Question: https://leetcode.com/problems/largest-rectangle-in-histogram/

For this problem every bar will generate a rectangle where that rectangle will be considered as an whole; we can solve it using Brute force with complexity O(n^2).

Now we can also solve it in a better way where we use stack to optimize it.

We maintain an array for right boundary and another array for left boundary now we can easily find out the area.

Code:  
class Solution {

public int largestRectangleArea(int[] heights) {

int l = heights.length, presentArea=0, highestArea=0;

Stack<Integer> stck = new Stack();

int[] left=new int[l], right=new int[l];

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

if(stck.empty()){

left[i]=0;

stck.push(i);

}else{

while(!stck.empty() && heights[stck.peek()]>=heights[i])

stck.pop();

left[i]=stck.empty()?0:stck.peek()+1;

stck.push(i);

}

}

while(!stck.empty()) stck.pop();

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

if(stck.empty()){

right[i]=l-1;

stck.push(i);

}else{

while(!stck.empty() && heights[stck.peek()]>=heights[i])

stck.pop();

right[i]=stck.empty()?l-1:stck.peek()-1;

stck.push(i);

}

}

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

presentArea=(right[i]-left[i]+1)\*heights[i];

if(presentArea>highestArea){

highestArea=presentArea;

}

}

return highestArea;

}

}

Github Link :<https://lnkd.in/ecwtJeaz>