Temple offering In C++

```
#include <iostream>
#include <algorithm>
using namespace std;

int totalOfferings(int* height, int n) {
    int* larr = new int[n]; // Left offerings
array
    int* rarr = new int[n]; // Right offerings
array

    // Calculate left offerings
    larr[0] = 1;
    for (int i = 1; i < n; i++) {
        if (height[i] > height[i - 1]) {
            larr[i] = larr[i - 1] + 1;
        } else {
            larr[i] = 1;
        }
    }

    // Calculate right offerings
    rarr[n - 1] = 1;
    for (int i = n - 2; i >= 0; i--) {
        if (height[i] > height[i + 1]) {
            rarr[i] = rarr[i + 1] + 1;
        } else {
            rarr[i] = 1;
        }
    }

    // Calculate total offerings
    int ans = 0;
    for (int i = 0; i < n; i++) {
        ans += max(larr[i], rarr[i]);
    }

    // Free allocated memory
    delete[] larr;
    delete[] rarr;

    return ans;
}

int main() {
    int height[] = {2, 3, 5, 6, 4, 8, 9};
    int n = sizeof(height) / sizeof(height[0]);
    cout << totalOfferings(height, n) <<
endl;
    return 0;
}
```

Step-by-Step Execution for Input: {2, 3, 5, 6, 4, 8, 9}

1. **Initialization**:

   height = {2, 3, 5, 6, 4, 8, 9}
   n = 7
   larr = {1, 1, 1, 1, 1, 1, 1}
   rarr = {1, 1, 1, 1, 1, 1, 1} **Calculating Left Offerings**:

   - For i = 1: height[1] = 3 > height[0] = 2, so larr[1] = larr[0] + 1 = 2
   - For i = 2: height[2] = 5 > height[1] = 3, so larr[2] = larr[1] + 1 = 3
   - For i = 3: height[3] = 6 > height[2] = 5, so larr[3] = larr[2] + 1 = 4
   - For i = 4: height[4] = 4 <= height[3] = 6, so larr[4] = 1
   - For i = 5: height[5] = 8 > height[4] = 4, so larr[5] = larr[4] + 1 = 2
   - For i = 6: height[6] = 9 > height[5] = 8, so larr[6] = larr[5] + 1 = 3

   After this, larr = {1, 2, 3, 4, 1, 2, 3}.

2. **Calculating Right Offerings**:
   - For i = 5: height[5] = 8 > height[6] = 9, so rarr[5] = 1
   - For i = 4: height[4] = 4 <= height[5] = 8, so rarr[4] = 1
   - For i = 3: height[3] = 6 > height[4] = 4, so rarr[3] = rarr[4] + 1 = 2
   - For i = 2: height[2] = 5 <= height[3] = 6, so rarr[2] = 1
   - For i = 1: height[1] = 3 > height[2] = 5, so rarr[1] = rarr[2] + 1 = 2
   - For i = 0: height[0] = 2 <= height[1] = 3, so rarr[0] = 1

   After this, rarr = {1, 2, 1, 2, 1, 1, 1}.

3. **Final Offerings Calculation**: Now calculate the total offerings by summing the maximum of left and right offerings for each person:

   total = max(larr[0], rarr[0]) + max(larr[1], rarr[1]) + max(larr[2], rarr[2]) +

	max(larr[3], rarr[3]) + max(larr[4], rarr[4]) + max(larr[5], rarr[5]) + max(larr[6], rarr[6]) = 1 + 2 + 3 + 4 + 1 + 2 + 3 = 16
Output:- 16	