210. Course Schedule II

作者：qianrong wu

思路

1. 这道题我们得找出要上的课程的顺序，即有向图的拓扑排序。
2. 拓扑排序的做法如下： 2.1. 每次找入度为0的点，输出该入度为0的点，并删除与之相连接的边 2.2. 重复1直到没有入度为0的点。之前输出入度为0的点若小于原图的结点数，那么说明图有环，即拓扑排序不存在，否则即为拓扑排序。
3. 我们定义二维数组graph来表示这个有向图，一位数组in来表示每个顶点的入度。我们开始先根据输入来建立这个有向图，并将入度数组也初始化好。然后我们定义一个queue变量，将所有入度为0的点放入队列中，然后开始遍历队列，从graph里遍历其连接的点，每到达一个新节点，将其入度减一，如果此时该点入度为0，则放入队列末尾。我们从queue中每取出一个数组就将其存在结果中，最终若有向图中有环，则结果中元素的个数不等于总课程数，那我们将结果清空即可。

解法c++

```
class Solution {
public:
    vector<int> findOrder(int numCourses, vector<pair<int, int>>& prerequisites) {
        vector<int> res;
        vector<vector<int> > graph(numCourses, vector<int>(0));
        vector<int> in(numCourses, 0);
        for (auto & a : prerequisites) {
            graph[a.second].push_back(a.first);
            ++in[a.first];
        }
        queue<int> q;
        for (int i = 0; i < numCourses; ++i) {
            if (in[i] == 0) q.push(i);
        }
        while (!q.empty()) {
            int t = q.front();
            res.push_back(t);
            q.pop();
            for (auto &a : graph[t]) {
                --in[a];
                if (in[a] == 0) q.push(a);
            }
        }
        if (res.size() != numCourses) res.clear();
        return res;
    }
};
```

总结

LeetCode中关于图的题很少，有向图的仅此一道，还有一道关于无向图的题是 Clone Graph 无向图的复制。图这种数据结构相比于树，链表要更为复杂一些，尤其是有向图，很麻烦。