包含整数的二维矩阵 M 表示一个图片的灰度。你需要设计一个平滑器来让每一个单元的灰度成为平均灰度 (向下舍入)，平均灰度的计算是周围的8个单元和它本身的值求平均，如果周围的单元格不足八个，则尽可能多的利用它们。

示例 1:

输入:

```
[[1,1,1],
 [1,0,1],
 [1,1,1]]
```

输出:

```
[[0, 0, 0],
 [0, 0, 0],
 [0, 0, 0]]
```

解释:

```
对于点 (0,0), (0,2), (2,0), (2,2): 平均(3/4) = 平均(0.75) = 0
对于点 (0,1), (1,0), (1,2), (2,1): 平均(5/6) = 平均(0.83333333) = 0
对于点 (1,1): 平均(8/9) = 平均(0.88888889) = 0
```

注意:

1. 给定矩阵中的整数范围为 [0, 255]。
2. 矩阵的长和宽的范围均为 [1, 150]。

思路：按理来说可以用线代解题（个人水平差，没想出来），直接用暴力解法，把周围相加除以平均个数。

```
class Solution {
public:
    vector<vector<int>> imageSmoother(vector<vector<int>>& M) {
        int n=M.size();
        int m=M[0].size();
        vector<vector<int>> ans(n,vector<int>(m));
        for(int i=0;i<n;++i){
            for(int j=0;j<m;++j){
                int num=M[i][j];
                int gs=1;
                if(i>0){
                    num+=M[i-1][j];
                    ++gs;
                    if(j>0){
                        num+=M[i-1][j-1];
                        ++gs;
                    }
                    if(j+1<m){
                        num+=M[i-1][j+1];
```

```
                        ++gs;
                    }
                }
                if(j>0){
                    num+=M[i][j-1];
                    ++gs;
                }
                if(j+1<m){
                        num+=M[i][j+1];
                    ++gs;
                }
                if(i+1<n){
                    num+=M[i+1][j];
                    ++gs;
                    if(j>0){
                        num+=M[i+1][j-1];
                        ++gs;
                    }
                    if(j+1<m){
                        num+=M[i+1][j+1];
                        ++gs;
                    }
                }
                ans[i][j]=num/gs;
            }
        }
        return ans;
    }
};
```