**6 kyu**

**Prefix Permutation**

3194% of 1726 of48[medmelki](https://www.codewars.com/users/medmelki)

Java

* [TRAIN AGAIN](https://www.codewars.com/kata/prefix-permutation/train/java)
* [NEXT KATA](https://www.codewars.com/trainer/java)

Details

[Solutions](https://www.codewars.com/kata/prefix-permutation/solutions/java)

[Discourse (23)](https://www.codewars.com/kata/prefix-permutation/discuss/java)

* Add to Collection
* |
* Share this kata:

Given a sequence of integers of length n, print the number of prefix permutations within it.

A prefix permutation is an index P that satisfies :

* 0 <= P < N
* the sequence A[0], A[1], .....A[P] is a permutation of integers from 1 to P + 1

*Example* :

if given sequence is [2,1,3,5,4], then ...

[2] is NOT a permuation of [1]

[2,1] IS a permutation of [1,2]

[2,1,3] IS a permutation of [1,2,3]

[2,1,3,5] is NOT a permutation of [1,2,3,4]

[2,1,3,5,4] IS a permutation of [1,2,3,4,5]

...and there are 3 prefix permutations

<https://www.codewars.com/kata/prefix-permutation/solutions>

**import java.util.TreeSet;**

**import java.util.SortedSet;**

**class PrefixPermutation {**

**static int prefix(final int[] sequence) {**

**final SortedSet<Integer> encountered = new TreeSet<>();**

**int prefixPermutations = 0;**

**for (int i = 0; i < sequence.length; i++) {**

**encountered.add(sequence[i]);**

**if (encountered.size() == i + 1 && encountered.first() == 1 && encountered.last() == i + 1) {**

**prefixPermutations++;**

**}**

**}**

**return prefixPermutations;**

**}**

**}**

//esto no funciona

public static int prefix(int[] a)

{

int ans = 0;

int sum = 0;

for (int i = 0; i < a.length; i++)

{

if(a[i] > a.length ||a[i] < 1 )

{

break;

}

sum += a[i];

int n = i + 1;

if (sum == (n \* (n + 1)) / 2)

{

ans++;

}

}

return ans;

}