463 Island Perimeter

2018年4月12日 11:37

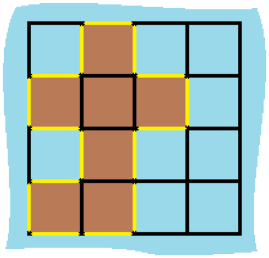
You are given a map in form of a two-dimensional integer grid where 1 represents land and 0 represents water. Grid cells are connected horizontally/vertically (not diagonally). The grid is completely surrounded by water, and there is exactly one island (i.e., one or more connected land cells). The island 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:



来自 <https://leetcode.com/problems/island-perimeter/description/>

给定一个包含 0 和 1 的二维网格地图，其中 1 表示陆地 0 表示水域。网格中的格子水平和垂直方向相连（对角线方向不相连）。整个网格被水完全包围，但其中恰好有一个岛屿（或者说，一个或多个表示陆地的格子相连组成的岛屿）。岛屿中没有“湖”（“湖” 指水域在岛屿内部且不和岛屿周围的水相连）。格子是边长为 1 的正方形。网格为长方形，且宽度和高度均不超过 100 。计算这个岛屿的周长。

Solution for Python3:

```python
class Solution1:
    def islandPerimeter(self, grid):
        """
        :type grid: List[List[int]]
        :rtype: int
        """
        cnt, repeat = 0, 0
        for i in range(len(grid)):
           for j in range(len(grid[i])):
               if grid[i][j]:
                   cnt += 1
                   if i and grid[i-1][j]:
                       repeat += 1
                   if j and grid[i][j-1]:
                       repeat += 1
        return 4 * cnt - 2 * repeat

class Solution2:
    def islandPerimeter(self, grid):
        """
        :type grid: List[List[int]]
        :rtype: int
        """
        grid_ext = ['0' + ''.join(str(x) for x in row) + '0' for row in grid]
        grid_trans = list(map(list, zip(*grid)))
```

```
        grid_ext += ['0' + ''.join(str(x) for x in row) + '0' for row in grid_trans]
        return sum(row.count('01') + row.count('10') for row in grid_ext)

class Solution3:
    def islandPerimeter(self, grid):
        """
        :type grid: List[List[int]]
        :rtype: int
        """
        return sum(sum(map(operator.ne, [0] + row, row + [0]))
           for row in grid + list(map(list, zip(*grid))))
```

Solution for C++:

```
class Solution {
public:
    int islandPerimeter(vector<vector<int>>& grid) {
        int cnt = 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) {
                    cnt++;
                    if (i != 0 && grid[i-1][j] == 1) repeat++;
                    if (j != 0 && grid[i][j-1] == 1) repeat++;
                }
            }
        }
        return 4 * cnt - 2 * repeat;
    }
};
```

Appendix:

两个思路:

1) 计算所有为1的个数cnt以及每行和每列重叠的边repeat。然后4 * cnt - 2 * repeat。
2) 统计每行每列相邻不同的次数，其中若统计每行相邻元素不同的次数可以采用[0]+row与row+[0]对应相比是否不同。

Python zip(iterable,...):

1) 字符串s='abcabcabc'为含有重复子字符串的字符串。
   zip() 函数用于将可迭代的对象作为参数，将对象中对应的元素打包成一个个元组，然后返回由这些元组组成的列表。如果各个迭代器的元素个数不一致，则返回列表长度与最短的对象相同，利用 * 号操作符，可以将元组解压为列表。
2) >>>a = [1,2,3]
3) >>> b = [4,5,6]
4) >>> c = [4,5,6,7,8]
5) >>> zipped = zip(a,b) # 打包为元组的列表 [(1, 4), (2, 5), (3, 6)]
6) >>> zip(a,c) # 元素个数与最短的列表一致 [(1, 4), (2, 5), (3, 6)]
7) >>> zip(*zipped) # 与 zip 相反，可理解为解压，返回二维矩阵式 [(1, 2, 3), (4, 5, 6)]
8) 其中对于形容L=[[1,2,3],[4,5,6]]矩阵的转置可以采用解压方式：
   a. L' = zip(*L)=[[1,4],[2,5],[3,6]]

Python operator库:

操作	语法	函数
相加	a + b	add(a, b)
字符串拼接	seq1 + seq2	concat(seq1, seq2)
包含测试	obj in seq	contains(seq, obj)
普通除法	a / b	truediv(a, b)
取整除法	a // b	floordiv(a, b)
按位与	a & b	and_(a, b)
按位异或	a ^ b	xor(a, b)
按位取反	~ a	invert(a)
按位或	a | b	or_(a, b)
指数运算	a ** b	pow(a, b)
识别	a is b	is_(a, b)
识别	a is not b	is_not(a, b)
索引赋值	obj[k] = v	setitem(obj, k, v)
索引删除	del obj[k]	delitem(obj, k)
索引	obj[k]	getitem(obj, k)
左移	a << b	lshift(a, b)
取模	a % b	mod(a, b)
乘法	a * b	mul(a, b)
负数	-a	neg(a)
非运算	not a	not_(a)
正数	+ a	pos(a)
右移运算	a >> b	rshift(a, b)
切片赋值	seq[i:j] = values	setitem(seq, slice(i, j), values)
切片删除	del seq[i:j]	delitem(seq, slice(i, j))
切片	seq[i: j]	getitem(seq, slice(i, j))
字符串格式化	s % obj	mod(s, obj)
减法	a - b	sub(a, b)
真值测试	obj	truth(obj)
小于	a < b	lt(a, b)
小于等于	a <= b	le(a, b)
等于	a == b	eq(a, b)
不等于	a != b	ne(a, b)
大于等于	a >= b	ge(a, b)
大于	a > b	gt(a, b)

来自 <https://blog.csdn.net/shengmingqijiquan/article/details/53005129>