搜索-DFS之连通性模型

一、AcWing 1112. 迷宫

【题目描述】

一天Extense在森林里探险的时候不小心走入了一个迷宫，迷宫可以看成是由$n*n$的格点组成，每个格点只有2种状态，`.`和`#`，前者表示可以通行后者表示不能通行。

同时当Extense处在某个格点时，他只能移动到东南西北(或者说上下左右)四个方向之一的相邻格点上，Extense想要从点A走到点B，问在不走出迷宫的情况下能不能办到。

如果起点或者终点有一个不能通行(为`#`)，则看成无法办到。

注意：A,B不一定是两个不同的点。

【输入格式】

第1行是测试数据的组数k，后面跟着k组输入。

每组测试数据的第1行是一个正整数n，表示迷宫的规模是$n*n$的。

接下来是一个$n*n$的矩阵，矩阵中的元素为`.`或者`#`。

再接下来一行是4个整数x_1,y_1,x_2,y_2，描述A处在第x_1行，第y_1列，B处在第x_2行，第y_2列。

注意到x_1,y_1,x_2,y_2全部是从0开始计数的。

【输出格式】

k行，每行输出对应一个输入。

能办到则输出`YES`，否则输出`NO`。

【数据范围】

$1 \leq n \leq 100$

【输入样例】

```
2
3
.##
..#
#..
0 0 2 2
5
.....
###.#
..#..
###..
...#.
0 0 4 0
```

【输出样例】

```
YES
NO
```

【分析】

很简单的题目，当不需要求解最短距离时使用DFS求出起点与终点是否连通即可，注意本题的坑点为起点也可能是`#`，因此将判断条件放在DFS的开头部分比较好。

【代码】

```cpp
#include <iostream>
#include <cstring>
#include <algorithm>
using namespace std;

const int N = 110;
char g[N][N];
int x1, y1, x2, y2;
int n, k;
bool st[N][N];
int dx[4] = { -1, 0, 1, 0 }, dy[4] = { 0, 1, 0, -1 };

bool dfs(int x, int y)
{

    if (g[x][y] == '#' || x < 0 || x >= n || y < 0 || y >= n || st[x][y])
```

```
    return false;
16      if (x == x2 && y == y2) return true;
17      st[x][y] = true;
18      for (int i = 0; i < 4; i++)
19          if (dfs(x + dx[i], y + dy[i])) return true;
20      return false;
21  }
22
23  int main()
24  {
25      cin >> k;
26      while (k--)
27      {
28          cin >> n;
29          for (int i = 0; i < n; i++) cin >> g[i];
30          memset(st, false, sizeof st);
31          cin >> x1 >> y1 >> x2 >> y2;
32          if (dfs(x1, y1)) puts("YES");
33          else puts("NO");
34      }
35      return 0;
36  }
```

二、AcWing 1113. 红与黑

【题目描述】

有一间长方形的房子，地上铺了红色、黑色两种颜色的正方形瓷砖。

你站在其中一块黑色的瓷砖上，只能向相邻（上下左右四个方向）的黑色瓷砖移动。

请写一个程序，计算你总共能够到达多少块黑色的瓷砖。

【输入格式】

输入包括多个数据集合。

每个数据集合的第一行是两个整数W和H，分别表示x方向和y方向瓷砖的数量。

在接下来的H行中，每行包括W个字符。每个字符表示一块瓷砖的颜色，规则如下

1. `.`：黑色的瓷砖；
2. `#`：红色的瓷砖；
3. `@`：黑色的瓷砖，并且你站在这块瓷砖上。该字符在每个数据集合中唯一出现一次。

当在一行中读入的是两个零时，表示输入结束。

【输出格式】

对每个数据集合，分别输出一行，显示你从初始位置出发能到达的瓷砖数（记数时包括初始位置的瓷砖）。

【数据范围】

$1 \leq W, H \leq 20$

【输入样例】

```
6 9
....#.
.....#
......
......
......
......
......
#@...#
.#..#.
0 0
```

【输出样例】

```
45
```

【分析】

也是很简单的题目，求解起点所在的连通块中点的数量，唯一的坑点是输入n, m时是和传统的输入相反的。

【代码】

```
#include <iostream>
#include <cstring>
#include <algorithm>
using namespace std;

const int N = 30;
char g[N][N];
bool st[N][N];

int n, m;
```

```cpp
int dx[4] = { -1, 0, 1, 0 }, dy[4] = { 0, 1, 0, -1 };

int dfs(int x, int y)
{
    int res = 1;
    st[x][y] = true;
    for (int i = 0; i < 4; i++)
    {
        int nx = x + dx[i], ny = y + dy[i];
        if (nx >= 0 && nx < n && ny >= 0 && ny < m && g[nx][ny] == '.' && !st[nx][ny])
            res += dfs(nx, ny);
    }
    return res;
}

int main()
{
    while (cin >> m >> n, n || m)
    {
        int sx, sy;
        for (int i = 0; i < n; i++)
            for (int j = 0; j < m; j++)
            {
                cin >> g[i][j];
                if (g[i][j] == '@') sx = i, sy = j;
            }
        memset(st, false, sizeof st);
        cout << dfs(sx, sy) << endl;
    }
    return 0;
}
```