



Full Name: GUILHERME FERNANDES

Email: contato@guifr.com.br

Test Name: Mock Test

Taken On: 6 Dec 2022 04:45:01 IST

Time Taken: 5 min 23 sec/ 60 min

Invited by: Ankush

Invited on: 6 Dec 2022 04:44:53 IST

Skills Score:

Tags Score:

Algorithms 75/75

Basic Programming 0/110

Core CS 75/185

General Programming 0/110

Medium 75/185

Search 75/75

problem-solving 75/185

40.5%

75/185

scored in Mock Test in 5 min 23 sec on 6 Dec 2022 04:45:01 IST

Recruiter/Team Comments:

No Comments.

	Question Description	Time Taken	Score	Status
Q1	Pairs > Coding	4 min 37 sec	75/ 75	✓
Q2	Almost Sorted > Coding	32 sec	0/ 110	✗

QUESTION 1

✓

Correct Answer

Score 75

Pairs > Coding

Search Algorithms Medium problem-solving Core CS

QUESTION DESCRIPTION

Given an array of integers and a target value, determine the number of pairs of array elements that have a difference equal to the target value.

Example

$k = 1$

$arr = [1, 2, 3, 4]$

There are three values that differ by  $k = 1$ :  $2 - 1 = 1$ ,  $3 - 2 = 1$ , and  $4 - 3 = 1$ . Return 3.

Function Description

Complete the *pairs* function below.

pairs has the following parameter(s):

- *int* *k*: an integer, the target difference
- *int* *arr[n]*: an array of integers

#### Returns

- *int*: the number of pairs that satisfy the criterion

#### Input Format

The first line contains two space-separated integers *n* and *k*, the size of *arr* and the target value.

The second line contains *n* space-separated integers of the array *arr*.

#### Constraints

- $2 \leq n \leq 10^5$
- $0 < k < 10^9$
- $0 < arr[i] < 2^{31} - 1$
- each integer *arr[i]* will be unique

#### Sample Input

STDIN	Function
5 2	arr[] size n = 5, k =2
1 5 3 4 2	arr = [1, 5, 3, 4, 2]

#### Sample Output

3

#### Explanation

There are 3 pairs of integers in the set with a difference of 2: [5,3], [4,2] and [3,1]. .

#### CANDIDATE ANSWER

Language used: **Java 8**

```
1 class Result {
2
3     /*
4      * Complete the 'pairs' function below.
5      *
6      * The function is expected to return an INTEGER.
7      * The function accepts following parameters:
8      * 1. INTEGER k
9      * 2. INTEGER_ARRAY arr
10     */
11
12     public static int pairs(int k, List<Integer> arr) {
13         int result = 0;
14         Collections.sort(arr);
15
16         int startIndex = 0;
17         int finalIndex = 0;
18
19         while(startIndex < arr.size() && finalIndex < arr.size()){
20             while(finalIndex < arr.size() && arr.get(finalIndex) -
21 arr.get(startIndex) < k) finalIndex++;
22
23             if(finalIndex < arr.size() && arr.get(finalIndex) -
```

```

24 arr.get(startIndex) == k) result++;
25
26     startIndex++;
27 }
28
29 return result;
30
31 }
32
33 }

