Container With Most Water

11. Container With Most Water

Medium | Topics | Companies | Hint

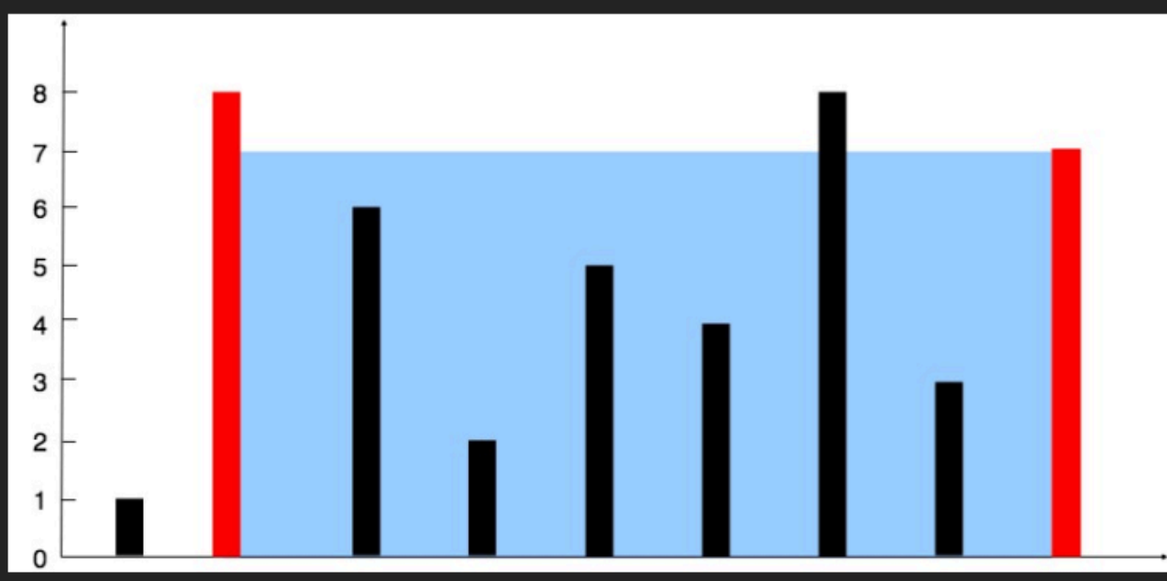
You are given an integer array `height` of length `n`. There are `n` vertical lines drawn such that the two endpoints of the `i^th` line are `(i, 0)` and `(i, height[i])`.

