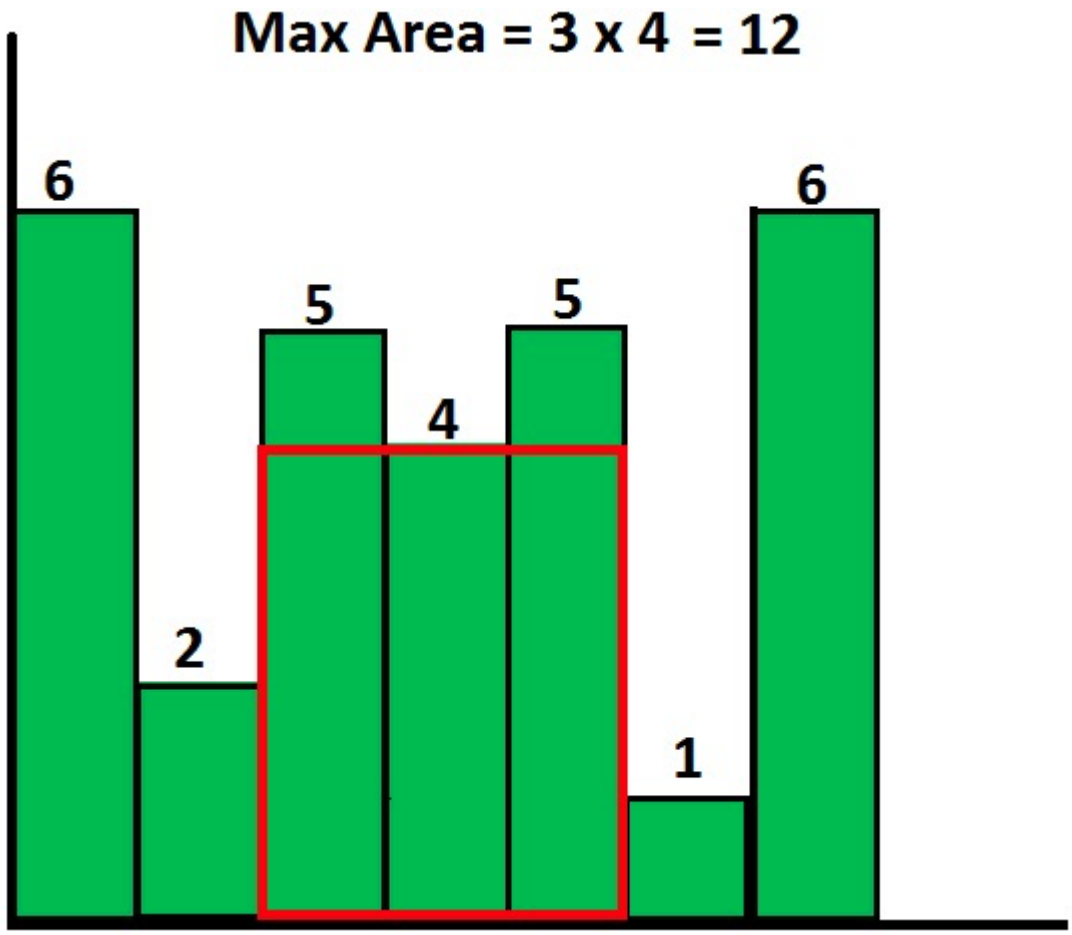
[数组]20180322_最大矩形面积

时间限制：1秒

空间限制：32768K

给定一组非负整数组成的数组h，代表一组柱状图的高度，其中每个柱子的宽度都为1。 在这组柱状图中找到能组成的最大矩形的面积（如图所示）。 入参h为一个整型数组，代表每个柱子的高度，返回面积的值。



输入描述:

输入包括两行,第一行包含一个整数n(1 ≤ n ≤ 10000)
第二行包括n个整数,表示h数组中的每个值,h_i(1 ≤ h_i ≤ 1,000,000)

输出描述:

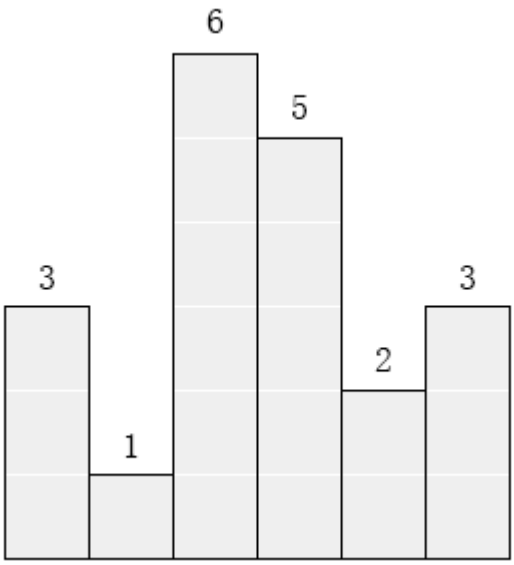
输出一个整数,表示最大的矩阵面积。

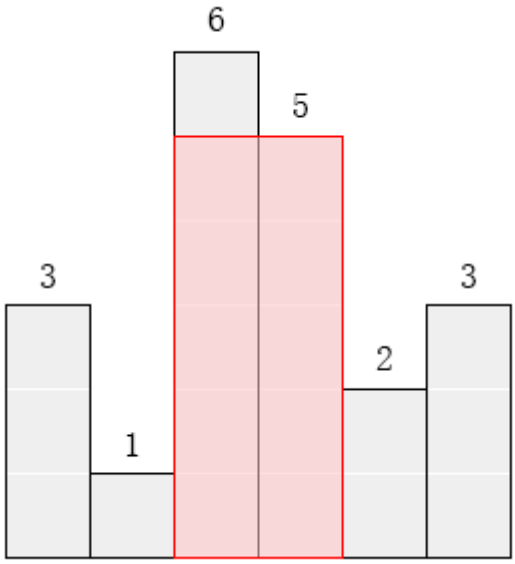
输入例子1:

```
6
2 1 5 6 2 3
```

输出例子1:

```
10
```





条形图里最大的矩形总有一个最矮的小矩形（即最小的数），所以我们可以遍历所有的小矩形，每个小矩形分为两条路找它能匹配的最大矩形，一条往前，一条往后，两边都找比它高的矩形（所以它是最矮小矩形），每找到一个，就算出现在的矩形面积大小，最后拿这个小矩形跟当前的最大矩形面积比大小，若比当前最大矩形面积大，就把它定为当前最大矩形面积；否则循环到下一个小矩形（即X轴方向加1）

如果数组给的太小，会发生越界的错误。

```
#include <iostream>
#include <string>
#include <time.h>
using namespace std;

int a[10001];        /// 定义一个足够大的数组，来容纳第二行的数
```

```
int main(){
    int n ; // 数的个数
    cin >> n;
    for( int i = 0; i <n;i++ ){
        cin >> a[i];
    }

    int max = a[0];     /// 初始值为第一个矩形条
    for( int i=1;  i<n ; i++ ){
        int nowHeight = a[i];
        /// 往前走
        for( int j = i-1; j>=0; j-- ){
            if( a[j] >= a[i] )
                nowHeight += a[i];
            else break;        /// 否则直接退出
        }

        /// 往后走
        for(int j= i+1; j < n;j++){
             if( a[j] >= a[i] )
                nowHeight += a[i];
            else break;        /// 否则直接退出
        }

        max = (max < nowHeight)? nowHeight: max;
    }

    cout << max;
    getchar();
    return 0;
}
```