

# 7.DFS

## 1、模板

```
1  int mp[100][100],vis[100][100];
2  bool flag = false;
3
4  int dx[4] = {1,0,-1,0};
5  int dy[4] = {0,-1,0,1};
6
7  void dfs(int x, int y) //传入参数是当前的坐标，注意起点位置是否合法
8  {
9      if (vis[x][y]) return; //表示已经访问过了 递归回溯
10     vis[x][y] = true;
11
12     if( x == n && y == n) //递归终止
13         return;
14
15     for (int i = 0; i < 4; i++)
16     {
17         int nx = x + dx[i];
18         int ny = y + dy[i];
19         if (nx > 0 && nx <= H && ny > 0 && ny <= W && !vis[nx][ny] && mp[nx][ny])
20             dfs(nx,ny);
21     }
22
23     vis[x][y] = false;
24 }
```

## 2、中国象棋

```
1  #include <iostream>
2  #include <stack>
3  using namespace std;
4  char s[10][10];
5  bool vis[10][10];
6  bool flag = false;
7
8  int dx[8] = {2,2,1,1,-1,-1,-2,-2};
9  int dy[8] = {1,-1,2,-2,2,-2,1,-1};
10
11 void dfs(int x, int y)
12 {
13     if (vis[x][y]) return;
```

```

14     vis[x][y] = true;
15     if (s[x][y] == 'T')
16     {
17         flag =true;
18         return;
19     }
20     for (int i = 0; i < 8; i++)
21     {
22         int nx = x + dx[i];
23         int ny = y + dy[i];
24         if (nx >= 0 && nx <10 && ny >= 0 && ny < 10 && s[nx][ny] !='#' && !\
25         {
26             dfs(nx,ny);
27         }
28     }
29 }
30
31 int main()
32 {
33     int x,y;
34     for (int i = 0; i < 10; i++)
35     {
36         scanf("%s",s[i]);
37     }
38     for (int i = 0; i < 10; i++)
39     {
40         for (int j = 0; j < 10; j++)
41         {
42             if (s[i][j] == 'S')
43             {
44                 x = i;
45                 y = j;
46             }
47         }
48     }
49     dfs(x,y);
50     if (flag)
51         printf("YES");
52     else
53         printf("NO");
54
55     return 0;
56 }

```

### 3、迷宫最短路

```

1  #include <iostream>
2  #include <string>
3  using namespace std;
4  int n,m;
5  string maze[110];

```

```

6  bool vis[110][110];
7  bool flag = false;
8  int ans = 1000000;
9  int dx[4] = {1,0,-1,0};
10 int dy[4] = {0,-1,0,1};
11
12 void dfs(int x, int y,int step)
13 {
14     if (vis[x][y]) return;
15     vis[x][y] = true;
16     if (maze[x][y] == 'T')
17     {
18         if (step < ans)
19             ans = step;
20         return;
21     }
22     for (int i = 0; i < 4; i++)
23     {
24         int nx = x + dx[i];
25         int ny = y + dy[i];
26         if (nx >= 0 && nx < n && ny >= 0 && ny < m && maze[nx][ny] != '*' && !
27         {
28             dfs(nx,ny,step+1);
29         }
30     }
31     vis[x][y] = false;
32 }
33
34 int main()
35 {
36     int x,y;
37     cin >> n >> m;
38     for (int i = 0; i < n; i++)
39     {
40         cin >> maze[i];
41     }
42     for (int i = 0; i < n; i++)
43     {
44         for (int j = 0; j < m; j++)
45         {
46             if (maze[i][j] == 'S')
47             {
48                 x = i;
49                 y = j;
50             }
51         }
52     }
53     dfs(x,y,0);
54     cout << ans <<endl;
55
56     return 0;

```

```
57 }
```

## 4、踏青

```
1 测试样例
2 5 6
3 .#....
4 ..#...
5 ..#..#
6 ...##.
7 .#....
```

```
1 #include <iostream>
2 #include <string>
3 using namespace std;
4 int n,m;
5 string mp[105];
6 bool vis[105][105];
7 bool flag = false;
8 int cnt = 0;
9 int dx[4] = {1,0,-1,0};
10 int dy[4] = {0,-1,0,1};
11
12 void dfs(int x, int y)
13 {
14     if (vis[x][y]) return;
15     vis[x][y] = true;
16     for (int i = 0; i < 4; i++)
17     {
18         int nx = x + dx[i];
19         int ny = y + dy[i];
20         if (nx >= 0 && nx < n && ny >= 0 && ny < m && mp[nx][ny] != '.' && !vis[nx][ny])
21         {
22             dfs(nx,ny);
23         }
24     }
25 }
26
27 int main()
28 {
29     int x,y;
30     cin >> n >> m;
31     for (int i = 0; i < n; i++)
32     {
33         cin >> mp[i];
34     }
35     for (int i = 0; i < n; i++)
36     {
37         for (int j = 0; j < m; j++)
38         {
```

```

39         if (mp[i][j] == '#' && !vis[i][j])
40         {
41             dfs(i,j);
42             cnt ++;
43         }
44     }
45 }
46 cout << cnt <<endl;
47
48 return 0;
49 }

