Largest Elements (left, right))

Given a vector of values each of which is larger than zero, for each value, find the larger of two values, one that is larger than it and comes before it, and one that is larger than it and comes after it. In either case, the indices of the larger values, if they both exist, should be closest to the index of the original value.

For example, consider the following vector (in the following, I will refer to it as the *input vector.*)

10	5	9	70	200	50	70	300	75	120	180	130	95	310	360	72	78	400	30	22
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19

For this input vector, your function should produce the following vector (I will use *output vector* to refer to this vector.)

70	10	70	200	300	200	300	310	300	300	310	310	310	360	400	360	400	0	400	30
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19

A few notes...

1. The first element of the output vector is 70 because it is the larger of the following two values (0 and 70):
   The larger of the values that precede 10 in the input vector. Since 10 is the first element of the input, there is no value that precedes it and a a result, we use 0.
   The larger of the values that follow 10 in the input vector. The first value that follows 10 in the input vector and is larger than it is 70 (at index 2). Therefore, we use 70.
2. The second element of the output vector is 10 because it is the larger of the following two values (10 and 9):
   The larger value that precedes 5 in the input vector. That is 10 at index 0.
   The larger value that follows 5 in the input vector. That is 9 at index 2.
3. The element at index 7 in the output vector contains 310. This is because it is the larger of the following two values (0 and 310):
   There is no value in the input vector that comes before 300, at index 7, that is larger than 300. Therefore, we use zero.
   310 at index 13 in the input vector is the first value that comes after 300 (at index 7) and is larger than 310.
4. The input vector will not contain any zeros. For any give element in the input, if we do not find an element that precedes it and is larger than it, we use zero. Likewise, if we do not find an element that

follows it and is larger than it, we use zero. That explains why element with index 17 of the output vector contains 0. The value of the input vector at index 17 is 400 and it is the largest element of the input vector. As a result, there is no element that precedes it and is larger than it, and there is no element that follows it and is larger than it. Therefore, we use zero as the largest element that precedes it and also zero for the largest element that succeeds it. The larger of 0 and 0 is zero.

5. On the other hand, for every other element of the input vector, it might be the case that it doesn't have an element that precedes it and is larger than it, or it might not have an element that succeeds it and is larger than it, but not both.

Base Name: larger_left_right

Function prototypes

void larger_left_right_iterative(const std::vector<int> &numbers, std::vector<int> &result); void

larger_left_right_recursive(const std::vector<int> &numbers, std::vector<int> &result, int n);

The function prototype for both functions is stored in project2.hpp. The first argument in both cases is a vector that contains the input values. The second argument is an empty vector. You store the result of your computation in this vector by use of vector's push_back function.

In both functions, numbers.size() will produce the number of elements in "numbers". In the recursive version, n is initially numbers.size(). However, in the subsequent calls to the recursive function, n helps to reduce the number of elements under consideration while numbers.size() still provides access to the number of elements.

Output

The contents of the resulting vector, one per line. Therefore, the number of output lines is the same as the number of input lines.