Approach

This is a variation of **Calculate Contribution of each element** type of problem

- In this type of problems where we need to find something such as min element, unique chars etc in all the possible substrings/subarrays and then we need to return the sum of all the results from each subarrays/substrings.
- So, here we need to calculate the ***CONTRIBUTION*** of each element in the total answer. We need to find the ending point in left as well as right side.
- X X (X |E| X X X) X
- -> Here '(' is the last valid end point on left side for E
- -> ')' is the last valid end point on right side for E

- To get the contribution : which means the total number of substrings/subarray this element will be part of , we use the below formula
- contribution[i] = left[i] * right[i]
- -> left[i] & right[i] is length of last possible bracket on left and right side resp.

- Lets see with example : X X (X |E| X X) X , here left(E) = 2 & right(E) = 3
    - So number of possible substrings/subarrays this element E will be part of is equal to 2*3 = 6 . How ?
        - For every left(E) number of parts in left side we have choice of picking right(E) number of parts.
        - X X (A |B| C D) X X : so possible substrings with B in it are

    -> {B}, {BC}, {BCD} , {AB}, {ABC}, {ABCD}
- Thus using the above technique, we can simply solve these type of problems in Linear time. Just we need to calculate left and right end points of every element beforhand. Rest answer will be contribution of each item.

Now coming back to this problem, we know that for every element we need the minimum of every subarray. So, we can find the next smallest element to left and right for each item.
And the calculate the contribution.
contribution[i] = left[i] * right[i] * arr[i]

Since we need to pick the element which is minimum, so select that subarray in which current element is minimum.

Ex: 1 (3 |2| 4 5) 0 -> Here current element is 2.

-> left is untill we find the next smaller element than 2 to its left : left[i] = 2

-> right is also until we find next smaller element than 2 to its right : right[i] = 3

=> contribution = 2 * 3 = 6 , possible subarrays {2}, {2,4}, {2,4,5}, {3,2}, {3,2,4}, {3,2,4,5}

To find the next smaller element to left and right, And also learn about Monotonic stack.
For this, we maintain a increasing stack from left for NEXT SMALLER ELEMENT TO LEFT. Also, similar stack for NEXT SMALLER ELEMENT TO RIGHT. These stack will store the index so that when calculating contribution, we will get the distance on left as well as right side.