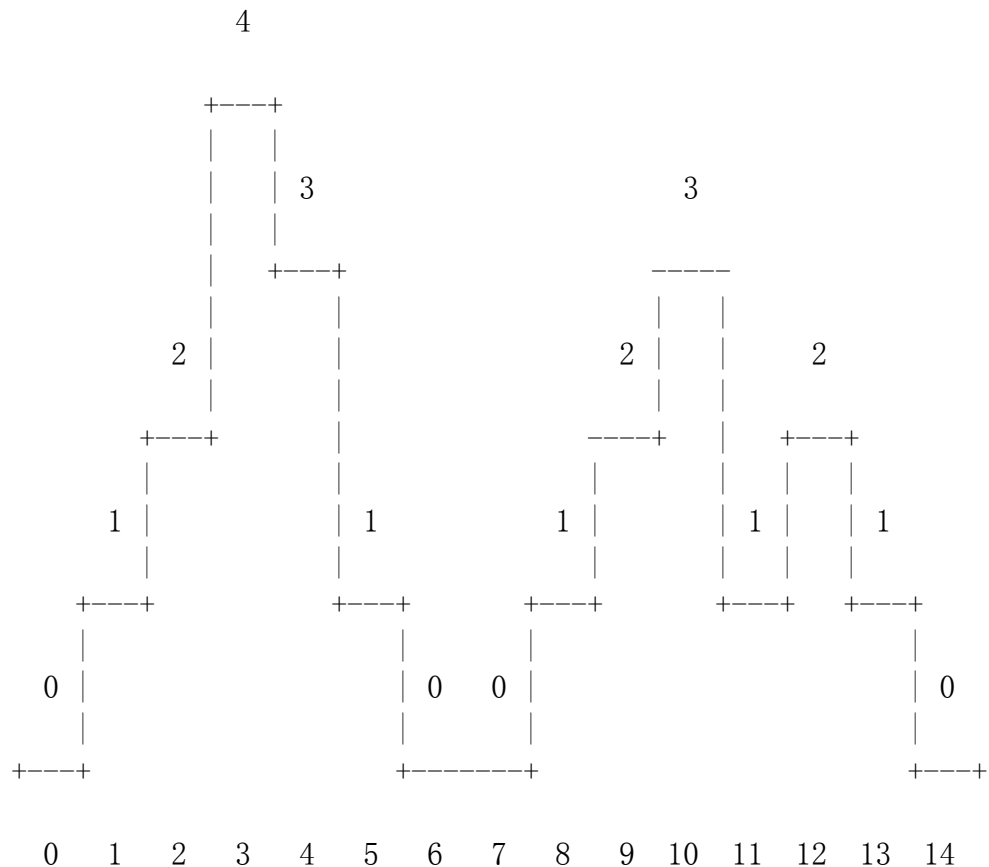


攀登者喜欢寻找各种地图，并且尝试攀登到最高的山峰。
地图表示为一维数组，数组的索引代表水平位置，数组的高度代表相对海拔高度。
其中数组元素 0 代表地面。
例如 [0, 1, 2, 4, 3, 1, 0, 0, 1, 2, 3, 1, 2, 1, 0]，代表如下图所示的地图，地图中有两个山脉位置分别为 1, 2, 3, 4, 5 和 8, 9, 10, 11, 12, 13，最高峰高度分别为 4, 3。最高峰位置分别为 3, 10。
一个山脉可能有多座山峰(高度大于相邻位置的高度，或在地图边界且高度大于相邻的高度)。



登山时会消耗登山者的体力(整数)，上山时，消耗相邻高度差两倍的体力，下坡时消耗相邻高度差一倍的体力，平地不消耗体力，登山者体力消耗到零时会有生命危险。
例如，上图所示的山峰，从索引 0，走到索引 1，高度差为 1，需要消耗 $2 \times 1 = 2$ 的体力，从索引 2 高度 2 走到高度 4 索引 3 需要消耗 $2 \times 2 = 4$ 的体力。如果是从索引 3 走到索引 4 则消耗 $1 \times 1 = 1$ 的体力。

登山者想要知道一张地图中有多少座山峰

示例 1

输入：

[0, 1, 4, 3, 1, 0, 0, 1, 2, 3, 1, 2, 1, 0]

输出：

3

说明：

山峰所在的索引分别为 2, 10, 12

import java.util.*;

```

public class Solution {
    /**
     * 返回地图中山峰的数量
     * @param hill_map int 整型一维数组 地图数组(长度大于 1)
     * @return int 整型
     */
    public int count_peaks (int[] hill_map) {
        // write code here
        int lenght=hill_map.length;
        int result=0;
        for(int i=0;i<lenght;i++){
            if(i==0){
                if(hill_map[i]>0 && lenght>1 && hill_map[i]>hill_map[i+1]){
                    result++;
                }
            }
            else if(i==lenght-1){
                if(hill_map[i]>0 && lenght>1 && hill_map[i]>hill_map[i-1]){
                    result++;
                }
            }
            else{
                if(hill_map[i]>hill_map[i-1] && hill_map[i] > hill_map[i+1]){
                    result++;
                }
            }
        }
        return result;
    }
}

```