题目

A TOPO!

时间限制：1000ms 内存限制：65536kb

通过率：183/183 (100.00%) 正确率：183/765 (23.92%)

题目描述

给定 n 个点，m 条有向边的有向无环图，求拓扑排序。

P.S. 对于拓扑排序不唯一的情况，先输出序号大的点，再输出序号小的点。即输出字典序最大的拓扑排序。

输入格式

第一行两个正整数 n, m $(2 \leq n \leq 10^5, 1 \leq m \leq 4 \times 10^5)$

接下来 m 行，每行两个正整数 u，v $(1 \leq u_i, v_i \leq n)$，表示 u 到 v 存在有向边。

输出格式

输出一行，表示拓扑排序，点的序号以空格分隔

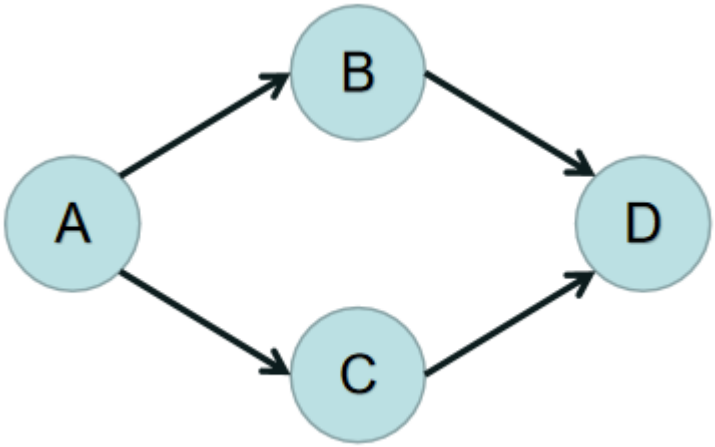
思路

本题考察拓扑排序

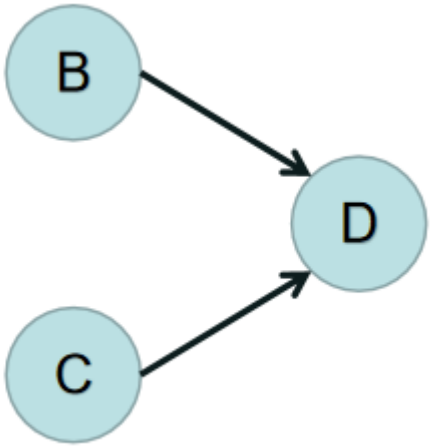
啥是拓扑排序?

- 一个有向图，如果图中有入度为 0 的点，就把这个点删掉，同时也删掉这个点所连的边。
- 一直进行上面出处理，如果所有点都能被删掉，则这个图可以进行拓扑排序。

举例子



开始时，图是这样的状态，发现A的入度为



0，所以删除A和A上所连的边，结果如下图 这时发现B的入度为 0，C的入度为 0，所以删除B和B上所连的边、C和C上所连的边，结果如下图：

这时发现发现D的入度为 0，所以删除D和D上所连的边(如果有就删)，这时整个图被删除干净，所有能进行拓扑排序。

解题思路

- 首先记录各个点的入度
- 然后将入度为 0 的点放入队列
- 将队列里的点依次出队列，然后找出所有出队列这个点发出的边，删除边，同事边的另一侧的点的入度 -1。
- 如果所有点都进过队列，则可以拓扑排序，输出所有顶点。否则输出-1，代表不可以进行拓扑排序。

代码实现

```
#include <iostream>
#include <vector>
#include <queue>
#include <cstring>
#include <algorithm>

using namespace std;

const int N = 4e5 + 10;

int h[N], e[N], ne[N], idx;
int d[N];
int n, m;

void add(int a, int b) {
    e[idx] = b;
    ne[idx] = h[a];
    h[a] = idx++;
}

bool topsort(vector<int>& result) {
    priority_queue<int> pq;

    for (int i = 1; i <= n; i++) {
        if (d[i] == 0) {
            pq.push(i);
        }
    }

    while (!pq.empty()) {
        int a = pq.top();
        pq.pop();

        result.push_back(a);

        for (int i = h[a]; i != -1; i = ne[i]) {
            int b = e[i];
            d[b]--;
            if (d[b] == 0) {
                pq.push(b);
            }
        }
    }


    return result.size() == n;
}

int main() {
```

```
    cin >> n >> m;
    memset(h, -1, sizeof h);

    for (int i = 0; i < m; i++) {
        int u, v;
        cin >> u >> v;
        add(u, v);
        d[v]++;
    }

    vector<int> result;
    if (topsort(result)) {
        for (int i = 0; i < n; i++) {
            cout << result[i] << " ";
        }
    }

    return 0;
}
```

要注意的细节

N = 4e5 + 10;