

```

1  /*
2  给定一个  $m \times n$  的矩阵，如果一个元素为 0，则将其所在行和列的所有元素都设为 0。请使用原地
   算法。
3
4  示例 1:
5
6  输入:
7  [
8    [1,1,1],
9    [1,0,1],
10   [1,1,1]
11  ]
12  输出:
13  [
14    [1,0,1],
15    [0,0,0],
16    [1,0,1]
17  ]
18  示例 2:
19  输入:
20  [
21    [0,1,2,0],
22    [3,4,5,2],
23    [1,3,1,5]
24  ]
25  输出:
26  [
27    [0,0,0,0],
28    [0,4,5,0],
29    [0,3,1,0]
30  ]
31  进阶:
32      一个直接的解决方案是使用  $O(mn)$  的额外空间，但这并不是一个好的解决方案。
33      一个简单的改进方案是使用  $O(m + n)$  的额外空间，但这仍然不是最好的解决方案。
34      你能想出一个常数空间的解决方案吗？
35
36  来源：力扣（LeetCode）
37  链接：https://leetcode-cn.com/problems/set-matrix-zeroes
38  著作权归领扣网络所有。商业转载请联系官方授权，非商业转载请注明出处。
39  */

```

分析:

- 方法一:遍历矩阵,使用vector<pair<int,int>>保存出现0的位置,然后再遍历*vp*,将对应行列全部置0,空间复杂度 $O(mn)$;
- 方法二:遍历矩阵,使用两个map<int,int>来保存出现过0的行和列,然后再遍历两个*mii*,将对应行列全部置0,空间复杂度 $O(mn)$.
- 方法三:遍历矩阵做标记,这个算法有缺陷,数组中的数不能出现-10000和-20000,空间复杂度 $O(1)$.

方法一:C++_vp11

```
1  class Solution
2  {
3
4      private:
5
6          void setRowZero(vector<vector<int>>& matrix, int row_number, int
cols )
7          {
8              int i = 0 ;
9
10             for( i = 0 ; i < cols ; i++)
11             {
12                 matrix[row_number][i] = 0;
13             }
14         }
15
16
17         void setColZero(vector<vector<int>>& matrix, int col_number, int
rows)
18         {
19             int i = 0;
20             for(i = 0; i < rows;i++)
21             {
22                 matrix[i][col_number] = 0;
23             }
24         }
25
26
27     public:
28         void setZeroes(vector<vector<int>>& matrix)
29         {
30             int m = matrix.size();
31             int n = matrix[0].size();
32             vector<pair<int,int>>  vp11;
33             int i = 0;
34             int j = 0;
35
36             /*1.遍历找0的位置*/
37             for(i = 0 ; i < m ; i++)
38             {
39                 for(j = 0 ; j < n ; j ++ )
40                 {
41                     if(matrix[i][j]==0)
42                     {
43                         vp11.push_back(make_pair(i,j));
44                     }
45                 }
46             }
47
48             /*2.置零*/
49             for(i = 0; i < vp11.size();i++)
50             {
51                 setRowZero(matrix,vp11[i].first,n);
52                 setColZero(matrix,vp11[i].second,m);
53             }
```

```

54     }
55
56 };
57
58
59 /*
60 执行结果:
61 通过
62 显示详情
63 执行用时 :60 ms, 在所有 cpp 提交中击败了74.42% 的用户
64 内存消耗 :11.5 MB, 在所有 cpp 提交中击败了30.16%的用户
65 */

```

方法二:C++_mii

```

1  class Solution
2  {
3
4      private:
5
6          void setRowZero(vector<vector<int>>& matrix, int row_number, int
cols )
7          {
8              int i = 0 ;
9
10             for( i = 0 ; i < cols ; i++)
11             {
12                 matrix[row_number][i] = 0;
13             }
14         }
15
16
17         void setColZero(vector<vector<int>>& matrix, int col_number, int
rows)
18         {
19             int i = 0;
20             for(i = 0; i < rows;i++)
21             {
22                 matrix[i][col_number] = 0;
23             }
24         }
25
26
27     public:
28         void setZeroes(vector<vector<int>>& matrix)
29         {
30             int m = matrix.size();
31             int n = matrix[0].size();
32             //vector<pair<int,int>>  vpii;
33
34             map<int,int> mii_row;
35             map<int,int> mii_col;
36
37             int i = 0;
38             int j = 0;
39

```

```

40      /*1.遍历找0的位置*/
41      for(i = 0 ; i < m ; i++)
42      {
43          for(j = 0 ; j < n ; j ++ )
44          {
45              if(matrix[i][j]==0)
46              {
47                  mii_row[i] = 0;
48                  mii_col[j] = 0;
49              }
50          }
51      }
52  }
53
54      /*2.置零*/
55      map<int, int>::iterator iter = mii_row.begin();
56      while(iter!=mii_row.end())
57      {
58          setRowZero(matrix,iter->first,n);
59          iter++;
60      }
61
62      iter = mii_col.begin();
63      while(iter != mii_col.end())
64      {
65          setColZero(matrix,iter->first,m);
66          iter++;
67      }
68  }
69  };
70
71
72  /*
73  执行结果:
74  通过
75  显示详情
76  执行用时 :56 ms, 在所有 cpp 提交中击败了89.67% 的用户
77  内存消耗 :11.6 MB, 在所有 cpp 提交中击败了11.11%的用户
78  */

```

方法三:C++_标记法(有缺陷)

```

1  class solution
2  {
3      private:
4          void setRowNeg(vector<vector<int>>& matrix, int row_number, int
cols )
5      {
6          int i = 0 ;
7          for( i = 0 ; i < cols ; i++)
8          {
9              if(matrix[row_number][i]==0 || matrix[row_number]
[i]==-20000)
10             {
11                 matrix[row_number][i] = -20000;
12             }

```

```

13         else
14         {
15             matrix[row_number][i] = -10000;
16         }
17     }
18 }
19
20 void setColNeg(vector<vector<int>>& matrix, int col_number, int
rows)
21 {
22     int i = 0;
23     for(i = 0; i < rows;i++)
24     {
25         if(matrix[i][col_number]==0 || matrix[i]
[col_number]==-20000)
26         {
27             matrix[i][col_number] = -20000;
28         }
29         else
30         {
31             matrix[i][col_number] = -10000;
32         }
33     }
34 }
35
36 public:
37 void setZeroes(vector<vector<int>>& matrix)
38 {
39     int m = matrix.size() ;
40     int n = matrix[0].size() ;
41     int i = 0 ;
42     int j = 0 ;
43
44     /*1.遍历找0的位置,并置负数*/
45     for(i = 0 ; i < m ; i++)
46     {
47         for(j = 0 ; j < n ; j++)
48         {
49             if(matrix[i][j]==0 || matrix[i][j]==-20000 )
50             {
51                 setRowNeg(matrix,i,n);
52                 setColNeg(matrix,j,m);
53             }
54         }
55     }
56
57     /*2.挑出负数置0*/
58     for(i = 0 ; i < m ; i++)
59     {
60         for(j = 0 ; j < n ; j++)
61         {
62             if(matrix[i][j]==-10000 || matrix[i][j]==-20000)
63             {
64                 matrix[i][j] = 0;
65             }
66         }
67     }
68 }

```

```
69  };
70  /*
71  执行结果:
72  通过
73  显示详情
74  执行用时 :96 ms, 在所有 cpp 提交中击败了26.06% 的用户
75  内存消耗 :11.4 MB, 在所有 cpp 提交中击败了66.83%的用户
76  */
```

AlimyBreak
2019.11.19