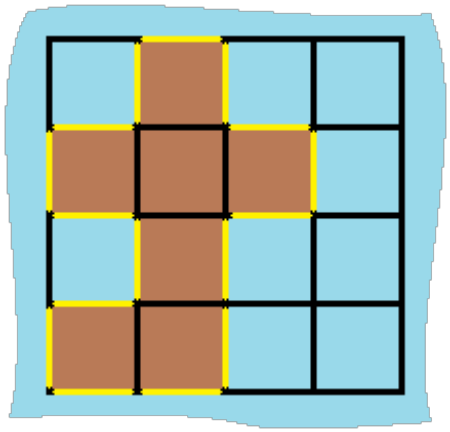
Island Perimeter

You are given a map in form of a two-dimensional integer grid where 1 represents land and 0 represents water. Grid cells are connected horizontally/vertically (not diagonally). The grid is completely surrounded by water, and there is exactly one island (i.e., one or more connected land cells). The island doesn't have "lakes" (water inside that isn't connected to the water around the island). One cell is a square with side length 1. The grid is rectangular, width and height don't exceed 100. Determine the perimeter of the island.

Example:

```
[[0,1,0,0],
 [1,1,1,0],
 [0,1,0,0],
 [1,1,0,0]]

Answer: 16
Explanation: The perimeter is the 16 yellow stripes in the image below:
```



Solution 1

1. loop over the matrix and count the number of islands;
2. if the current dot is an island, count if it has any right neighbour or down neighbour;
3. the result is islands * 4 - neighbours * 2

```
public class Solution {
    public int islandPerimeter(int[][] grid) {
        int islands = 0, neighbours = 0;

        for (int i = 0; i < grid.length; i++) {
            for (int j = 0; j < grid[i].length; j++) {
                if (grid[i][j] == 1) {
                    islands++; // count islands
                    if (i < grid.length - 1 && grid[i + 1][j] == 1) neighbours++;
// count down neighbours
                    if (j < grid[i].length - 1 && grid[i][j + 1] == 1) neighbours
++; // count right neighbours
                }
            }
        }

        return islands * 4 - neighbours * 2;
    }
}
```

written by liupangzi original link here

Solution 2

1. find how many 1 in the map. If without the consideration of surrounding cells, the total perimeter should be the total amount of 1 times 4.
2. find how many cell walls that connect with both lands. We need to deduct twice of those lines from total perimeter

```
int islandPerimeter(vector<vector<int>>& grid) {
        int count=0, repeat=0;
        for(int i=0;i<grid.size();i++)
        {
            for(int j=0; j<grid[i].size();j++)
                {
                    if(grid[i][j]==1)
                    {
                        count ++;
                        if(i!=0 && grid[i-1][j] == 1) repeat++;
                        if(j!=0 && grid[i][j-1] == 1) repeat++;
                    }
                }
        }
        return 4*count-repeat*2;
    }
```

@msg thanks for the edit

written by beckswu original link here

Solution 3

```
public static int islandPerimeter(int[][] grid) {
        if (grid == null || grid.length == 0 || grid[0].length == 0) return 0;
        int result = 0;
        for (int i = 0; i < grid.length; i++) {
            for (int j = 0; j < grid[0].length; j++) {
                if (grid[i][j] == 1) {
                    result += 4;
                    if (i > 0 && grid[i-1][j] == 1) result -= 2;
                    if (j > 0 && grid[i][j-1] == 1) result -= 2;
                }
            }
        }
        return result;
    }
```

written by dreamchase original link here