Set matrix zero

Intuition

To traverse through thw whole matrix and if a zero is found then set whole row and column to 0

Approach 1:

Traversing the matrix and if 0 is found then setting it to a negative number cause setting it to zero can hamper values of other row and columns where initially 0 was not present after setting with negative number traversing again through matrix and setting to zero wherever negative number is found. This approach works only if no negative numbers are present in the matrix.

Complexity

- Time complexity:

```
O(N*M)*O(N+M)
```

- Space complexity:

```
O(1)
```

Code

```
class Solution {
public:
    void setRowColumn(vector<vector<int>>& matrix,int m,int n,int r,int c)
    {
        for(int i=0;i<n;i++)
        {
            if(matrix[r][i]!=0)
```

```
                matrix[r][i]=-1;
        }
        for(int j=0;j<m;j++)
        { if(matrix[j][c]!=0)
                matrix[j][c]=-1;
        }
    }
    void setZeroes(vector<vector<int>> &matrix)
    {
        // Write your code here.
        int m=matrix.size(), n=matrix[0].size();
        for(int i=0;i<m;i++)
        {
            for(int j=0;j<n;j++)
            {
                if(matrix[i][j]==0)
                {
                    setRowColumn(matrix,m,n,i,j);
                }
            }
        }
        for(int i=0;i<m;i++)
        {
            for(int j=0;j<n;j++)
            {
                if(matrix[i][j]<0)
                    matrix[i][j]=0;
            }
        }

    }
};
```

Approach 2:

Using dummy arrays to keep track of zeroes. Setting that particular row to 0 in dummy row as well as dummy column arrays wherever zero is found in the cell. Again traversing through the matrix and replacing current value by 0 if current cell dummy row index or dummy column index is set to 0.

Complexity

- Time complexity:

```
O(N*M + N*M)
```

- Space complexity:

```
O(N)+O(M)
```

Code

```
class Solution {
public:
    void setZeroes(vector<vector<int>>& matrix) {
        int m=matrix.size(),n=matrix[0].size();
  vector<int> rows(m,1),cols(n,1);
  for(int i=0;i<m;i++)
  {
    for(int j=0;j<n;j++)
    {
      if(matrix[i][j]==0)
      {
        rows[i]=0;
        cols[j]=0;
      }
    }
  }
  for(int i=0;i<m;i++)
  {
    for(int j=0;j<n;j++)
    {
      if(rows[i]==0 || cols[j]==0)
      {
        matrix[i][j]=0;
      }
    }
  }

  }
};
```

Approach 3:

Using dummy arrays increases space complexity to O(N) so trying to reduce space complexity.

We use the 1st row and 1st column of given array as dummy array but matrix[0][0] coincides so to resolve this we use dummy column from 1 to n and dummy row from 0 to m. so for keeping track of whether dummy row contains any 0 we use a variable col and set it to 0 if a 0 is found in dummy row.

We again traverse the array apart from dummy row and dummy column i.e from i-> 1 to m and j-> 1 to n to find if matrix cell contains any zero if it does then we check whether dummy column or row is set if it is set then we set the matrix cell to zero.

Now we again need to perform similar operation for dummy row as well as dummy column. dummy row depends on matrix [0][0] whereas as dummy column depends on cols variable so we set it accordingly.

Complexity

- Time complexity:

```
O(N*M + N*M)
```

- Space complexity:

```
O(1)
```

Code

```
class Solution {
public:
    void setZeroes(vector<vector<int>>& matrix) {
    bool cols=1;
  int m=matrix.size(),n=matrix[0].size();
  for(int i=0;i<m;i++)
  {
    for(int j=0;j<n;j++)
    {
      if(matrix[i][j]==0)
      {
        matrix[i][0]=0;
        if(j!=0)
          matrix[0][j]=0;
```

```
        else
          cols=0;
      }
    }
  }
  for(int i=1;i<m;i++)
  {
    for(int j=1;j<n;j++)
    {
      if(matrix[i][j]!=0)
      {
                if (matrix[i][0] == 0 || matrix[0][j] == 0)
        {
                    matrix[i][j] = 0;
                }
            }
        }
    }
  if(matrix[0][0]==0)
  {
    for (int i = 0; i < n; i++)
    {
      matrix[0][i] = 0;
    }
    }
  if(cols==0)
  {
    for (int j = 0; j < m; j++)
    {
      matrix[j][0] = 0;
    }
    }
    }
};
```