剑指32 从上到下打印二叉树2

不会做

思考后：每层打印到一行，需要知道树的每一层分别有多少个节点，那么就应该在上一行节点在队列中pop的时候记录下他们的子节点数量总和。内部建立一个循环，循环中pop一行的节点。

```
// 经过提示思路后，我的答案。

class Solution {
public:
    vector<vector<int>> levelOrder(TreeNode* root) {
        if (root == NULL) return {};
        queue <TreeNode*> value;
        vector<vector<int>> output;
        TreeNode* q = root;
        value.push(q);
        int linelen = 1;
        while(!value.empty()) {
            int linelennext = 0;
            output.push_back({}); // push一个空vector代表下一行开始
            for (int i = 0; i < linelen; i++) {
                q = value.front();
                output.back().push_back(q->val);
                value.pop();
                if (q->left != NULL) {
                    value.push(q->left);
                    linelennext++;
                }
                if (q->right != NULL) {
                    value.push(q->right);
                    linelennext++;
                }
            }
            linelen = linelennext;
        }
        return output;
    }
};
```

```
// linelen其实不必要！ 可以优化！！
// 每次要打印一行的时候，queue的大小就代表了一行的长度。
// 优化后：

class Solution {
public:
    vector<vector<int>> levelOrder(TreeNode* root) {
        if (root == NULL) return {};
        queue <TreeNode*> value;
```

```
        vector<vector<int>> output;
        TreeNode* q = root;
        value.push(q);
        while (!value.empty()) {
            int linelen = value.size();
            output.push_back({});
            for (int i = 0; i < linelen; i++) {
                q = value.front();
                output.back().push_back(q->val);
                value.pop();
                if (q->left)
                    value.push(q->left);
                if (q->right) {
                    value.push(q->right);
                }
            }
        }
        return output;
    }
};
```