# Subarrays with Product Less than a Target (medium)

**We'll cover the following**

## Problem Statement #

Given an array with positive numbers and a target number, find all of its contiguous subarrays whose **product is less than the target number**.

### Example 1:

```
Input: [2, 5, 3, 10], target=30
Output: [2], [5], [2, 5], [3], [5, 3], [10]
Explanation: There are six contiguous subarrays whose product is less than the target.
```

### Example 2:

```
Input: [8, 2, 6, 5], target=50
Output: [8], [2], [8, 2], [6], [2, 6], [5], [6, 5]
Explanation: There are seven contiguous subarrays whose product is less than the target.
```

## Try it yourself #

Try solving this question here:

Java Python3 JS C++

```
import java.util.*;

class SubarrayProductLessThanK {

  public static List<List<Integer>> findSubarrays(int[] a
    List<List<Integer>> subarrays = new ArrayList<>();
    // TODO: Write your code here
    int start =0, product = 1;
    for(int end =0;end<arr.length;end++){
      product *= arr[end];
      while(product>=target && start<=end){
        product = product/arr[start++];
      }
      ArrayList<Integer> temp = new ArrayList<>();
      for(int i = end;i>=start;i--){
        temp.add(0,arr[i]);
        subarrays.add(new ArrayList<>(temp));
      }
    }
    return subarrays;
  }
}

```

Show Results Show Console

2 of 2 Tests Passed

| Result | Input | Expected Output | Actual Output | Reason |
|---|---|---|---|---|
| ✓ | findSubarrays([2, 5, 3, 10],30) | [[2], [3], [3, 5], [5], [10], [5, 2]] | [[2], [5], [2, 5], [3], [5, 3], [10]] | Succeeded |
| ✓ | findSubarrays([8, 2, 6, 5],50) | [[2], [5, 6], [5], [6], [2, 6], [8], [2, … | [[8], [2], [8, 2], [6], [2, 6], [5], [6, … | Succeeded |

8.418s

## Solution #

This problem follows the **Sliding Window** and the **Two Pointers** pattern and shares similarities with Triplets with Smaller Sum (https://www.educative.io/collection/page/5668639101419520/5671464854355968/5554621957275648/) with two differences:

1. In this problem, the input array is not sorted.

2. Instead of finding triplets with sum less than a target, we need to find all subarrays having a product less than the target.

The implementation will be quite similar to Triplets with Smaller Sum (https://www.educative.io/collection/page/5668639101419520/5671464854355968/5554621957275648/).

### Code #

Here is what our algorithm will look like:

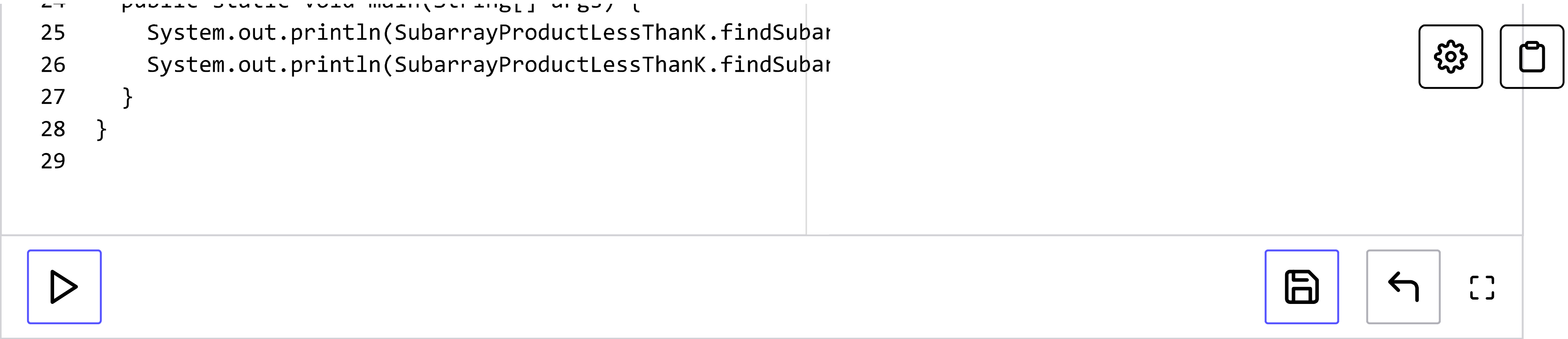
Java Python3 C++ JS

```
import java.util.*;

class SubarrayProductLessThanK {

  public static List<List<Integer>> findSubarrays(int[] a
    List<List<Integer>> result = new ArrayList<>();
    int product = 1, left = 0;
    for (int right = 0; right < arr.length; right++) {
      product *= arr[right];
      while (product >= target && left < arr.length)
        product /= arr[left++];
      // since the product of all numbers from left to ri
      // all subarrays from left to right will have a pro
      // duplicates, we will start with a subarray contai
      List<Integer> tempList = new LinkedList<>();
      for (int i = right; i >= left; i--) {
        tempList.add(0, arr[i]);
        result.add(new ArrayList<>(tempList));
      }
    }
    return result;
  }

  public static void main(String[] args) {
```



```
  public static void main(String[] args) {
    System.out.println(SubarrayProductLessThanK.findSubar
    System.out.println(SubarrayProductLessThanK.findSubar
  }
}

```

### Time complexity #

The main `for-loop` managing the sliding window takes $O(N)$ but creating subarrays can take up to $O(N^2)$ in the worst case. Therefore overall, our algorithm will take $O(N^3)$.

### Space complexity #

Ignoring the space required for the output list, the algorithm runs in $O(N)$ space which is used for the temp list.

Can you try estimating how much space will be required for the output list?

Show Hint

It is not all the Combinations of all elements of the array!

For an array with distinct elements, finding all of its contiguous subarrays is like finding the number of ways to choose two indices i and j in the array such that `i <= j`.

If there are a total of `n` elements in the array, here is how we can count all the contiguous subarrays:

- When `i = 0`, `j` can have any value from ‘0’ to ‘n-1’, giving a total of ‘n’ choices.
- When `i = 1`, `j` can have any value from ‘1’ to ‘n-1’, giving a total of ‘n-1’ choices.
- Similarly, when `i = 2`, `j` can have ‘n-2’ choices.

  …

  …
- When `i = n-1`, `j` can only have ‘1’ choice.

Let’s combine all the choices:

```
n + (n-1) + (n-2) + ... 3 + 2 + 1
```

Which gives us a total of: $n * (n+1)/2$

So, at the most, we need a space of $O(n^2)$ for all the output lists.

← Back

Triplets with Smaller Sum (medium)

Next →

Dutch National Flag Problem (medium)

☑ Mark as Completed

8% completed, meet the criteria and claim your course certificate!

Claim Certificate

Ask a Question

Report an Issue (https://discuss.educative.io/tag/subarrays-with-product-less-than-a-target-medium__pattern-two-pointers__grokking-the-coding-interview-patterns-for-coding-questions)