Largest area Histogram in C++

```
#include <iostream>
#include <stack>
#include <vector>
using namespace std;

class LargestRectangleInHistogram {
public:
    int largestRectangleArea(vector<int>& heights) {
        stack<int> s;
        int ans = 0;
        for (int i = 0; i <= heights.size(); i++) {
            int temp = (i != heights.size()) ? heights[i] : 0;
            while (!s.empty() && temp < heights[s.top()]) {
                int tbs = s.top();
                s.pop();
                int nsr = i;
                int x1 = nsr - 1;
                int nsl = (s.empty()) ? -1 : s.top();
                int x2 = nsl + 1;
                int area = heights[tbs] * (x1 - x2 + 1);
                ans = max(ans, area);
            }
            s.push(i);
        }
        return ans;
    }
};

int main() {
    vector<int> heights = {2, 1, 5, 6, 2, 3};
    LargestRectangleInHistogram histogram;
    int maxArea =
histogram.largestRectangleArea(heights);
    cout << "The largest rectangle area is: " <<
maxArea << endl;
    return 0;
}
```

Step-by-step Table Dry Run

i	temp	Stack (Index)	Action	Computed Area	Max Area
0	2	[]	Push index 0	—	0
1	1	[0]	Pop 0 → height = 2, width = 1 → 2×1=2	2	2
		[]	Push index 1	—	2
2	5	[1]	Push index 2	—	2
3	6	[1, 2]	Push index 3	—	2
4	2	[1, 2, 3]	Pop 3 → height = 6, width = 1 → 6×1=6	6	6
		[1, 2]	Pop 2 → height = 5, width = 2 → 5×2=10	10	10
		[1]	Push index 4	—	10
5	3	[1, 4]	Push index 5	—	10
6	0	[1, 4, 5]	Pop 5 → height = 3, width = 1 → 3×1=3	3	10
		[1, 4]	Pop 4 → height = 2, width = 3 → 2×3=6	6	10
		[1]	Pop 1 → height = 1, width = 6 → 1×6=6	6	10
		[]	Push index 6 (extra 0 at end)	—	10

	✅ **Final Output:** The largest rectangle area is: 10
The largest rectangle area is: 10	