# 矩阵中的路径

## 题目

请设计一个函数，用来判断在一个矩阵中是否存在一条包含某字符串所有字符的路径。路径可以从矩阵中的任意一个格子开始，每一步可以在矩阵中向左，向右，向上，向下移动一个格子。如果一条路径经过了矩阵中的某一个格子，则之后不能再次进入这个格子。 例如 a b c e s f c s a d e e 这样的 3 X 4 矩阵中包含一条字符串"bcced"的路径，但是矩阵中不包含"abcb"路径，因为字符串的第一个字符b占据了矩阵中的第一行第二个格子之后，路径不能再次进入该格子。

## 解题思路

1. 简单的回溯查找

static int[][] steps = {{0, 1}, {1, 0}, {0, -1}, {-1, 0}};

public boolean hasPath(char[] matrix, int rows, int cols, char[] str) {
 char[][] \_matrix = new char[rows][cols];

 int k = 0;
 for (int i = 0; i < \_matrix.length; i++) {
 for (int j = 0; j < \_matrix[i].length; j++) {
 \_matrix[i][j] = matrix[k++];
 }
 }

 int[][] flag = new int[rows][cols];

 for (int i = 0; i < \_matrix.length; i++) {
 for (int j = 0; j < \_matrix[i].length; j++) {
 if (\_matrix[i][j] == str[0]) {
 if (hasPath(\_matrix, flag, i, j, str, 0)) {
 return true;
 }
 }
 }
 }

 return false;
}

private boolean hasPath(char[][] matrix, int[][] flag, int x, int y, char[] str, int index) {
 if (x < 0 || y < 0) return false;
 if (x >= matrix.length || y >= matrix[0].length) return false;

 if (flag[x][y] == 1) return false;

 boolean subRes = false;
 if (matrix[x][y] == str[index]) {
 if (index == str.length - 1) return true;

 flag[x][y] = 1;

 for (int[] step : steps) {
 subRes |= hasPath(matrix, flag, x + step[0], y + step[1], str, index + 1);
 }

 flag[x][y] = 0;
 }

 return subRes;
}