**Lily’s Homework**

Whenever George asks Lily to hang out, she's busy doing homework. George wants to help her finish it faster, but he's in over his head! Can you help George understand Lily's homework so she can hang out with him?

Consider an array of  distinct integers, . George can swap any two elements of the array any number of times. An array is *beautiful* if the sum of  among  is minimal possible (after, possibly, performing some swaps).

Given the array , find and print the minimum number of swaps that should be performed in order to make the array *beautiful*.

**Input Format**

The first line contains a single integer, , denoting the number of elements in the array .   
The second line contains  space-separated integers describing the respective distinct values of .

**Constraints**

**Output Format**

Print the minimum number of swaps that should be performed in order to make the array *beautiful*.

**Sample Input**

4

2 5 3 1

**Sample Output**

2

**Explanation**

Let's define array  to be the beautiful reordering of array , as the sum of the absolute values of differences between its adjacent elements is minimal among all permutations and only two swaps ( with  and then  with ) was performed.

**Solution**

#include <bits/stdc++.h>

using namespace std;

int *lilysHomework* (vector <int> arr, vector<int> asc, vector<int>des, int length, map <int, int> maps)

{

int swaps = 0;

map<int, int> :: iterator it;

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

{

if(asc[i] != arr[i])

{

//find the index of the element from the map

it = maps.find(asc[i]);

int index = it->second;

swap(arr[i], arr[index]); //swap performed

//updating the maps value

it = maps.find(arr[index]);

it->second = index;

swaps+=1;

}

}

return swaps;

}

int main() {

int n;

cin >> n;

map <int, int> maps;

vector<int> arr(n);

vector<int> asc(n);

vector<int> des(n);

for(int arr\_i = 0; arr\_i < n; arr\_i++){

cin >> arr[arr\_i];

maps.insert(pair <int, int> (arr[arr\_i], arr\_i));

}

//Copying the input data to ascending and descending vectors

asc = arr;

des = arr;

sort(asc.begin(), asc.end());

sort(des.begin(), des.end(), greater<int>());

//Finding the swaps by comparing input data array with ascending Sorted array

int result = lilysHomework(arr, asc, des, n, maps);

//Finding the swaps by comparing input data array with Descending sorted array

maps.clear(); //Reset the maps

for(int i=0; i<n; i++)

maps.insert(pair <int, int> (arr[i], i));

int result2 = lilysHomework(arr, des, asc, n, maps);

cout<<min(result, result2)<<endl;

return 0;

}

`’Anshul AgGarwal