Disable Alarm Editorial

A simple solution is to use two nested loops. The outer loop picks a starting element, the inner loop considers all elements (on right side of current start) as ending element.
Whenever sum of elements between current start and end becomes more than the given number, update the result if current length is smaller than the smallest length so far.
Time Complexity: Time complexity of the above approach is O(n^2).
Space Complexity: O(1).

Efficient Approach:
Idea is to use the sliding window approach. Start with an empty subarray, add elements to the subarray until the sum is less than x. If the sum is greater than x,
remove elements from the start of the current subarray.Perform the following step until you reach the end of the given array.And cover all the subarray which satisfy
the given condition .Compare the size of all those subarray and find minimum size among them.
Time Complexity: Time complexity of the above approach is O(n).
Space Complexity: O(1).
As constant extra space is required.