**一共四个题（DFS），其中1、2题思路和代码基本一样**

|  |  |
| --- | --- |
| 使用沉岛思想：遍历二维矩阵，发现有岛屿，就从该岛屿开始进行DFS，即从该岛屿的上下左右去搜索，某一个岛屿如果已经搜索过，就将其沉掉（改变原来的值’1’）        思路：也是沉岛法思想，遍历二维矩阵，遇到岛屿就将结果加1，并从该岛屿进行DFS，把DFS遇到的岛屿值改变。          思路：类似于沉岛法，因为需要找被包围的边界，将其设置为’X’，我们可以从二维数组的的四个边界去遍历，如果发现有’O’,将其进行标记（通过改变其为特定的字符，比如’\*’）,并从这个位置开始进行DFS，将DFS到的‘O’都进行标记。当从二维矩阵的四边都遍历完了，在二维矩阵种剩余的‘O’说明就是被X包围的，将其值修改为X即可，最后把标记为\*的字符修改为’O’即得到最后结果   |  | | --- | | class Solution {      public void solve(char[][] board) {          if(board.length == 0) {              return;          }          int row = board.length;          int col = board[0].length;          //从矩阵的上下边界进行DFS          for(int i = 0; i < col; i ++) {              if(board[0][i] == 'O'){                  findIs(0,i,board);              }              if(board[row - 1][i] == 'O') {                  findIs(row - 1,i,board);              }          }          //从矩阵的左右边界进行DFS          for(int i = 0; i < row; i ++) {              if(board[i][0] == 'O') {                  findIs(i,0,board);              }              if(board[i][col - 1] == 'O') {                  findIs(i,col - 1,board);              }          }          //最后遍历DFS完的二维数组，改变值          for(int i = 0; i < board.length; i ++) {              for(int j = 0; j < board[0].length; j ++) {                  if(board[i][j] == 'O'){                      board[i][j] = 'X';                  }                  if(board[i][j] == '\*') {                      board[i][j] = 'O';                  }              }          }      }      //进行DFS      private void findIs(int row,int col, char[][] board) {          if(row < 0 || row >= board.length || col < 0 || col >= board[0].length  || board[row][col] == 'X' || board[row][col] == '\*') {              return;          }          if(board[row][col] == 'O'){              board[row][col] = '\*';          }          findIs(row - 1, col, board);          findIs(row + 1, col, board);          findIs(row, col - 1, board);          findIs(row, col + 1, board);      }  } |         （要理解二维矩阵的行和列代表什么）  思路：使用一个大小为N的数组记录该位置代表的人是否已经形成过朋友圈，默认值是false,都还没形成，  从第一个人开始去DFS，如果找到与第一个人有联系，将找到的这个人对应的数组值改为true(改为true,说明已经参与形成过朋友圈，下次遍历到这个人的时候，发现记录该位置的数组为true,直接跳过，不进行朋友圈计数) |