LeetCode11: Containter with most water

Given n non-negative integers a1, a2, ..., an, where each represents a point at coordinate (i, ai).
n vertical lines are drawn such that the two endpoints of line i is at (i, ai) and (i, 0).
Find two lines, which together with x-axis forms a container, such that the container contains the most water.

Note: You may not slant the container.

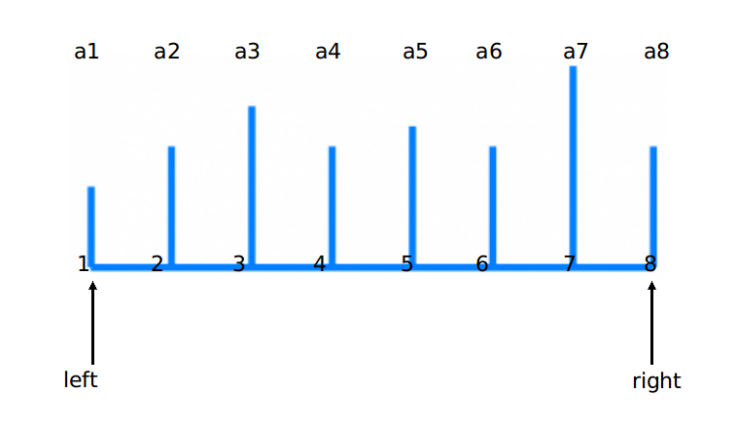
@param {integer[]} height --- an array of integer number {a1, a2, a3, ..., an}
@return {integer}

Code (in Python)

```
10
11  def maxArea(height):
12      l = 0
13      r = len(height)-1
14      maxA = 0
15
16      while l < r :
17          A = min(height[l], height[r]) * (r-l)
18          maxA = max(maxA, A)    #update the result
19
20          if height[l] < height[r]:
21              l += 1
22          else
23              r -= 1
24
25      return maxA
```

Analysis:
Initially we can assume the result is 0. Then we scan from both sides. If leftHeight < rightHeight, move right and find a value that is greater than leftHeight. Similarily, if leftHeight > rightHeight, move left and find a value that is greater than rightHeight. Additionally, keep tracking the max value.

In more detail:
imagine we have two pointer “left” and “right” pointing to the leftmost a1 and rightmost line a8, as shown in the following figure:



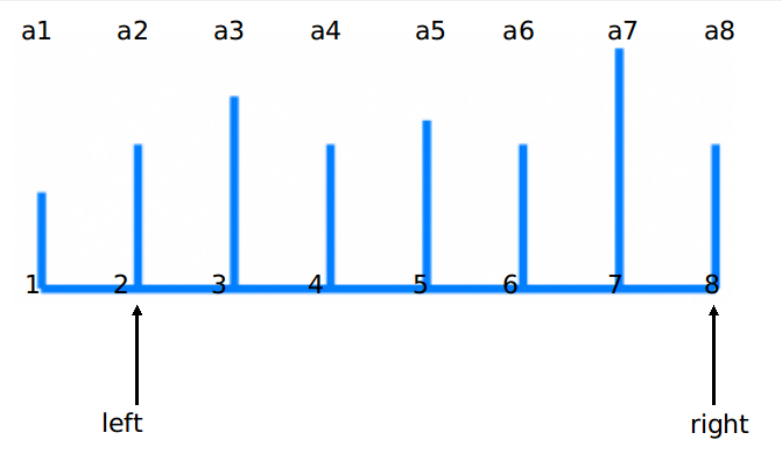
first we calculate the area of a1 and a7, ***since the water can not climb up, so the volume of water only depend of the height of the shorter line***, here a1<a8

$$A = \min(a1, a8) * (8-1) = a1*7 \qquad (1)$$

now we set (store the biggest area so far into the parameter “res”)

$$res = A \qquad (2)$$

It is easy to see from the picture, no matter how we chose the right line, from a2 to a7, the biggest area that constructed by line a1 is calculated in equation(1). Meaning that choosing ai where (i != 7), combine with a1 can not get any bigger area than A. Example, if you choose a1 and a7 to form a container, the area is a1*6 < a1*7=A.
In other words, the biggest area that a1 can form has been found, and the result has been stored in parameter “res”, and so we no longer need a1, so we move the left pointer toward right by one unit:

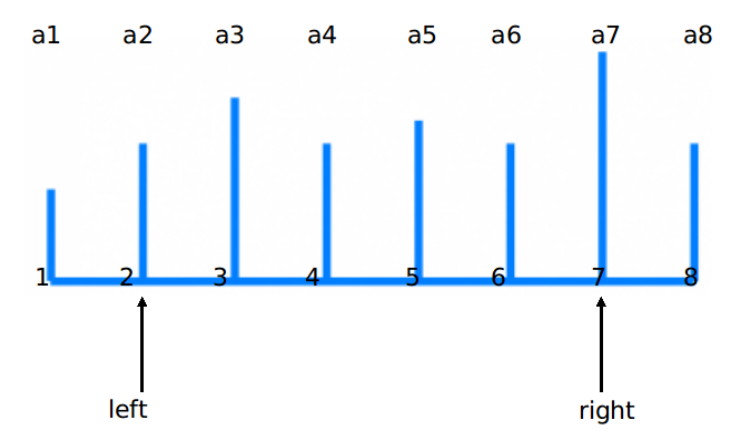


Now we update the area A = min(a2, a8)* (8-2) = a8*6, is this the biggest area we can found? We don’t know yet. Not until we compare this A with the res we have.

$$res = \max(A, res) \qquad (3)$$

Now that we update our result, keep saving the biggest area into the “res” parameter. What’s next?

Similar to the previous step, since a2 > a8, the biggest area the a8 can form has been found, we no long need a8. So we move the right pointer to the left by one unit.



Keep doing these steps until the left and right pointing meet each other, then we finish the searching process, the parameter “res” keep saving the biggest area we can find.