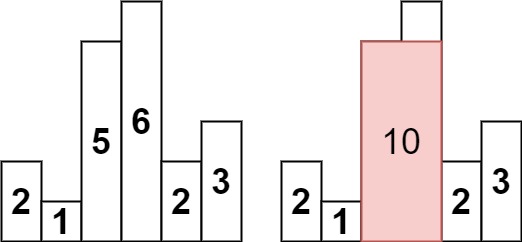
Given an array of integers heights representing the histogram's bar height where the width of each bar is 1, return *the area of the largest rectangle in the histogram*.

**Example 1:**



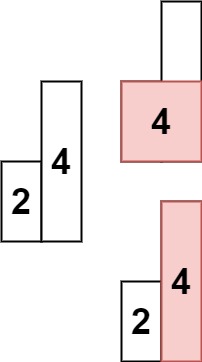
**Input:** heights = [2,1,5,6,2,3]

**Output:** 10

**Explanation:** The above is a histogram where width of each bar is 1.

The largest rectangle is shown in the red area, which has an area = 10 units.

**Example 2:**



**Input:** heights = [2,4]

**Output:** 4

Solution:

class Solution {

public int largestRectangleArea(int[] heights) {

int n = heights.length;

Stack<Integer> st = new Stack<>();

int leftSmall[] = new int[n];

int rightSmall[] = new int[n];

for(int i = 0;i<n;i++) {

while(!st.isEmpty() && heights[st.peek()] >= heights[i]) {

st.pop();

}

if(st.isEmpty()) leftSmall[i] = 0;

else leftSmall[i] = st.peek() + 1;

st.push(i);

}

// clear the stack to be re-used

while(!st.isEmpty()) st.pop();

for(int i = n-1;i>=0;i--) {

while(!st.isEmpty() && heights[st.peek()] >= heights[i]) {

st.pop();

}

if(st.isEmpty()) rightSmall[i] = n-1;

else rightSmall[i] = st.peek() - 1;

st.push(i);

}

int maxA = 0;

for(int i = 0;i<n;i++) {

maxA = Math.max(maxA, heights[i] \* (rightSmall[i] - leftSmall[i] + 1));

}

return maxA;

}

}

T.C:= O(N) [Linearly traverse] + O(N) [For stack method to find next smaller element = O(n), S.C = O(3N)

**More Optimal(In single pass only):**

|  |
| --- |
| class Solution { |
|  | public int largestRectangleArea(int[] heights) { |
|  | int n = heights.length; |
|  | Stack<Integer> st = new Stack<>(); |
|  | int maxA = 0; |
|  | for(int i = 0;i<=n;i++) { |
|  | while(!st.isEmpty() && (i==n || heights[st.peek()] >= heights[i])) { |
|  | int height = heights[st.pop()]; |
|  | int width; |
|  | if(st.isEmpty()) width = i; |
|  | else width = i - st.peek() - 1; |
|  |  |
|  | // cout << i << " " << width << " " << height << endl; |
|  | maxA = Math.max(maxA, width \* height); |
|  | } |
|  | st.push(i); |
|  | } |
|  | return maxA; |
|  | } |
|  | } |

T.C: = O(N), S.C= O(N)