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.
```

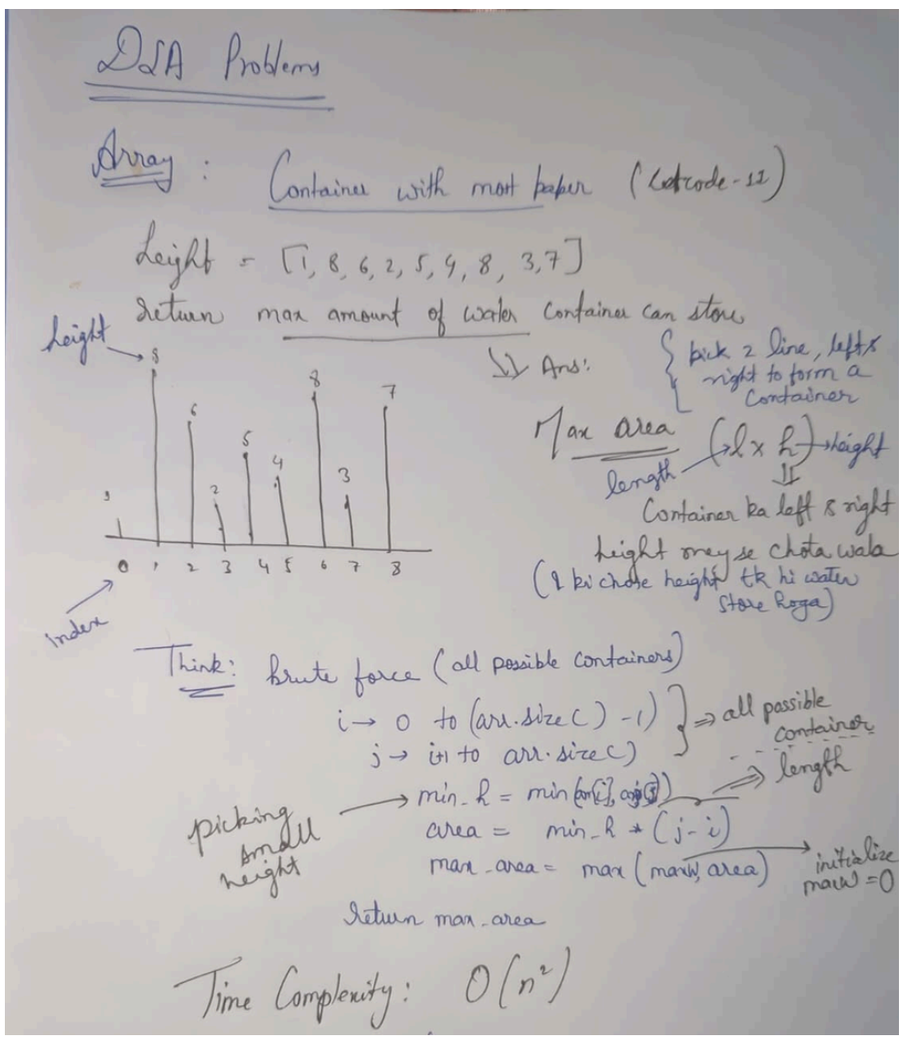
Example 2:

```
Input: height = [1,1]
Output: 1
```

Constraints:

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

Approach: Brute Force



Code

C++ Auto

```cpp
class Solution {
public:
    int maxArea(vector<int>& height) {
        int maxWater = 0;
        int n = height.size();
        for(int i=0;i<n-1;i++){
            for(int j=i+1;j<n;j++){
                int h = min(height[i],height[j]); //min height
                int w = j-i; // width
                int currWater = h*w;
                maxWater=max(maxWater,currWater);
            }
        }
        return maxWater;

    }
```

Saved

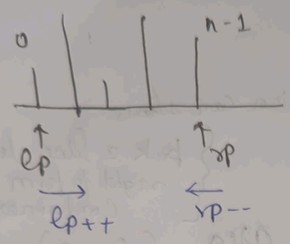
Testcase | Test Result

Accepted Runtime: 0 ms

• Case 1 • Case 2

Approach: Optimal

Optimal approach (2 pointer)



approach:

agr left pointer (lp) ka height min h right pointer (rp) se ⇒ lp++

agr rp ka height min h lp k height se ⇒ rp--

[because usko reverse krenge toh area kmsaa decrease hi hoga since area is driven by chota height bar]

```
lp = 0, rp = n-1
while( lp<rp) {
    w = rp - lp   //width
    h = min [ht[lp], ht[rp]]   //height
    Curr Wt = w * h
    max Water = max (maxWate, curWt)
    ht [lp] < ht [rp] ? lp++ : rp--;
}
```

Code

C++ Auto

```cpp
class Solution {
public:
    int maxArea(vector<int>& height) {
        int maxWater = 0;
        int n = height.size();
        int lp= 0; int rp= n-1;
        while(lp<rp){
            int w =rp-lp; //width
            int h = min(height[lp],height[rp]);
            int currWater=w*h;
            maxWater = max(maxWater,currWater);
            height[lp]<height[rp]?lp++:rp--;
        }
        return maxWater;

    }
```

Saved

Testcase | Test Result

Accepted Runtime: 0 ms

- Case 1
- Case 2