LC797所有可能的路径

- https://leetcode.cn/problems/all-paths-from-source-to-target/submissions/

希冀平台（3457. 独立路径数计算）

C++

```
#include<stdio.h>
#include<stdlib.h>
#define MAXN  512

int  vu[MAXN];
int  vv[MAXN];
int vid[MAXN];

typedef char bool;
typedef struct edge *pnode;
struct edge{
    int   id; //edge index
    int   v;
    pnode next;
};
pnode adj[MAXN];
bool  vis[MAXN];
int  path[MAXN];
int  cnt=0;

int  n,m;
int  s,t;

void insertEdge(int u, int v, int id){
    pnode ed = (pnode)malloc(sizeof(struct edge));
    ed->v    = v;
    ed->id   = id;
    ed->next = adj[u];
    adj[u]   = ed;
}

void dfs(int u){
    //printf("%d\n",u);
    if(u==t){
        for(int i=1;i<=cnt;++i) printf("%d ",path[i]);
        printf("\n");
        return;
    }
    for(pnode e=adj[u]; e; e=e->next) {
        //printf("%d %d\n",e->id, e->v);
        int v = e->v;
        if(vis[v]) continue;
        vis[v] = 1;
        path[++cnt] = e->id;
```

```
45          dfs(v);
46          --cnt;
47          vis[v] = 0;
48      }
49  }
50
51  int main(){
52      scanf("%d %d", &n, &m);
53      for(int i=0; i<m; ++i){
54          scanf("%d %d %d", &vid[i], &vu[i], &vv[i]);
55      }
56      for(int i=m-1; i>=0; --i){
57          insertEdge(vu[i],vv[i],vid[i]);
58          insertEdge(vv[i],vu[i],vid[i]);
59      }
60      s = 0; t = n-1;
61      vis[s] = 1;
62      dfs(s);
63      return 0;
64  }
```

C++

```cpp
class Solution {
public:

    vector<vector<int>> allPathsSourceTarget(vector<vector<int>>& graph)
{
        int n = graph.size();
        vector<bool> vis(n);
        int t = n-1;
        int s = 0;

        vector<vector<int>> res;
        vector<int>  path;

        function<void(int)> dfs;
        dfs = [&](int u){
            if(u==t){
                res.push_back(path);
                return;
            }
            for(auto v:graph[u]){
                if(vis[v]) continue;
                vis[v] = true;
                path.push_back(v);
                dfs(v);
                path.pop_back();
                vis[v] = false;
            }
        };

        path.push_back(0);
        dfs(s);
        return res;
    }
};
```