- ## 225. Implement Stack using Queues

Hint:

1. Use ArrayList

```
class MyStack {
    List<Integer> list;
    /** Initialize your data structure here. */
    public MyStack() {
        list = new ArrayList<>();
    }

    /** Push element x onto stack. */
    public void push(int x) {
        list.add(x);
    }

    /** Removes the element on top of the stack and returns that element. */
    public int pop() {
        int temp = list.get(list.size() - 1);
        list.remove(list.size() - 1);
        return temp;
    }

    /** Get the top element. */
    public int top() {
        return list.get(list.size() - 1);
    }

    /** Returns whether the stack is empty. */
    public boolean empty() {
        return list.isEmpty();
    }
}
```

- ## 73. Set Matrix Zeroes

Hint:

1. **Space O(m+n)方法，用两个set记录需要改变为0的行和列index**
2. **分别将set中的行和列设为0**

```
class Solution {
    public void setZeroes(int[][] matrix) {
        Set<Integer> rowlist = new HashSet<>();
        Set<Integer> collist = new HashSet<>();
        for(int i = 0; i < matrix.length; i++ ){
            for(int j = 0; j < matrix[0].length; j++){
                if(matrix[i][j] == 0){
                    rowlist.add(i);
                    collist.add(j);
                }
            }
        }
        for(Integer row: rowlist){
            for(int i = 0; i < matrix[row].length; i++){
                matrix[row][i] = 0;
            }
        }
        for(Integer col: collist){
            for(int i = 0; i < matrix.length; i++){
                matrix[i][col] = 0;
            }
        }
    }
}
```

- ## 175. Combine Two Tables

SQL Hint：

1. **Left Join**

- ## 150. Evaluate Reverse Polish Notation

```
class Solution {
    public int evalRPN(String[] tokens) {
        Stack<Integer> stack = new Stack<>();
        for(String temp: tokens){
            if(temp.equals("+")){
                stack.push(stack.pop() + stack.pop());
            }
            else if(temp.equals("-")){
                int first = stack.pop();
                stack.push(stack.pop() - first);
            }
            else if(temp.equals("*")){
                stack.push(stack.pop() * stack.pop());
            }
            else if(temp.equals("/")){
                int first = stack.pop();
                stack.push(stack.pop() / first);
            }
            else{
                stack.push(Integer.parseInt(temp));
            }
        }
        return stack.pop();
    }
}
```