/\*

Given a 2d grid map of '1's (land) and '0's (water), count the number of islands. An island is surrounded by water and is formed by connecting adjacent lands horizontally or vertically. You may assume all four edges of the grid are all surrounded by water.

Example 1:

11110

11010

11000

00000

Answer: 1

Example 2:

11000

11000

00100

00011

Answer: 3

思路就是左上角开始找点，如果是1，就广度遍历（BFS）或者深度遍历（DFS）这个island的点，全部设为2。

然后ret++;

\*/

struct POS

{

int x;

int y;

POS(int xx,int yy):x(xx),y(yy){}

};

class Solution {

public:

void BFS(vector<vector<char>>& grid,int i,int j,int row,int col)

{

queue<POS\*> pos;

POS \*a=new POS(i,j);

pos.push(a);

grid[i][j]='2';

while(!pos.empty())

{

POS\* b=pos.front();

pos.pop();

if(b->x>0 && grid[b->x-1][b->y]=='1')

{

POS \*c=new POS(b->x-1,b->y);

pos.push(c);

grid[b->x-1][b->y]='2';

}

if(b->x<row-1 && grid[b->x+1][b->y]=='1')

{

POS \*c=new POS(b->x+1,b->y);

pos.push(c);

grid[b->x+1][b->y]='2';

}

if(b->y>0 && grid[b->x][b->y-1]=='1')

{

POS \*c=new POS(b->x,b->y-1);

pos.push(c);

grid[b->x][b->y-1]='2';

}

if(b->y<col-1 && grid[b->x][b->y+1]=='1')

{

POS \*c=new POS(b->x,b->y+1);

pos.push(c);

grid[b->x][b->y+1]='2';

}

}

}

void DFS(vector<vector<char>>& grid,int i,int j,int row,int col)

{

stack<POS\*> pos;

POS\* a=new POS(i,j);

pos.push(a);

grid[i][j]='2';

while(!pos.empty())

{

POS \*b=pos.top();

if(b->x>0 && grid[b->x-1][b->y]=='1')

{

POS \*c=new POS(b->x-1,b->y);

pos.push(c);

grid[b->x-1][b->y]='2';

continue;

}

if(b->x<row-1 && grid[b->x+1][b->y]=='1')

{

POS \*c=new POS(b->x+1,b->y);

pos.push(c);

grid[b->x+1][b->y]='2';

continue;

}

if(b->y>0 && grid[b->x][b->y-1]=='1')

{

POS \*c=new POS(b->x,b->y-1);

pos.push(c);

grid[b->x][b->y-1]='2';

continue;

}

if(b->y<col-1 && grid[b->x][b->y+1]=='1')

{

POS \*c=new POS(b->x,b->y+1);

pos.push(c);

grid[b->x][b->y+1]='2';

continue;

}

pos.pop();

}

}

int numIslands(vector<vector<char>>& grid)

{

int ret=0;

int row=grid.size();

if(row==0)

return ret;

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

for(int i=0;i<row;i++)

{

for(int j=0;j<col;j++)

{

if(grid[i][j]=='1')

{

ret++;

//BFS(grid,i,j,row,col);

DFS(grid,i,j,row,col);

}

}

}

return ret;

}

};