一、题目说明

题目200. Number of Islands，在一个0（代表水）和1（代表陆地）组成的2d地图中，计数“岛屿”的数量。

二、我的解答

本题目计算图的最大连通分量，可以用图的深度优先遍历，也可用图的广度优先遍历。

深度优先遍历算法，代码如下：

```
class Solution {
    public:
    //图的深度优先遍历
    void dfs(vector<vector<char>>& grid,int i,int j,int row,int col){
        if(grid[i][j]=='1'){
            grid[i][j] = '.';
            //向左
            if(j>0){
                dfs(grid,i,j-1,row,col);
            }
            //向右
            if(j<col-1){
                dfs(grid,i,j+1,row,col);
            }
            //向下
            if(i<row-1){
                dfs(grid,i+1,j,row,col);
            }
            //向上
            if(i>0){
                dfs(grid,i-1,j,row,col);
            }

        }
        return;
    }
    int numIslands(vector<vector<char>>& grid) {
        if(grid.size()<1 || grid[0].size()<1) return 0;

        count = 0;
        int row,col;
        row = grid.size();
        col = grid[0].size();

        for(int i=0;i<row;i++){
            for(int j=0;j<col;j++){
                if(grid[i][j]=='1'){
                    count++;
                    //cout<<"in for:i="<<i<<",j="<<j<<"\n";
                    dfs(grid,i,j,row,col);
                }
            }
        }
        return count;
    }
```

```
    private:
        int count;
};
```

性能:

```
Runtime: 12 ms, faster than 92.90% of C++ online submissions for Number of
Islands.
Memory Usage: 10.8 MB, less than 85.39% of C++ online submissions for Number of
Islands.
```

三、优化措施

用广度优先遍历算法，需要一个栈，或者队列。

```
class Solution {
    public:
    //图的广度优先遍历
    int numIslands(vector<vector<char>>& grid) {
        if(grid.size()<1 || grid[0].size()<1) return 0;

        count = 0;
        int row,col;
        row = grid.size();
        col = grid[0].size();

        for(int i=0;i<row;i++){
            for(int j=0;j<col;j++){
                if(grid[i][j]=='1'){
                    count++;
                    grid[i][j] = '.';
                    q.push(pair<int,int>(i,j));
                    while(! q.empty()){
                        pair<int,int> p = q.front();
                        q.pop();
                        int iTmp = p.first;
                        int jTmp = p.second;
                        //左
                        if(jTmp>0 && grid[iTmp][jTmp-1] == '1'){
                            grid[iTmp][jTmp-1] = '.';
                            q.push(pair<int,int>(iTmp,jTmp-1));
                        }
                        //右
                        if(jTmp+1<col && grid[iTmp][jTmp+1] == '1'){
                            grid[iTmp][jTmp+1] = '.';
                            q.push(pair<int,int>(iTmp,jTmp+1));
                        }
                        //上
                        if(iTmp>0 && grid[iTmp-1][jTmp] == '1'){
                            grid[iTmp-1][jTmp] = '.';
                            q.push(pair<int,int>(iTmp-1,jTmp));
                        }
                        //下
```

```
                        if(iTmp+1<row && grid[iTmp+1][jTmp] == '1'){
                            grid[iTmp+1][jTmp] = '.';
                            q.push(pair<int,int>(iTmp+1,jTmp));
                        }
                    }
                }
            }
        }

        return count;
    }
    private:
        int count;
        queue<pair<int,int>> q;
};
```

性能如下：

```
Runtime: 16 ms, faster than 63.82% of C++ online submissions for Number of
Islands.
Memory Usage: 11 MB, less than 47.19% of C++ online submissions for Number of
Islands.
```