1. Number of Islands

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:**

Input: grid = [  
 ["1","1","1","1","0"],  
 ["1","1","0","1","0"],  
 ["1","1","0","0","0"],  
 ["0","0","0","0","0"]  
]  
Output: 1

**Example 2:**

Input: grid = [  
 ["1","1","0","0","0"],  
 ["1","1","0","0","0"],  
 ["0","0","1","0","0"],  
 ["0","0","0","1","1"]  
]  
Output: 3

**解1** bfs

class Solution {  
public:  
 int dx[4] = {-1, 1, 0, 0};  
 int dy[4] = {0, 0, -1, 1};  
 bool valid(int x, int y, int m, int n){  
 if(x < 0 || x >= m || y < 0 || y >= n)return false;  
 return true;  
 }  
 int numIslands(vector<vector<char>>& grid) {  
 if(grid.size() == 0)return 0;  
 vector<vector<bool>> vis(grid.size(), vector<bool>(grid[0].size(), false));  
 int ans = 0;  
 for(int i = 0; i < grid.size(); ++i){  
 for(int j = 0; j < grid[0].size(); ++j){  
 if(vis[i][j] == false && grid[i][j] == '1'){  
 ans++;  
 bfs(grid, vis, i, j);  
 //dfs(grid, vis, i, j);  
 }  
 }  
 }  
 return ans;  
 }  
 void bfs(vector<vector<char>>& grid, vector<vector<bool>>& vis,   
 int x, int y){  
 queue<pair<int, int>>q;  
 q.push(make\_pair(x,y));  
 vis[x][y] = true;  
 while(!q.empty()){  
 int tmpx = q.front().first, tmpy = q.front().second;  
 q.pop();  
 for(int i = 0; i < 4; ++i){  
 int tmpxx = tmpx + dx[i], tmpyy = tmpy + dy[i];  
 if(valid(tmpxx, tmpyy, grid.size(), grid[0].size())  
 && !vis[tmpxx][tmpyy] && grid[tmpxx][tmpyy]=='1'){  
 q.push(make\_pair(tmpxx, tmpyy));  
 vis[tmpxx][tmpyy] = true;  
 }  
 }  
 }  
 }  
};

**解2** dfs

void dfs(vector<vector<char>>& grid, vector<vector<bool>>& vis,   
 int x, int y){  
 vis[x][y] = true;  
 for(int i = 0; i < 4; ++i){  
 int tmpx = x + dx[i], tmpy = y + dy[i];  
 if(valid(tmpx, tmpy, grid.size(), grid[0].size())  
 && !vis[tmpx][tmpy] && grid[tmpx][tmpy] == '1'){  
 dfs(grid, vis, tmpx, tmpy);  
 }  
 }  
 }