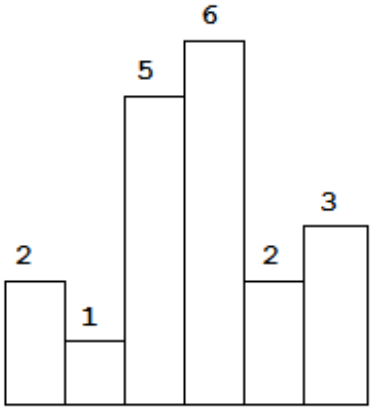
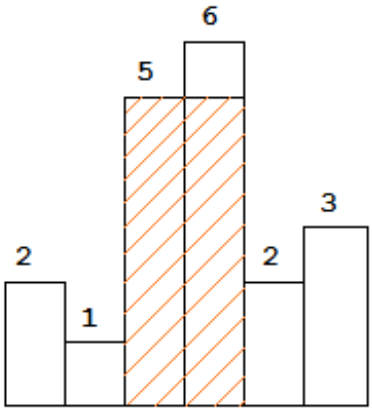
Largest Rectangle in Histogram

Given *n* non-negative integers representing the histogram's bar height where the width of each bar is 1, find the area of largest rectangle in the histogram.



Above is a histogram where width of each bar is 1, given height = `[2,1,5,6,2,3]`.



The largest rectangle is shown in the shaded area, which has area = `10` unit.

For example,
Given heights = `[2,1,5,6,2,3]`,
return `10`.

Solution 1

I push a sentinel node back into the end of height to make the code logic more concise.

```
class Solution {
  public:
      int largestRectangleArea(vector<int> &height) {

          int ret = 0;
          height.push_back(0);
          vector<int> index;

          for(int i = 0; i < height.size(); i++)
          {
              while(index.size() > 0 && height[index.back()] >= height[i])
              {
                  int h = height[index.back()];
                  index.pop_back();

                  int sidx = index.size() > 0 ? index.back() : -1;
                  if(h * (i-sidx-1) > ret)
                      ret = h * (i-sidx-1);
              }
              index.push_back(i);
          }

          return ret;
      }
  };
```

written by sipiprotoss5 original link here

Solution 2

For explanation, please see http://www.geeksforgeeks.org/largest-rectangle-under-histogram/

```
public class Solution {
    public int largestRectangleArea(int[] height) {
        int len = height.length;
        Stack<Integer> s = new Stack<Integer>();
        int maxArea = 0;
        for(int i = 0; i <= len; i++){
            int h = (i == len ? 0 : height[i]);
            if(s.isEmpty() || h >= height[s.peek()]){
                s.push(i);
            }else{
                int tp = s.pop();
                maxArea = Math.max(maxArea, height[tp] * (s.isEmpty() ? i : i - 1
- s.peek()));
                i--;
            }
        }
        return maxArea;
    }
}
```

written by wz366 original link here

Solution 3

I was stuck and took an eye on Geeks4Geeks. I got the idea and tried to figure it out by myself... It takes me a lot of time to make it through....

EDITED: Now it is pretty concise....

```
public class Solution {
public int largestRectangleArea(int[] height) {
    if (height==null) return 0;//Should throw exception
    if (height.length==0) return 0;

    Stack<Integer> index= new Stack<Integer>();
    index.push(-1);
    int max=0;

    for  (int i=0;i<height.length;i++){
            //Start calculate the max value
        while (index.peek()>-1)
            if (height[index.peek()]>height[i]){
                int top=index.pop();
                max=Math.max(max,height[top]*(i-1-index.peek()));
            }else break;

        index.push(i);
    }
    while(index.peek()!=-1){
        int top=index.pop();
        max=Math.max(max,height[top]*(height.length-1-index.peek()));
    }
    return max;
}
```

}

written by reeclapple original link here