```

## 5、迷宫解的方案数

```

1 5 5
2 s####
3 .####
4 .####
5 .####
6 ....e

```

```

1 #include <iostream>
2 #include <string>
3 using namespace std;
4 int n,m;
5 string mp[15];
6 bool vis[15][15];
7 bool flag = false;
8 int cnt = 0;
9 int dx[4] = {1,0,-1,0};
10 int dy[4] = {0,-1,0,1};
11
12 void dfs(int x, int y)
13 {
14     if (vis[x][y]) return;
15     vis[x][y] = true;
16     if (mp[x][y] == 'e')
17     {
18         cnt++;
19         return;
20     }
21     for (int i = 0; i < 4; i++)
22     {
23         int nx = x + dx[i];
24         int ny = y + dy[i];
25         if (nx >= 0 && nx < n && ny >= 0 && ny < m && mp[nx][ny] != '#' && !vis[nx][ny])
26         {
27             dfs(nx,ny);
28         }
29     }

```

```

30     vis[x][y] = false;
31 }
32
33 int main()
34 {
35     int x,y;
36     cin >> n >> m;
37     for (int i = 0; i < n; i++)
38     {
39         cin >> mp[i];
40     }
41     for (int i = 0; i < n; i++)
42     {
43         for (int j = 0; j < m; j++)
44         {
45             if (mp[i][j] == 's' && !vis[i][j])
46             {
47                 dfs(i,j);
48             }
49         }
50     }
51     cout << cnt << endl;
52
53     return 0;
54 }

```

## 6、最大的蛋糕块

```

1 5 6
2 .#....
3 ..#...
4 ..#..#
5 ...##.
6 .#....

```

```

1 #include <iostream>
2 #include <string>
3 using namespace std;
4 int n,m,ans;
5 string mp[1005];
6 bool vis[1005][1005];
7 bool flag = false;
8 int cnt = 0;
9 int dx[4] = {1,0,-1,0};
10 int dy[4] = {0,-1,0,1};
11
12 void dfs(int x, int y)
13 {
14     if (vis[x][y]) return;
15     vis[x][y] = true;

```

```

16     cnt++;
17     if (mp[x][y] == 'e')
18     {
19         cnt++;
20         return;
21     }
22     for (int i = 0; i < 4; i++)
23     {
24         int nx = x + dx[i];
25         int ny = y + dy[i];
26         if (nx >= 0 && nx < n && ny >= 0 && ny < m && mp[nx][ny] != '.' && !vis[nx][ny])
27         {
28             dfs(nx,ny);
29         }
30     }
31 }
32
33 int main()
34 {
35     int x,y;
36     cin >> n >> m;
37     for (int i = 0; i < n; i++)
38     {
39         cin >> mp[i];
40     }
41     for (int i = 0; i < n; i++)
42     {
43         for (int j = 0; j < m; j++)
44         {
45             if (mp[i][j] == '#' && !vis[i][j])
46             {
47                 cnt = 0;
48                 dfs(i,j);
49                 if (cnt > ans)
50                     ans =cnt;
51             }
52         }
53     }
54     cout << ans <<endl;
55
56     return 0;
57 }

```

## 7、家谱？

输出  $n$  行，每行有一个整数，表示第  $i$  个人有多少个直系后代。

样例输入

复制

```
4
1 2
1 3
2 4
```

样例输出

复制

```
3
1
0
0
```

```
1 #include <iostream>
2 #include <string>
3 #include <vector>
4 using namespace std;
5 int n;
6 vector<int> son[100005];
7 bool f[100005];
8 int ans[100005];
9 int dx[4] = {1,0,-1,0};
10 int dy[4] = {0,-1,0,1};
11
12 int dfs(int u)
13 {
14     int ret = 0;
15     for (int i = 0; i < son[u].size(); i++)
16     {
17         ret+= dfs(son[u][i]);
18     }
19     ans[u] = ret;
20     return ret + 1;
21 }
22
23 int main()
24 {
25     int x,y,u;
26     cin >> n ;
27     for (int i = 0; i < n-1; i++)
28     {
29         cin >> x >> y;
30         son[x].push_back(y);
31         f[y] = true;
32     }
33     for (int i = 1; i <= n; i++)
34     {
35         if (!f[i])
36         {
37             u = i;
38             break;
39         }
40     }
41     dfs(u);
```



```

42     for (int i = 1; i <= n; i++)
43     {
44         printf("%d\n",ans[i]);
45     }
46
47     return 0;
48 }

```

## 8、马的覆盖点

```

1  #include <iostream>
2  #include <string>
3  #include <vector>
4  using namespace std;
5  int n,m;
6  char s[105][105];
7  int dx[8] = {2,2,1,1,-1,-1,-2,-2};
8  int dy[8] = {1,-1,2,-2,2,-2,1,-1};
9
10 void dfs(int x, int y,int step)
11 {
12     if (step > 3)
13         return;
14     for (int i = 0; i < 8; i++)
15     {
16         int nx = x + dx[i];
17         int ny = y + dy[i];
18         if (nx >= 0 && nx < n && ny >= 0 && ny < m && s[nx][ny] == '.' )
19         {
20             s[x][y] = '#';
21             dfs(nx,ny,step+1);
22         }
23     }
24 }
25
26 int main()
27 {
28     int x,y;
29     cin >> n >> m >> x >> y;
30
31     for (int i = 0; i < n; i++)
32     {
33         for (int j = 0; j < m; j++)
34         {
35             s[i][j] = '.';
36
37         }
38     }
39     dfs(x-1,y-1,0);
40     for (int i = 0; i < n; i++)
41     {

```

```
42     printf("%s\n",s[i]);
43 }
44
45
46     return 0;
47 }
```