } else {
            parent?.right = child
        }

        hash[i[0]] = parent
        hash[i[1]] = child

        if noparent.contains(i[1]) {
            noparent.remove(i[1])
            parentset.insert(i[1])
        }

        if !parentset.contains(i[0]) {
            noparent.insert(i[0])
        }

        parentset.insert(i[1])

        // print(hash)
        // print("noparent ",noparent)

        // print("parentset ",parentset)
    }
```

```
        if let a = noparent.first {
            if let b = hash[a] {
                return b
            }
        }
        return nil
    }
}
```