# Count Triplets ★

Problem Submissions Leaderboard Editorial

RATE THIS CHALLENGE

★ ★ ★ ★ ★

You are given an array and you need to find number of tripets of indices $(i, j, k)$ such that the elements at those indices are in geometric progression for a given common ratio $r$ and $i < j < k$.

**Example**

$arr = [1, 4, 16, 64]$ $r = 4$

There are $[1, 4, 16]$ and $[4, 16, 64]$ at indices $(0, 1, 2)$ and $(1, 2, 3)$. Return $2$.

**Function Description**

Complete the countTriplets function in the editor below.

countTriplets has the following parameter(s):

- int arr[n]: an array of integers
- int r: the common ratio

**Returns**

- int: the number of triplets

**Input Format**

The first line contains two space-separated integers $n$ and $r$, the size of $arr$ and the common ratio.
The next line contains $n$ space-seperated integers $arr[i]$.

**Constraints**

- $1 \leq n \leq 10^5$
- $1 \leq r \leq 10^9$
- $1 \leq arr[i] \leq 10^9$

**Sample Input 0**

```
4 2
1 2 2 4
```

**Sample Output 0**

```
2
```

**Explanation 0**

There are $2$ triplets in satisfying our criteria, whose indices are $(0, 1, 3)$ and $(0, 2, 3)$

**Sample Input 1**

```
6 3
1 3 9 9 27 81
```

**Sample Output 1**

```
6
```

**Explanation 1**

The triplets satisfying are index $(0, 1, 2)$, $(0, 1, 3)$, $(1, 2, 4)$, $(1, 3, 4)$, $(2, 4, 5)$ and $(3, 4, 5)$.

**Sample Input 2**

```
5 5
1 5 5 25 125
```

**Sample Output 2**

```
4
```

**Explanation 2**

The triplets satisfying are index $(0, 1, 3)$, $(0, 2, 3)$, $(1, 3, 4)$, $(2, 3, 4)$.

Change Theme Language C#

```
using System.CodeDom.Compiler;
using System.Collections.Generic;
using System.Collections;
using System.ComponentModel;
using System.Diagnostics.CodeAnalysis;
using System.Globalization;
using System.IO;
using System.Linq;
using System.Reflection;
using System.Runtime.Serialization;
using System.Text.RegularExpressions;
using System.Text;
using System;

class Solution {

    // Complete the countTriplets function below.
    static long countTriplets(List<long> arr, long r) {

        long count = 0;
            var arrayLength = arr.Count;
            var dict = new Dictionary<long, long>();
            var dictPairs = new Dictionary<long, long>();

            foreach (long val in arr)
            {
                if (dict.ContainsKey(val))
```

Line: 61 Col: 1

Upload Code as File  Test against custom input  Run Code  Submit Code