Let's try Minesweeper!  
Each cell of Minesweeper gameboard can be:

* a mine (appears as 9)
* or a number representing the number of mines in its *surrounding* cells  
  (a cell is considered as *surrounding* another cell when this cell meets that cell on at least 1 corner) (appears as 0 - 8)

Your task is to check whether a gameboard is a valid gameboard.

**Example**  
For

gameboard = [[0, 1, 9, 1],

[0, 1, 1, 1],

[0, 0, 0, 0]]

the output should be  
minesweeper1(gameboard) = true.

**Input/Output**

* **[time limit] 3000ms (cs)**
* **[input] array.array.integer gameboard**

*Guaranteed constraints:*  
3 ≤ gameboard.length ≤ 100,  
3 ≤ gameboard[i].length ≤ 100,  
gameboard[i].length = gameboard[j].length,  
0 ≤ gameboard[i][j] ≤ 9.

* **[output] boolean**

<https://codefights.com/challenge/ZXRv42mfhbiTTDrEB/solutions>

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace ConsoleApplication1

{

class Program

{

static bool minesweeper1(int[][] gameboard)

{

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

{

for (int j = 0; j < gameboard[i].Length; j++)

{

if (gameboard[i][j] != 9)

{

int cont = 0;

if (i - 1 >= 0)

{

if (gameboard[i - 1][j] == 9)

{

cont++;

}

}

if (i + 1 < gameboard.Length)

{

if (gameboard[i + 1][j] == 9)

{

cont++;

}

}

if (j - 1 >= 0)

{

if (gameboard[i][j - 1] == 9)

{

cont++;

}

}

if (j + 1 < gameboard[i].Length)

{

if (gameboard[i][j + 1] == 9)

{

cont++;

}

}

if (i - 1 >= 0 && j - 1 >= 0)

{

if (gameboard[i - 1][j - 1] == 9)

{

cont++;

}

}

if (i - 1 >= 0 && j + 1 < gameboard[i].Length)

{

if (gameboard[i - 1][j + 1] == 9)

{

cont++;

}

}

if (i + 1 < gameboard.Length && j - 1 >= 0)

{

if (gameboard[i + 1][j - 1] == 9)

{

cont++;

}

}

if (i + 1 < gameboard.Length && j + 1 < gameboard[i].Length)

{

if (gameboard[i + 1][j + 1] == 9)

{

cont++;

}

}

if (cont != gameboard[i][j])

{

return false;

}

}

}

}

return true;

}

static void Main(string[] args)

{

}

}

}