**<https://leetcode.com/problems/largest-rectangle-in-histogram/description/>**

**A screen shot of a computer screen

Description automatically generated**

**Time :o(n) , space: o(n)**

class Solution:

    def largestRectangleArea(self, heights: List[int]) -> int:

        stack = []

        maxarea = -1

        for i in range(len(heights)):

            if not stack or heights[i] >= heights[stack[-1]]:

                stack.append(i)

            else:

                while stack and heights[stack[-1]] > heights[i]:

                    index = stack.pop()

                    area = -1

                    if not stack:

                        area = heights[index]\*i

                    else:

                        area = heights[index]\*(i-stack[-1]-1)

                    if area > maxarea:

                        print(index,area)

                        maxarea = area

                stack.append(i)

        i = len(heights)

        while stack:

            index = stack.pop()

            area = -1

            if not stack:

                area = heights[index]\*i

            else:

                area = heights[index]\*(i-stack[-1]-1)

            if area > maxarea:

                print(index,area)

                maxarea = area

        return maxarea

**Applied teacher code**



**BETTER CODE:**

**<https://leetcode.com/problems/largest-rectangle-in-histogram/solutions/28917/ac-python-clean-solution-using-stack-76ms/>**

**Modified scanfromleft and scanfromright:**

[**https://leetcode.com/problems/largest-rectangle-in-histogram/solutions/28902/5ms-o-n-java-solution-explained-beats-96/**](https://leetcode.com/problems/largest-rectangle-in-histogram/solutions/28902/5ms-o-n-java-solution-explained-beats-96/)

**Brute force: Try to find out rectangles of all the height**

**eg: [2,1,5,6,2,3] height width**

**Rectange of height 2 : 2 \* 1**

**Rectange of height 1 : 1 \* 6**

**Rectange of height 5 : 5 \* 2**

**Rectange of height 6 : 6 \* 1**

**Rectange of height 2 : 2 \* 4**

**Rectange of height 3 : 3 \* 1**

**Approach:For each height , i will try to find out it max width by expand its left index and rightindex**

**Time: o(n^2) space : o(1)**

class Solution:

    def largestRectangleArea(self, heights: List[int]) -> int:

        maxheight = -sys.maxsize

        for i in range(len(heights)):

            leftindex = i

            rightindex = i

            while leftindex >=0 and heights[i] <= heights[leftindex]:

                leftindex-=1

            leftindex +=1

            while rightindex < len(heights) and heights[i] <= heights[rightindex]:

                rightindex +=1

            rightindex -=1

            if (rightindex - leftindex + 1)\*heights[i] > maxheight:

                maxheight = (rightindex - leftindex + 1)\*heights[i]

        return maxheight