**数据结构与算法笔记**

讨论一个算法的快慢时，我们考虑的是随着输入规模的增加，其操作数（或运行时间）会以什么样的速度增长。

数组和链表的优缺点：

数组：支持随机访问（立即访问到一个随机位置的元素），但是需要申请一大段连续内存，插入元素可能会导致整个数组转移到其他地方。（可以预留，但是有两个问题：一是预留的可能用不上造成浪费，二是预留的位置可能还是会被用完从而整个数组要转移）

链表：只能顺序访问（逐个元素读取），但是插入和删除元素操作很快。

每个递归都需要有两个部分：停止条件（或称基线条件）和递归条件。基线条件避免递归形成无限循环。/\*

// Definition for a Node.

class Node {

public:

int val = NULL;

vector<Node\*> children;

Node() {}

Node(int \_val, vector<Node\*> \_children) {

val = \_val;

children = \_children;

}

};

\*/

class Solution {

public:

vector<vector<int>> levelOrder(Node\* root) {

vector<vector<int> > ans;

if(root == NULL) return ans;

queue<Node\*> q;

q.push(root);

q.push(NULL);

vector<int> v;

while(!q.empty()) {

Node\* temp = q.front();

q.pop();

if(temp != NULL) {

v.push\_back(temp->val);

vector<Node\*> vec = temp->children;

for(auto a: vec) {

q.push(a);

}

} else {

ans.push\_back(v);

v.clear();

if(!q.empty())

q.push(NULL);

}

}

return ans;

}

};