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

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

```
# import sys

# for line in sys.stdin:
#         a = line.split()
#         print(int(a[0]) + int(a[1]))


def readIn ():
    n , m , k = map(int , input().split())
```

```
    s = [input() for i in range(n)]
    return n , m , k , s
ans , res = 0 , 0
dx = [0 , 0 , 1 , -1]
dy = [1 , -1 , 0 , 0]
bk = [[False for i in range(105)] for j in range(105)]
n , m , k , s = readIn()
cnt = 0
for i in range(n):
    cnt += i
def dfs (x , y):
    global ans , n , m , res
    bk[x][y] = True
    if s[x][y] == 'E':
        res += 1
    for i in range(4):
        nx = x + dx[i]
        ny = y + dy[i]
        if nx < 0 or nx >= n or ny < 0 or ny >= m or s[nx][ny] == '#' or \
            bk[nx][ny]:
            continue
        dfs (nx , ny)
    return
def is_invalid (x , y):
    return s[x][y] == '#' or bk[x][y]
for i in range(n):
    for j in range(m):
        if is_invalid(i , j):
            continue
        res = 0
        dfs(i , j)
        if res < k:
            ans += 1
cnt = 0
for i in range(n):
    cnt += i
print(ans)
```