116. Populating Next Right Pointers in Each Node

116 Populating Next Right Pointers in Each Node

- **Depth-first Search** + Tree

Description

Given a binary tree

```
struct TreeLinkNode {
  TreeLinkNode *left;
  TreeLinkNode *right;
  TreeLinkNode *next;
}
```

Populate each next pointer to point to its next right node. If there is no next right node, the next pointer should be set to `NULL`.

Initially, all next pointers are set to `NULL`.

Note:

- You may only use constant extra space.
- You may assume that it is a perfect binary tree (ie, all leaves are at the same level, and every parent has two children).

For example,
Given the following perfect binary tree,

```
      1
    /  \
   2    3
  / \  / \
 4  5  6  7
```

After calling your function, the tree should look like:

```
      1 -> NULL
    /  \
   2 -> 3 -> NULL
  / \  / \
 4->5->6->7 -> NULL
```

1. Thought line

2. Depth-first Search + Tree

```
/**
 * Definition for binary tree with next pointer.
```

```
 * struct TreeLinkNode {
 *  int val;
 *  TreeLinkNode *left, *right, *next;
 *  TreeLinkNode(int x) : val(x), left(NULL), right(NULL), next(NULL) {}
 * };
 */
class Solution {
private:
    void connect_fct(TreeLinkNode *node, TreeLinkNode *nodeNext) {
        if(node == nullptr) return;
        node->next = nodeNext;
        connect_fct(node->left, node->right);
        connect_fct(node->right, (nodeNext==nullptr)?nullptr:nodeNext->left);

    }
public:
    void connect(TreeLinkNode *root) {
        connect_fct(root, nullptr);
    }
};
```