在一个给定形状的棋盘（形状可能是不规则的）上面摆放棋子，棋子没有区别。要求摆放时任意的两个棋子不能放在棋盘中的同一行或者同一列，请编程求解对于给定形状和大小的棋盘，摆放k个棋子的所有可行的摆放方案C。

**Input**

输入含有多组测试数据。   
每组数据的第一行是两个正整数，n k，用一个空格隔开，表示了将在一个n\*n的矩阵内描述棋盘，以及摆放棋子的数目。 n <= 8 , k <= n   
当为-1 -1时表示输入结束。   
随后的n行描述了棋盘的形状：每行有n个字符，其中 # 表示棋盘区域， . 表示空白区域（数据保证不出现多余的空白行或者空白列）。

**Output**

对于每一组数据，给出一行输出，输出摆放的方案数目C （数据保证C<2^31）。

**Sample Input**

2 1

#.

.#

4 4

...#

..#.

.#..

#...

-1 -1

**Sample Output**

2

1

#include<stdio.h>

#include<iostream>

#include<cstring>

using namespace std;

int ans,n,num;

int col[10];

char a[10][10];

void dfs(int row,int k)

{

if(k==num)

{

ans++;

return;

}

if(row>n)return;

for(int j=1;j<=n;j++)

if(a[row][j]=='#'&&col[j]==0)

{

col[j]=1;

dfs(row+1,k+1);

col[j]=0;

}

dfs(row+1,k);//在这一行先不放棋子

}

int main()

{

//freopen("input.txt","r",stdin);

while(cin>>n>>num)

{

if(n==-1&&num==-1) return 0;

for(int i=1;i<=n;i++)

for(int j=1;j<=n;j++)

cin>>a[i][j];

memset(col,0,sizeof(col));

ans=0;

dfs(1,0);

cout<<ans<<endl;

}

return 0;

}

//这个题和N皇后的区别是一是多了障碍点，但却少了斜行的要求

二是棋子个数不确定，所以要加个回溯

行数既可以通过一个总的循环来处理，也可以就是作为dfs的一个参数，而在dfs中判个越界就行了