102.二叉树的层序遍历 & 107.二叉树的层序遍历 II & 103.二叉树的锯齿形层序遍历

题目大意1

给定一个二叉树，把每一层从左到右打包成一个vector。

题目分析1

由于涉及到把一层的节点放到一起处理，我们考虑对树进行 BFS。我们给节点打上一个额外的标记 dep ，每次发现 dep 发生了改变，就把当前积累的元素打包成vector，清空当前存储。时间复杂度为 $O(n)$ ，具体代码如下：

```
vector< vector<int> > ans;
vector<int> temp;
struct Info {
    TreeNode* node;
    int dep;
};
int now_dep = 0;
queue<Info> q;
void process_and_push() {
    ans.push_back(temp);
    temp.clear();
}
vector<vector<int>> levelOrder(TreeNode* root) {
    if (!root) return ans;
    q.push({ root, 0 });
    while (!q.empty()) {
        auto now = q.front();
        q.pop();
        if (now.dep != now_dep) {
            process_and_push();
            now_dep = now.dep;
        }
        temp.push_back(now.node->val);
        if (now.node->left) q.push({ now.node->left, now.dep + 1 });
        if (now.node->right) q.push({ now.node->right, now.dep + 1 });
    }
    process_and_push();
```

```
    return ans;
}
```

题目大意2

在前一题基础上，最深的一层的 vector 包最先输出，根节点所在的包最后输出

题目分析2

这并不困难，我们只需使用 swap 把所返回的 vector< vector > 翻转一下即可，部分代码如下：

```
for(int i=0;i<ans.size()/2;i++) swap(ans[i], ans[ans.size()-i-1]);
```

题目大意3

在第一题基础上，对于奇数层，把节点从左向右打包成一个 vector；对于偶数层，把节点从右到左打包成一个 vector。

题目分析3

我们仍然在遍历结束之后对输出结果进行处理，只需把偶数层 vector 中的内容翻转一下即可，部分代码如下：

```
for(int j=0;j<ans.size();j++)
    if(j%2) for(int i=0;i<ans[j].size()/2;i++)
            swap(ans[j][i], ans[j][ans[j].size()-i-1]);
```