

42. Trapping Rain Water

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

Example 1:



Input: height = [0,1,0,2,1,0,1,3,2,1,2,1]

Output: 6

Explanation: The above elevation map (black section) 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.

Example 2:

Input: height = [4,2,0,3,2,5]

Output: 9

Constraints:

- $n == \text{height.length}$
- $1 \leq n \leq 2 * 10^4$
- $0 \leq \text{height}[i] \leq 10^5$

```
class Solution:
    def trap(self, height: List[int]) -> int:
        maximumLeft = [0]*len(height)
        maximumLeft[0]=height[0]
        maximumRight = [0]*len(height)
        maximumRight[-1] = height[-1]

        for i in range(1,len(height)):
            maximumLeft[i] = max(maximumLeft[i-1],height[i])
```

```

for i in range(len(height)-2,-1,-1):
    maximumRight[i] = max(maximumRight[i+1],height[i])

# print(maximumRight)
# print(maximumLeft)

ans = 0
for i in range(len(height)):
    hg = min(maximumLeft[i],maximumRight[i])-height[i]
    ans=ans+hg*1
return ans

```

```

class Solution:
    def trap(self, height: List[int]) -> int:
        left = height[0]
        right = height[-1]
        n = len(height)
        l = 0
        r = n-2
        water = 0
        while l<=r:
            if left<right:
                if height[l]>=left:
                    left = height[l]
                else:
                    water+=left-height[l]
                    l=l+1
            else:
                if height[r]>=right:
                    right = height[r]
                else:
                    water+=right-height[r]
                    r=r-1
        return water

```