1212：LETTERS

题目描述

给出一个roe×col的大写字母矩阵，一开始的位置为左上角，你可以向上下左右四个方向移动，并且不能移向曾经经过的字母。问最多可以经过几个字母。

输入格式

第一行，输入字母矩阵行数R和列数S，1≤R,S≤20。

接着输出R行S列字母矩阵。

输出格式

最多能走过的不同字母的个数。

输入样例

```
3 6
HFDFFB
AJHGDH
DGAGEH
```

输出样例

```
6
```

解析

首先，先来判断一下使用哪种搜索。求最多或者最少的路径，一般来说广搜比较适合。但是对于本题，广搜是有些问题的，例如下图所示：

如图所示，我们从蓝色方块出发，肉眼可见最多经过的字母数应该为3，但是，如果我们按照常规的广搜代码来写，那么结果就是2。因为A和E在第一轮搜索时就被标记了已访问，而无法在第二轮被搜索。

因此，想要A和E能在后续的搜索被识别，我们就需要用到回溯算法，因而深搜是更加合适的。

编码

```
#include <bits/stdc++.h>

using namespace std;
//四个移动方向
int const FORWARD_NUM = 4;
//边界
int R, S;
//终点
int ex, ey;
//地图
char Map[21][21];
//记录当前节点是否行走过
bool Vis[21][21];
//字母访问记录
bool Alpha[26];
//最大经过数量
int MaxNum = 0;
//四个个移动方向
int Forward[4][2] = { //八方向，二维（行，列）
        {-1, 0},//上
        {1,  0},//下
        {0,  -1},//左
        {0,  1},//右
};

//判断指定的目标点是否可以行走
bool Check(int x, int y) {
    //1、是否越界
    if (x >= 0 && y >= 0 && x < R && y < S) {
        //2、未访问过
        if (!Vis[x][y]) {
            //3、这个字母没有访问过
            char cur = Map[x][y];
            if (!Alpha[cur - 'A']) {
                return true;
            }
        }
    }
    return false;
}

//更新每个字母的访问状态
void UpdateState(int x, int y, bool value) {
    char cur = Map[x][y];
    Alpha[cur - 'A'] = value;
    Vis[x][y] = value;
}
```

```
//深搜代码
void Dfs(int sx, int sy, int num) {
    MaxNum = max(MaxNum, num);
    for (int i = 0; i < FORWARD_NUM; ++i) {
        //新的xy坐标
       int newX = sx + Forward[i][0];
        int newY = sy + Forward[i][1];
        if (Check(newX, newY)) {
            UpdateState(newX, newY, true);
            //执行深搜
          Dfs(newX, newY, num + 1);
            UpdateState(newX, newY, false);
        }
    }
}

void Read() {
    //读入地图规模
   cin >> R >> S;
    //读入地图
   for (int j = 0; j < R; ++j) {
        cin >> Map[j];
    }
    UpdateState(0, 0, true);
    ex = R - 1;
    ey = S - 1;
    Dfs(0, 0, 1);
}

int main() {
    Read();
    cout << MaxNum;
    return 0;
}
```