# 1380. Lucky Numbers in a Matrix

### *Brute force (AC)*

```
/*
   Time complexity=O(m*n*(m+n))
   Extra space complexity: O(1)
*/
class Solution {
public:
   vector<int> luckyNumbers (vector<vector<int>>& matrix){
      int m=matrix.size();
      int n=matrix[0].size();

      std::vector<int> ans;

      for(int i=0;i<m;++i){
         for(int j=0;j<n;++j){
            int v=matrix[i][j];

            int mi=INT_MAX;
            for(int k=0;k<n;++k){
               if(mi>matrix[i][k]) mi=matrix[i][k];
            }

            int mx=INT_MIN;
            for(int k=0;k<m;++k){
               if(mx<matrix[k][j]) mx=matrix[k][j];
            }

            if(v==mi&&v==mx) return v;
         }
      }
      return {};
   }
};
```

# 1380. Lucky Numbers in a Matrix

### *Preprocessing (AC)*

```
/*
   Time complexity=O(2*(m*n))
   Extra space complexity: O(m+n)
*/
class Solution {
public:
   vector<int> luckyNumbers (vector<vector<int>>& matrix){
      int m=matrix.size();
      int n=matrix[0].size();

      std::vector<int> min_in_row(m,INT_MAX);
      std::vector<int> max_in_col(n,INT_MIN);

      for(int i=0;i<m;++i){
         for(int j=0;j<n;++j){
            min_in_row[i]=std::min(min_in_row[i],matrix[i][j]);
            max_in_col[j]=std::max(max_in_col[j],matrix[i][j]);
         }
      }

      std::vector<int> ans;

      for(int i=0;i<m;++i){
         for(int j=0;j<n;++j){
            int mi=min_in_row[i];
            int mx=max_in_col[j];
            if(mi==mx) return mi;

         }
      }
      return {};
   }
};
```

# 1380. Lucky Numbers in a Matrix

### *Optimal: needs a close observation(AC)*

```
class Solution {
public:
    vector<int> luckyNumbers (vector<vector<int>>& matrix){
        int m=matrix.size();
        int n=matrix[0].size();

        for(int i=0;i<m;++i){
            int mi=INT_MAX,col_min_index=-1;
            for(int j=0;j<n;++j){
                if(mi>matrix[i][j]){
                    mi=matrix[i][j];
                    col_min_index=j;
                }
            }

            int mx=INT_MIN,row_max_index=-1;
            for(int j=0;j<m;++j){
                if(mx<matrix[j][col_min_index]){
                    mx=matrix[j][col_min_index];
                    row_max_index=j;
                }
            }

            if(i==row_max_index) return {matrix[i][col_min_index]};
        }

        return {};
    }
};
```