一、题目说明

题目84. Largest Rectangle in Histogram，给定n个非负整数（每个柱子宽度为1）形成柱状图，求该图的最大面积。题目难度是Hard！

二、我的解答

这是一个 **看起来容易，做起来很容易错的题目。**我开始用的是“挖坑法”，遗憾的是总是**Time Limit Exceeded。**经过10次优化，还是很难看。“挖坑法”代码如下：

```
class Solution{
    public:
        int largestRectangleArea(vector<int>& heights){
            int len = heights.size();
            if(len<1) return 0;
            if(len==1) return heights[0];
            int result = 0;
            int start=len-1,end=0;
            int cnt = 0,sum=0;
            int min;

            while(1){
                min = INT_MAX;
                for(int i=0;i<len;i++){
                    if(heights[i]>0 && heights[i]<min){
                        min = heights[i];
                    }
                }

                //找到第1个正的
                while(end<len && heights[end]<=0){
                    end++;
                }
                while(start>0 && heights[start]<=0){
                    start--;
                }
                if(start==end){
                    sum = heights[start];
                    if(sum>result) result = sum;
                    break;
                }else if(end>start) {
                    break;
                }

                sum = 0;
                cnt = 0;
                //一次遍历，求大于0的连续长度  最大值
                while(end<len){
                    cnt = 0;
                    sum = 0;
                    while(end<len && heights[end]>=min){
                        if(heights[end]==min) heights[end] = 0;
                        cnt++;
                        end++;
                    }
                    sum = cnt * min;
```

```
                    if(sum>result) result = sum;
                    if(end<len) end++;
                }
                end = 0;
                start = len-1;
            }
            return result;
        }
};
```

性能:

```
Runtime: 1536 ms, faster than 4.53% of C++ online submissions for Largest
Rectangle in Histogram.
Memory Usage: 9.9 MB, less than 94.29% of C++ online submissions for Largest
Rectangle in Histogram.
```

还能想到的方法就是暴力计算法，性能也差不多:

```
class Solution{
    public:
        //brute force
        int largestRectangleArea(vector<int>& heights){
            int len = heights.size();
            if(len<1) return 0;
            if(len==1) return heights[0];
            int result = 0;
            bool allthesame = true;
            for(int i=0;i<len-1;i++){
                if(heights[i]!=heights[i+1]){
                    allthesame = false;
                }
            }
            if(allthesame){
                return heights[0]*len;
            }

            for(int i=0;i<len;i++){
                int minHeight = INT_MAX;
                for(int j=i;j<len;j++){
                    minHeight = min(minHeight,heights[j]);
                    result = max(result,minHeight * (j-i+1));
                }
            }

            return result;
        }
};
```

```
Runtime: 1040 ms, faster than 5.03% of C++ online submissions for Largest
Rectangle in Histogram.
Memory Usage: 10 MB, less than 94.29% of C++ online submissions for Largest
Rectangle in Histogram.
```

三、优化措施

用stack法，代码如下：

```
class Solution{
    public:
        //单调递增栈
        int largestRectangleArea(vector<int>& heights){
            int result = 0;
            stack<int> st;
            st.push(-1);//-1 放进栈的顶部来表示开始

            //按照从左到右的顺序，我们不断将柱子的序号放进栈中,直到 heights[i]
<heights[st.top]
            //将栈中的序号弹出，直到heights[stack[j]]≤heights[i]
            for(int i=0;i<heights.size();i++){
                while(st.top()!=-1 && heights[i]<heights[st.top()]){
                    int h = st.top();
                    st.pop();
                    result = max(result,heights[h]*(i - st.top() -1));
                }
                st.push(i);
            }

            // 遍历完了，但是没计算完
            while(st.top() != -1){
                int h = st.top();
                st.pop();
                int len = heights.size() - st.top() -1;
                result = max(result,heights[h]*len);
            }

            return result;
        }
};
```

```
Runtime: 16 ms, faster than 53.51% of C++ online submissions for Largest
Rectangle in Histogram.
Memory Usage: 10.4 MB, less than 91.43% of C++ online submissions for Largest
Rectangle in Histogram.
```

```
class Solution{
    public:
        //单调递增栈 ,借用i当栈
        int largestRectangleArea(vector<int>& heights){
            int result = 0;
            int len, wid;
            for (int i = 0; i < heights.size(); i++) {
                if(i != heights.size() - 1 && heights[i] <= heights[i + 1])
continue;   //这一步的判断很玄妙
                wid = heights[i];
                for (int j = i; j >= 0; j--) {
```

```
                    len = i - j + 1;
                    wid = min(wid, heights[j]);
                    result = max(result, len * wid);
                }
            }

            return result;
        }
};
```

```
Runtime: 12 ms, faster than 89.13% of C++ online submissions for Largest
Rectangle in Histogram.
Memory Usage: 10 MB, less than 94.29% of C++ online submissions for Largest
Rectangle in Histogram.
Next challenges:
```