```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 1	Easy	Hidden case	✔ Success	5	0.1186 sec	29.7 KB
Testcase 2	Easy	Hidden case	✔ Success	5	0.102 sec	30 KB
Testcase 3	Easy	Hidden case	✔ Success	5	0.1101 sec	30.3 KB
Testcase 4	Easy	Hidden case	✔ Success	5	0.1237 sec	30 KB
Testcase 5	Easy	Hidden case	✔ Success	5	0.122 sec	30 KB
Testcase 6	Easy	Hidden case	✔ Success	5	0.1672 sec	31.5 KB
Testcase 7	Easy	Hidden case	✔ Success	5	0.2098 sec	31.7 KB
Testcase 8	Easy	Hidden case	✔ Success	5	0.1382 sec	30.5 KB
Testcase 9	Easy	Hidden case	✔ Success	5	0.1962 sec	31 KB
Testcase 10	Easy	Hidden case	✔ Success	5	0.1839 sec	32.4 KB
Testcase 11	Easy	Hidden case	✔ Success	5	0.2681 sec	46.4 KB
Testcase 12	Easy	Hidden case	✔ Success	5	0.2912 sec	46.6 KB
Testcase 13	Easy	Hidden case	✔ Success	5	0.3269 sec	46.1 KB
Testcase 14	Easy	Hidden case	✔ Success	5	0.2333 sec	47.3 KB
Testcase 15	Easy	Hidden case	✔ Success	5	0.3098 sec	48 KB
Testcase 16	Easy	Sample case	✔ Success	0	0.1174 sec	29.6 KB
Testcase 17	Easy	Sample case	✔ Success	0	0.1001 sec	29.7 KB
Testcase 18	Easy	Sample case	✔ Success	0	0.1223 sec	30 KB

No Comments

## QUESTION 2



Wrong Answer

Score 0

## Almost Sorted > Coding

Medium

Basic Programming

problem-solving

Core CS

General Programming

### QUESTION DESCRIPTION

Given an array of integers, determine whether the array can be sorted in ascending order using only one of the following operations one time.

1. Swap two elements.
2. Reverse one sub-segment.

Determine whether one, both or neither of the operations will complete the task. Output is as follows.

1. If the array is already sorted, output `yes` on the first line. You do not need to output anything else.
2. If you can sort this array using one single operation (from the two permitted operations) then output

yes on the first line and then:

- If elements can only be swapped,  $d[l]$  and  $d[r]$ , output  $swap\ l\ r$  in the second line.  $l$  and  $r$  are the indices of the elements to be swapped, assuming that the array is indexed from  $1$  to  $n$ .
- If elements can only be reversed, for the segment  $d[l..r]$ , output  $reverse\ l\ r$  in the second line.  $l$  and  $r$  are the indices of the first and last elements of the subarray to be reversed, assuming that the array is indexed from  $1$  to  $n$ . Here  $d[l..r]$  represents the subarray that begins at index  $l$  and ends at index  $r$ , both inclusive.

If an array can be sorted both ways, by using either swap or reverse, choose swap.

3. If the array cannot be sorted either way, output *no* on the first line.

### Example

$arr = [2, 3, 5, 4]$

Either swap the **4** and **5** at indices 3 and 4, or reverse them to sort the array. As mentioned above, swap is preferred over reverse. Choose swap. On the first line, print `yes`. On the second line, print `swap 3 4`.

### Function Description

Complete the *almostSorted* function in the editor below.

*almostSorted* has the following parameter(s):

- *int arr[n]*: an array of integers

### Prints

- Print the results as described and return nothing.

### Input Format

The first line contains a single integer  $n$ , the size of *arr*.

The next line contains  $n$  space-separated integers  $arr[i]$  where  $1 \leq i \leq n$ .

### Constraints

$2 \leq n \leq 100000$

$0 \leq arr[i] \leq 1000000$

All  $arr[i]$  are distinct.

### Output Format

1. If the array is already sorted, output *yes* on the first line. You do not need to output anything else.
  2. If you can sort this array using one single operation (from the two permitted operations) then output *yes* on the first line and then:
    - a. If elements can be swapped,  $d[l]$  and  $d[r]$ , output  $swap\ l\ r$  in the second line.  $l$  and  $r$  are the indices of the elements to be swapped, assuming that the array is indexed from  $1$  to  $n$ .
    - b. Otherwise, when reversing the segment  $d[l..r]$ , output  $reverse\ l\ r$  in the second line.  $l$  and  $r$  are the indices of the first and last elements of the subsequence to be reversed, assuming that the array is indexed from  $1$  to  $n$ .
- $d[l..r]$  represents the sub-sequence of the array, beginning at index  $l$  and ending at index  $r$ , both inclusive.

If an array can be sorted by either swapping or reversing, choose swap.

