```
/*
给定一个二叉树，返回其节点值自底向上的层次遍历。 （即按从叶子节点所在层到根节点所在的层，逐层从左向右遍历）

例如：
给定二叉树 [3,9,20,null,null,15,7],

    3
   / \
  9  20
    /  \
   15   7

返回其自底向上的层次遍历为：

[
  [15,7],
  [9,20],
  [3]
]

来源：力扣（LeetCode）
链接：https://leetcode-cn.com/problems/binary-tree-level-order-traversal-ii
著作权归领扣网络所有。商业转载请联系官方授权，非商业转载请注明出处。
*/
```

分析：

- 从底向上遍历可以看成是从上向底遍历的逆序，我们可以先从上向底的结果依次入栈，然后逐一对单层结果出栈，由于要用到栈等数据结构，优先使用C++.

方法一: C++_Solution 先把遍历结果存储在stack中，然后再依次出栈压入容器.

```
/**
 * Definition for a binary tree node.
 * struct TreeNode {
 *     int val;
 *     TreeNode *left;
 *     TreeNode *right;
 *     TreeNode(int x) : val(x), left(NULL), right(NULL) {}
 * };
 */
class Solution
{
    public:
        vector<vector<int>> levelOrderBottom(TreeNode* root)
        {
            stack<vector<int>>      svi                 ;   /*层次遍历辅助队列*/
            queue<TreeNode*>        qt                  ;   /*逆序层次遍历辅助栈*/
```

```
        vector<vector<int>>     vvi                 ;   /*返回数据结构*/
        vector<int>             vi                  ;   /*存储每层遍历结果*/
        TreeNode*               temp    =   NULL    ;
        int                     num_1   =   0       ;
        int                     num_2   =   0       ;
        int                     i       =   0       ;

        if(root==NULL)
        {
            return vvi;
        }

        qt.push(root);
        num_1 = 0;
        num_2 = 1;

        /*从上而下的层序遍历并将结果推入栈*/
        while(num_2!=0)
        {
            num_1 = num_2;
            num_2 = 0;
            vi.clear();
            for(i=0;i<num_1;i++)
            {
                temp = qt.front();
                if(temp->left)
                {
                    num_2 ++;
                    qt.push(temp->left);
                }

                if(temp->right)
                {
                    num_2 ++;
                    qt.push(temp->right);
                }

                vi.push_back(temp->val);
                qt.pop();
            }
            svi.push(vi);
        }

        /*依次出栈,完成逆序*/
        while(svi.empty()!=true)
        {
            vvi.push_back(svi.top());
            svi.pop();
        }
        return vvi;
    }
};
```

```
/*
执行结果：
通过
显示详情
执行用时 :12 ms, 在所有 C++ 提交中击败了82.26%的用户
内存消耗 :13.6 MB, 在所有 C++ 提交中击败了96.42%的用户
*/
```

AlimyBreak
2019.07.22