

## 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。

```
1 // 经过提示思路后，我的答案。
2
3 class Solution {
4 public:
5     vector<vector<int>> levelOrder(TreeNode* root) {
6         if (root == NULL) return {};
7         queue <TreeNode*> value;
8         vector<vector<int>> output;
9         TreeNode* q = root;
10        value.push(q);
11        int linelen = 1;
12        while(!value.empty()) {
13            int linelennext = 0;
14            output.push_back({}); // push一个空vector代表下一行开始
15            for (int i = 0; i < linelen; i++) {
16                q = value.front();
17                output.back().push_back(q->val);
18                value.pop();
19                if (q->left != NULL) {
20                    value.push(q->left);
21                    linelennext++;
22                }
23                if (q->right != NULL) {
24                    value.push(q->right);
25                    linelennext++;
26                }
27            }
28            linelen = linelennext;
29        }
30        return output;
31    }
32};
```

```
1 // linelen其实不必要！ 可以优化！！
2 // 每次要打印一行时，queue的大小就代表了一行的长度。
3 // 优化后：
4
5 class Solution {
6 public:
7     vector<vector<int>> levelOrder(TreeNode* root) {
8         if (root == NULL) return {};
9         queue <TreeNode*> value;
```

```

10 vector<vector<int>> output;
11 TreeNode* q = root;
12 value.push(q);
13 while (!value.empty()) {
14     int linen = value.size();
15     output.push_back({});
16     for (int i = 0; i < linen; i++) {
17         q = value.front();
18         output.back().push_back(q->val);
19         value.pop();
20         if (q->left)
21             value.push(q->left);
22         if (q->right) {
23             value.push(q->right);
24         }
25     }
26 }
27 return output;
28 }
29 };

```