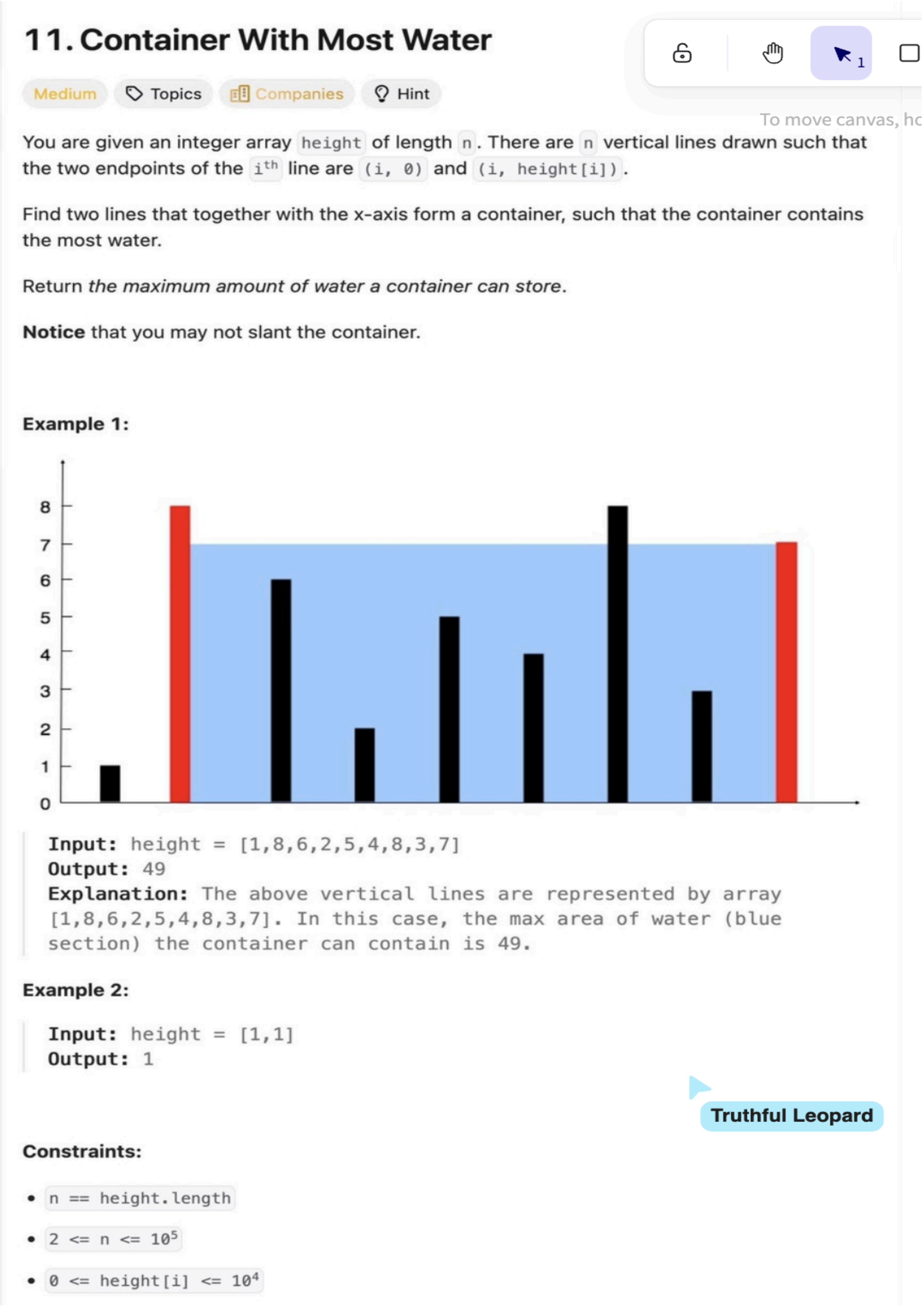

11. Container With Most Water

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, 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:** height = [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:** height = [1,1]
> **Output:** 1

Constraints:

- `n == height.length`
- `2 <= n <= 10`^{5}
- `0 <= height[i] <= 10`^{4}

Solution Pseudocode

brute force way

two-pointer
compare every element with every other element

```
result = 0
for i in range(height.length)
    for j in range(i+1, height.length)
        area = min(height[i] , height[j]) * (j - i)
        result = max(result, area)

return result
```

0(n^2)

we're limited by the minimum of height[i] and height[j]

two-pointer, but start each pointer at each end of the array for better coverage

Find the current area with the current i and

Only move the minimum of i and j

```
result = 0
left = 0
right = height.length - 1

while left < right:
    area = min(height[left] , height[right]) * (right - left)
    result = max(result, area)
    if height[left] < height[right]:
        left += 1
    else:
        right -= 1
return result
```

O(n)