210. Course Schedule II

题目描述：https://leetcode.com/problems/course-schedule-ii/

> 选课方案问题，求图的拓扑排序，给定很多对[i,j]意思是选i必须选j，求选课顺序。

解题思路：

拓扑排序问题。
1. 用indegree数组记录一下
2. DFS

代码1：

```
class Solution {
public:
    vector<int> findOrder(int num, vector<pair<int, int>>& pre) {
        vector<unordered_set<int>> graph(num);
        vector<int> res;
        vector<bool> visited(num, false);
        vector<int> indegree(num, 0);
        for(auto item: pre) {
            if(graph[item.second].find(item.first) == graph[item.second].end()) {
                graph[item.second].insert(item.first);
                indegree[item.first]++;
            }
        }
        bool ok = true;
        while(res.size() < num && ok) {
            ok = false;
            for(int i = 0; i < num; i++) {
                if(indegree[i] == 0 && !visited[i]) {
                    res.push_back(i);
                    visited[i] = true;
                    ok = true;
                    for(auto item: graph[i]) {
                        indegree[item]--;
                    }
                }
            }
        }
        if(res.size() == num) return res;
        return {};
    }
};
```

代码2:

```
class Solution {
public:
    bool dfs(vector<unordered_set<int> > g, vector<int> &res, vector<int> &visited
, int i) {
        if(visited[i] == 1) return true;
        if(visited[i] == -1) return false;
        visited[i] = -1;
        for(auto item: g[i]) {
            if(!dfs(g, res, visited, item)) {
                return false;
            }
        }
        visited[i] = 1;
        res.push_back(i);
        return true;
    }
    vector<int> findOrder(int numCourses, vector<pair<int, int>>& prerequisites) {
        vector<unordered_set<int> > g(numCourses);
        vector<int> res;
        vector<int> visited(numCourses, 0);
        for(auto item: prerequisites) {
            if(g[item.second].find(item.first) == g[item.second].end()) {
                g[item.second].insert(item.first);
            }
        }
        for(int i = 0; i < numCourses; i++) {
            if(visited[i] == 0 && !dfs(g, res, visited, i)) {
                return {};
            }
        }
        reverse(res.begin(), res.end());
        return res;
    }
};
```