战场索敌 题目描述：

有一个大小是 NxM 的战场地图， 被墙壁'#'分隔成大小不同的区域，上下左右四个方向相邻的空地'.'属于同一个区域，只有空地上可能存在敌人'E'，请求出地图上总共有多少区域里的敌人数小于 K。

输入描述：

第一行输入为 N,M,K;

N 表示地图的行数， M 表示地图的列数， K 表示目标敌人数量 N，M<=100;

之后为一个 NxM 大小的字符数组。

输出描述：

敌人数小于 K 的区域数量

补充说明：

示例 1

输入：

```
3 5 2
..#EE
E.#E.
###..
```

输出：

```
1
```

说明：

地图被墙壁分为两个区域， 左边区域有 1 个敌人， 右边区域有 3 个敌人， 符合条件的区域数量是 1

```
#include<sstream>

#include<iostream>

#include<istream>
```

```
#include<vector>
#include<algorithm>
#include<queue>
using namespace std;

int findenemy(int x, int y, vector<vector<int>>& matrix,
              vector<vector<bool>>& vist) {
    queue < pair<int, int>> q;
    vist[x][y] = true;
    q.push(make_pair(x, y));
    int row, col, num = 0;
    while (!q.empty()) {
        row = q.front().first;
        col = q.front().second;
        q.pop();
        if (matrix[row][col] == 2) num++;
        // down
        if (row + 1 < matrix.size() && matrix[row + 1][col] !=
0&&!vist[row+1][col]) {
            q.push(make_pair(row + 1, col));
            vist[row+1][col] = true;
        }
```

```
        //right
        if (col + 1 < matrix[row].size() && matrix[row][col + 1] !=
0&&!vist[row][col+1]) {
            q.push(make_pair(row, col + 1));
            vist[row][col+1] = true;
        }
        //up
if (row - 1 >=0 && matrix[row - 1][col] != 0&&!vist[row-1][col]) {
            q.push(make_pair(row - 1, col));
            vist[row-1][col] = true;
        }
        //left
        if (col - 1 >=0 && matrix[row][col - 1] != 0&&!vist[row][col-1]) {
            q.push(make_pair(row, col - 1));
            vist[row][col-1] = true;
        }

    }
    return num;
}
int main() {
    string tmp;
```

```
int N, M, K, ans = 0;
cin >> N >> M >> K;
vector<vector<int>> matrix(N, vector<int>(M, 0));
vector<vector<bool>> vist(N, vector<bool>(M, false));
// 0 墙 1 空地 2 人
for (int i = 0; i < N; i++) {
    cin >> tmp;
    for (int j = 0; j < M; j++) {
        if (tmp[j] == '.')  matrix[i][j] = 1;
        if (tmp[j] == '#') matrix[i][j] = 0;
        if (tmp[j] == 'E') matrix[i][j] = 2;
    }
}
for (int i = 0; i < N; i++) {
    for (int j = 0; j < M; j++) {
        if (!vist[i][j]&&matrix[i][j]!=0) {
            if (findenemy(i, j, matrix, vist) < K) ans++;
        } else continue;
    }
}
cout << ans << endl;
```

}