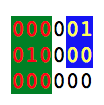
**DFS Connected Cell in a Grid**

Consider a matrix where each cell contains either a 0 or a 1 and any cell containing a 1 is called a *filled* cell. Two cells are said to be *connected* if they are adjacent to each other horizontally, vertically, or diagonally. In the diagram below, the two colored regions show cells connected to the filled cells. Black on white are not connected.

Cells adjacent to filled cells:

If one or more filled cells are also connected, they form a *region*. Note that each cell in a region is connected to at least one other cell in the region but is not necessarily directly connected to all the other cells in the region.

Regions:

Given an n\*m  matrix, find and print the number of cells in the largest *region* in the matrix.

**Function Description**

Complete the function *maxRegion* in the editor below. It must return an integer value, the size of the largest region.

maxRegion has the following parameter(s):

* *grid*: a two dimensional array of integers

**Input Format**

The first line contains an integer, n, the number of rows in the matrix, grid.   
The second line contains an integer,m , the number of columns in the matrix.

Each of the following n lines contains a row of m space-separated integers, grid[i][j].

**Constraints**

* 0<=n,m<=10
* Grid[i][j]<-{0,1}

**Output Format**

Print the number of cells in the largest *region* in the given matrix.

**Sample Input**

4

4

1 1 0 0

0 1 1 0

0 0 1 0

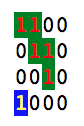
1 0 0 0

**Sample Output**

5

**Explanation**

The diagram below depicts two regions of the matrix:

  
The first region has five cells and the second region has one cell. We choose the larger region.

using System.CodeDom.Compiler;

using System.Collections.Generic;

using System.Collections;

using System.ComponentModel;

using System.Diagnostics.CodeAnalysis;

using System.Globalization;

using System.IO;

using System.Linq;

using System.Reflection;

using System.Runtime.Serialization;

using System.Text.RegularExpressions;

using System.Text;

using System;

using System.Windows;

using System.Numerics;

class Point1{

public int x;

public int y;

}

class Solution {

// Complete the maxRegion function below.

static int maxRegion(int[][] grid) {

//use DFS to search

Stack<Point1> stack = new Stack<Point1>();

//put flag in the gris

bool[,] visited = new bool[grid.Length, grid[0].Length];

int maxCount = 0;

int tempCount = 1;

//save point==1 in the stack

for(int i = 0; i < grid.Length; i++){

for(int j = 0; j < grid[0].Length; j++){

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

stack.Push(null);

stack.Push(new Point1() {x=i,y=j});

}

}

}

// base on pop point check 8 directs point {-1,-1},{-1,0},{-1,1},{0,-1},{0,1}

//{1,-1},{1,0},{1,1}

int[] dr = {1,-1,0,0,-1,1,-1,1};

int[] dc = {0,0,-1,1,-1,1,1,-1};

while(stack.Count > 0){

Point1 p = stack.Pop();

if(p == null){

maxCount = Math.Max(tempCount, maxCount);

tempCount = 1;

continue;

}

//set cell to visited continue to traversal the grid

visited[p.x,p.y] = true;

for(int i = 0; i < dr.Length; i++){

//if dr[i]=-1 check above line; dr[i]=1 check under line base on pop point

int newX = p.x + dr[i];

int newY = p.y + dc[i];

if(newX < 0 || newY < 0) continue;

if(newX >= grid.Length || newY >= grid[0].Length) continue;

if(visited[newX, newY]) continue;

if(grid[newX][newY] == 0) continue;

//find cell =1 && visited=false

visited[newX, newY] = true;

//next to check

stack.Push(new Point1() {x=newX,y=newY});

tempCount++;

}

}

return maxCount;

}

static void Main(string[] args) {

TextWriter textWriter = new StreamWriter(@System.Environment.GetEnvironmentVariable("OUTPUT\_PATH"), true);

int n = Convert.ToInt32(Console.ReadLine());

int m = Convert.ToInt32(Console.ReadLine());

int[][] grid = new int[n][];

for (int i = 0; i < n; i++) {

grid[i] = Array.ConvertAll(Console.ReadLine().Split(' '), gridTemp => Convert.ToInt32(gridTemp));

}

int res = maxRegion(grid);

textWriter.WriteLine(res);

textWriter.Flush();

textWriter.Close();

}

}

**Congratulations**

You solved this challenge. Would you like to challenge your friends?

[Next Challenge](https://www.hackerrank.com/challenges/matrix?h_l=interview&playlist_slugs%5B%5D=interview-preparation-kit&playlist_slugs%5B%5D=graphs&h_r=next-challenge&h_v=zen)

* **Test case 0**
* **Test case 1**
* **Test case 2**
* **Test case 3**
* **Test case 4**
* **Test case 5**
* **Test case 6**
* **Test case 7**

Compiler Message

**Success**

Input (stdin)

Download

* **4**
* **4**
* **1 1 0 0**
* **0 1 1 0**
* **0 0 1 0**
* **1 0 0 0**

Expected Output

Download

* **5**