

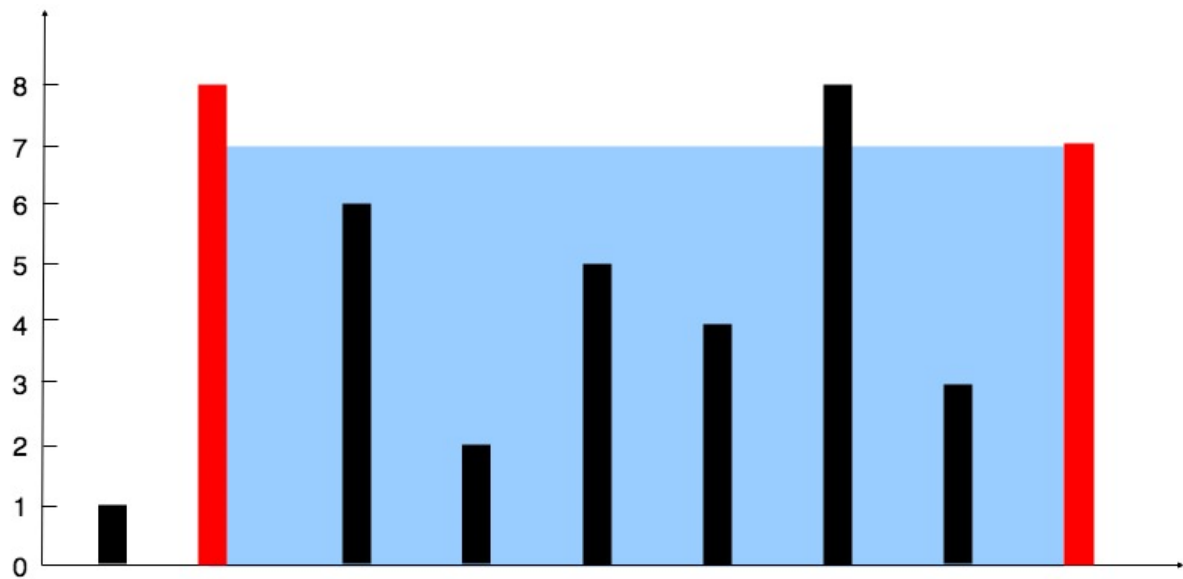
1 Container With Most Water

You are given an integer array `height` of length `n`. There are `n` vertical lines drawn such that the two endpoints of the `i`th line are $(i, 0)$ and $(i, \text{height}[i])$.

Find two lines that together with the x-axis form a container, such that the container contains the most water.

Return the maximum amount of water a container can store.

Notice that you may not slant the container.



Example 1:

Input: 1 8 6 2 5 4 8 3 7

Output: 49

Explanation: The above vertical lines are represented by array `[1,8,6,2,5,4,8,3,7]`. In this case, the max area of water (blue section) the container can contain is 49.

Example 2:

Input: 1 1

Output: 1

Source code:

```
arr=list(map(int,input().split()))
n = len(arr)
i,j = 0,n-1
mx = -1
while (i <= j):
    base = j-i
    h = min(arr[j],arr[i])
    mx = max(mx,base*h)
    if arr[j] <= arr[i]:
        j-=1
    else:
        i+=1
print(mx)
```

2 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: 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: 4 2 0 3 2 5

Output: 9

Source code:

```
height=list(map(int,input().split()))
re=0
l,r=0,len(height)-1
lm=rm=0
while l<r:
    lm=max(lm,height[l])
    rm=max(rm,height[r])
    if lm<=rm:
        re+=lm-height[l]
        l+=1
    else:
        re+=rm-height[r]
        r-=1
print(re)
```