Given an array nums[] of size n, construct a Product Array P (of same size n) such that P[i] is equal to the product of all the elements of nums except nums[i].

Example 1:

Input:
n = 5
nums[] = {10, 3, 5, 6, 2}
Output:
180 600 360 300 900
Explanation:
For i=0, P[i] = 3*5*6*2 = 180.
For i=1, P[i] = 10*5*6*2 = 600.
For i=2, P[i] = 10*3*6*2 = 360.
For i=3, P[i] = 10*3*5*2 = 300.
For i=4, P[i] = 10*3*5*6 = 900.
Example 2:

Input:
n = 2
nums[] = {12,0}
Output:
0 12

Your Task:

You do not have to read input. Your task is to complete the function productExceptSelf() that takes array nums[] and n as input parameters and returns a list of n integers denoting the product array P. If the array has only one element the returned list should should contains one value i.e {1}

Note: Try to solve this problem without using the division operation.

Expected Time Complexity: O(n)

Expected Auxiliary Space: O(n)

Constraints:

1 <= n <= 1000

0 <= nums$_i$ <= 200

Array may contain duplicates.

Solution:

```
class Solution
{
    public static long[] productExceptSelf(int nums[], int n)
    {
        long[] left = new long[n];
        long[] right = new long[n];
        long[] ans = new long[n];

        // Initialize left and right arrays
        left[0] = 1;  // There's no element to the left of the first element
        right[n-1] = 1;  // There's no element to the right of the last element

        // Fill left array
        for (int i = 1; i < n; i++) {
            left[i] = left[i - 1] * nums[i - 1];
        }

        // Fill right array
        for (int i = n - 2; i >= 0; i--) {
            right[i] = right[i + 1] * nums[i + 1];
        }

        // Compute the answer array
        for (int i = 0; i < n; i++) {
            ans[i] = left[i] * right[i];
        }

        return ans;
    }
}
```