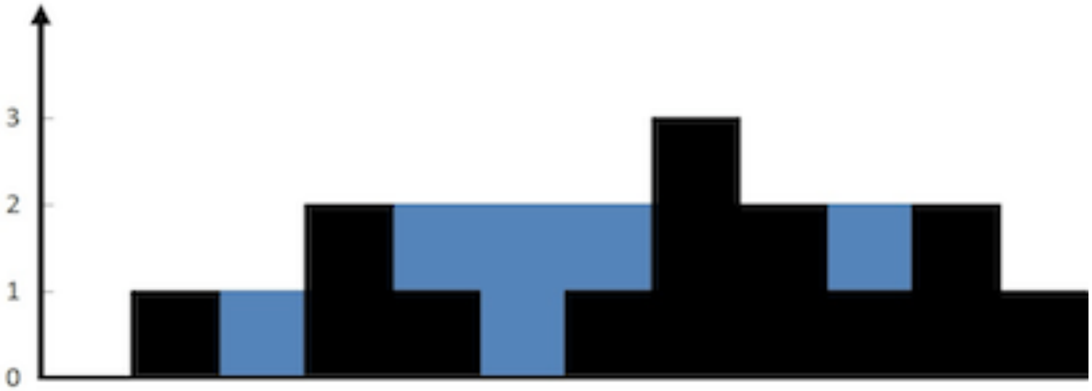
```
class Solution {
    public int trap(int[] height) {
        Stack<Integer> s = new Stack<>();
        int water = 0, i = 0, len = height.length;

        while (i < len) {
            if (s.isEmpty() || height[s.peek()] >= height[i]) {
                s.push(i++);
            } else {
                int idx = s.pop();
                if (!s.isEmpty()) {
                    int w = i - s.peek() - 1;
                    int h = Math.min(height[i], height[s.peek()]) - height[idx];
                    water += w * h;
                }
            }
        }
        return water;
    }
}
```

Given *n* non-negative integers representing an elevation map where the width of each bar is 1, compute how much water it is able to trap after raining.



The above elevation map is represented by array [0,1,0,2,1,0,1,3,2,1,2,1]. In this case, 6 units of rain water (blue section) are being trapped. **Thanks Marcos** for contributing this image!

Example:

```
Input: [0,1,0,2,1,0,1,3,2,1,2,1]
Output: 6
```