## horizontal line

**Importance of Time**

# **Problem Statements :**

There are N processes to be completed. All the processes have a unique number assigned to them from 1 to N.

Now, we are given two things:

1)The calling order in which all the processes are called.

2)The ideal order in which all the processes should have been executed.

Executing a process takes 1 unit of time. Changing the position takes 1 unit of time.

We have to find out the unit of time required to complete all the process such that a process is executed from the ideal order only when it exists at the same index in the calling order. We can push the first term from the calling order to the last thus rotating the order.

# **Input Format :**

A single number N

# **Constraints :**

N < 100

# **Output Format :**

The total time required

# **Sample Input :**

5  
5 4 2 3 1  
5 2 1 4 3

**Sample Output :**

7

**Explanation :**

Iteration #1: Since the ideal order and calling order both has process #5 to be executed first. Process #5 is executed taking 1 unit of time. The new calling order is: 4 - 2 - 3 - 1. Time taken in step #1: 1.

Iteration #2: Since the ideal order has process #2 to be executed firstly, the calling ordered has to be changed again, i.e., the first element has to be pushed to the last place. The new calling order is: 2 - 3 - 1 - 4 and process #2 is executed. Time taken in step #2: 2.

Iteration #3: Since the ideal order has process #1 to be executed firstly, the calling ordered has to be changed again, i.e., the first element has to be pushed to the last place. The new calling order is: 1 - 4 - 3 and process #1 is executed. Time taken in step #2: 2. Iteration #4: Since the new first element of the calling order is same as the ideal order, that process will be executed. Time taken in step #4: 1.

Iteration #5: Since the last element of the calling order is same as the ideal order, that process will be executed. Time taken in step #5: 1.

Total time taken = 7

**Time Limit :**

**none**