3. If you cannot sort the array either way, output *no* on the first line.

### Sample Input 1

STDIN	Function
2	arr[] size n = 2
4 2	arr = [4, 2]

### Sample Output 1

```
yes
swap 1 2
```

### Explanation 1

You can either *swap(1, 2)* or *reverse(1, 2)*. You prefer swap.

### Sample Input 2

```
3
3 1 2
```

### Sample Output 2

```
no
```

### Explanation 2

It is impossible to sort by one single operation.

### Sample Input 3

```
6
1 5 4 3 2 6
```

### Sample Output 3

```
yes
reverse 2 5
```


### Explanation 3

You can reverse the sub-array  $d[2...5] = "5\ 4\ 3\ 2"$ , then the array becomes sorted.

## CANDIDATE ANSWER

Language used: **Java 8**

```
1 class Result {
2
3     /*
4      * Complete the 'almostSorted' function below.
5      *
6      * The function accepts INTEGER_ARRAY arr as parameter.
7      */
8
9     public static void almostSorted(List<Integer> arr) {
10        // Write your code here
11
12    }
13
14 }
15
16
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 1	Easy	Sample case	 Wrong Answer	0	0.1379 sec	29.6 KB

Testcase 2	Easy	Hidden case	⊗ Wrong Answer	0	0.1167 sec	29.6 KB
Testcase 3	Easy	Hidden case	⊗ Wrong Answer	0	0.119 sec	30 KB
Testcase 4	Easy	Hidden case	⊗ Wrong Answer	0	0.1821 sec	31.6 KB
Testcase 5	Easy	Hidden case	⊗ Wrong Answer	0	0.1869 sec	42.1 KB
Testcase 6	Easy	Hidden case	⊗ Wrong Answer	0	0.1905 sec	42 KB
Testcase 7	Easy	Hidden case	⊗ Wrong Answer	0	0.1904 sec	42.7 KB
Testcase 8	Easy	Hidden case	⊗ Wrong Answer	0	0.2346 sec	42.6 KB
Testcase 9	Easy	Hidden case	⊗ Wrong Answer	0	0.1683 sec	42.6 KB
Testcase 10	Easy	Hidden case	⊗ Wrong Answer	0	0.2352 sec	42.3 KB
Testcase 11	Easy	Hidden case	⊗ Wrong Answer	0	0.1682 sec	42.6 KB
Testcase 12	Easy	Hidden case	⊗ Wrong Answer	0	0.1821 sec	42.5 KB
Testcase 13	Easy	Hidden case	⊗ Wrong Answer	0	0.2135 sec	42.4 KB
Testcase 14	Easy	Hidden case	⊗ Wrong Answer	0	0.1547 sec	42.3 KB
Testcase 15	Easy	Hidden case	⊗ Wrong Answer	0	0.1806 sec	42.6 KB
Testcase 16	Easy	Hidden case	⊗ Wrong Answer	0	0.2038 sec	42.6 KB
Testcase 17	Easy	Hidden case	⊗ Wrong Answer	0	0.1878 sec	42.3 KB
Testcase 18	Easy	Hidden case	⊗ Wrong Answer	0	0.2071 sec	42.2 KB
Testcase 19	Easy	Hidden case	⊗ Wrong Answer	0	0.2563 sec	42.2 KB
Testcase 20	Easy	Hidden case	⊗ Wrong Answer	0	0.2509 sec	41.6 KB
Testcase 21	Easy	Sample case	⊗ Wrong Answer	0	0.1446 sec	29.4 KB
Testcase 22	Easy	Sample case	⊗ Wrong Answer	0	0.1087 sec	29.7 KB
Testcase 23	Easy	Hidden case	⊗ Wrong Answer	0	0.1287 sec	29.8 KB
Testcase 24	Easy	Hidden case	⊗ Wrong Answer	0	0.0979 sec	29.8 KB
Testcase 25	Easy	Hidden case	⊗ Wrong Answer	0	0.1013 sec	29.3 KB

No Comments