Mock Test > contato@guifr.com.br

Full Name:	GUILHERME FERNANDES
Email:	contato@guifr.com.br
Test Name:	**Mock Test**
Taken On:	6 Dec 2022 03:36:09 IST
Time Taken:	59 min 51 sec/ 60 min
Invited by:	Ankush
Invited on:	6 Dec 2022 03:35:51 IST
Skills Score:	
Tags Score:	Algorithms 50/75
	Basic Programming 60/110
	Core CS 110/185
	General Programming 60/110
	Medium 110/185
	Search 50/75
	problem-solving 110/185

59.5%

110/185

scored in **Mock Test** in 59 min 51 sec on 6 Dec 2022 03:36:09 IST

Recruiter/Team Comments:

No Comments.

Plagiarism flagged

We have marked questions with suspected plagiarism below. Please review.

	Question Description	Time Taken	Score	Status
Q1	Pairs > **Coding**	11 min 20 sec	50/ 75	!
Q2	Almost Sorted > **Coding**	48 min 42 sec	60/ 110	✓

QUESTION 1

!

Needs Review

Score 50

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

- $2 \le n \le 10^5$
- $0 < k < 10^9$
- $0 < arr[i] < 2^{31} - 1$
- 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

The candidate did not manually submit any code. The last compiled version has been auto-submitted and the score you see below is for the auto-submitted version.

Language used: **Java 8**

```
class Result {

    /*
     * Complete the 'pairs' function below.
     *
     * The function is expected to return an INTEGER.
     * The function accepts following parameters:
     *  1. INTEGER k
     *  2. INTEGER_ARRAY arr
     */

    public static int pairs(int k, List<Integer> arr) {
```

```java

        int result = 0;

        for(int i = 0; i< arr.size();i++){
            for(int j = i +1; j < arr.size();j++){
                int abs = Math.abs(arr.get(i) - arr.get(j));
                if(abs == k) result++;
            }
        }

        return result;

    }

}

```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 1	Easy	Hidden case	✓ Success	5	0.1154 sec	29.6 KB
Testcase 2	Easy	Hidden case	✓ Success	5	0.1205 sec	30.1 KB
Testcase 3	Easy	Hidden case	✓ Success	5	0.1157 sec	29.6 KB
Testcase 4	Easy	Hidden case	✓ Success	5	0.1113 sec	29.9 KB
Testcase 5	Easy	Hidden case	✓ Success	5	0.1473 sec	30.1 KB
Testcase 6	Easy	Hidden case	✓ Success	5	0.2007 sec	31.8 KB
Testcase 7	Easy	Hidden case	✓ Success	5	0.2254 sec	31.7 KB
Testcase 8	Easy	Hidden case	✓ Success	5	0.1818 sec	30.7 KB
Testcase 9	Easy	Hidden case	✓ Success	5	0.2104 sec	31.2 KB
Testcase 10	Easy	Hidden case	✓ Success	5	0.3312 sec	32.2 KB
Testcase 11	Easy	Hidden case	✗ Terminated due to timeout	0	4.0043 sec	42.6 KB
Testcase 12	Easy	Hidden case	✗ Terminated due to timeout	0	4.0046 sec	42.5 KB
Testcase 13	Easy	Hidden case	✗ Terminated due to timeout	0	4.0069 sec	42.3 KB
Testcase 14	Easy	Hidden case	✗ Terminated due to timeout	0	4.0089 sec	42.1 KB
Testcase 15	Easy	Hidden case	✗ Terminated due to timeout	0	4.0057 sec	42.5 KB
Testcase 16	Easy	Sample case	✓ Success	0	0.1151 sec	29.9 KB
Testcase 17	Easy	Sample case	✓ Success	0	0.1509 sec	30 KB
Testcase 18	Easy	Sample case	✓ Success	0	0.1055 sec	29.7 KB

No Comments

QUESTION 2

Correct Answer

Score 60

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**

```
class Result {

    /*
     * Complete the 'almostSorted' function below.
     *
```

```java
     * The function accepts INTEGER_ARRAY arr as parameter.
     */

    public static void almostSorted(List<Integer> arr) {

        List<Integer> sorted = new ArrayList<>(arr);
        Collections.sort(sorted);
        if(arr.equals(sorted)) System.out.println("yes");

        List<String> result = trySortedUseOneSwap(new ArrayList<>
(arr),sorted);
        List<String> resultReverse = trySortedUseReverse(new ArrayList<>
(arr),sorted);

        if(result.size() == 0 && resultReverse.size() == 0){
            System.out.println("no");
        }else{
            if(result.size() > 0 )
                for(String print : result){
                    System.out.println(print);
                }
            else
                for(String print : resultReverse){
                    System.out.println(print);
                }
        }



    }

    public static List<String> trySortedUseOneSwap(List<Integer> arr,
List<Integer> sorted){


        List<String> result = new ArrayList<>();
        int i = 0;
        int smaller = arr.get(i);
        int smallerIndex = i;
        while(i+1 < arr.size() && smaller < arr.get(i+1)){
            i++;
            smallerIndex = i;
        }

        i = arr.size() - 1;
        int bigger = arr.get(i);
        int biggerIndex = i;
        while(i-1 >= 0 && bigger > arr.get(i-1)){
            i--;
            biggerIndex = i;
        }

        int swap = arr.set(smallerIndex, arr.get(biggerIndex));
        arr.set(biggerIndex,swap);

        if(arr.equals(sorted)){
            result.add("yes");
            result.add(String.format("swap %d
%d",smallerIndex+1,biggerIndex+1));
        }

        return result;

    }

    public static List<String> trySortedUseReverse (List<Integer> arr,
List<Integer> sorted){
         List<String> result = new ArrayList<>();

         int smallerIndex = 0;
         int biggerIndex = 0;

         while(biggerIndex < arr.size() || smallerIndex < arr.size()){
             if(arr.get(smallerIndex) < arr.get(smallerIndex + 1)){
                smallerIndex++;
                biggerIndex = smallerIndex;
                continue;
             }
             if(biggerIndex+1 < arr.size() && arr.get(biggerIndex) >
arr.get(biggerIndex + 1)){
                 biggerIndex++;
                 continue;
             }
             break;
         }

         List<Integer> toReverse = arr.subList(smallerIndex, biggerIndex +
1);
         Collections.reverse(toReverse);

         if(arr.equals(sorted)){
            result.add("yes");
            result.add(String.format("reverse %d
%d",smallerIndex+1,biggerIndex+1));
        }

         return result;
     }

}
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 1	Easy	Sample case	✓ Success	0	0.0975 sec	30 KB
Testcase 2	Easy	Hidden case	✗ Wrong Answer	0	0.133 sec	30.1 KB
Testcase 3	Easy	Hidden case	✗ Wrong Answer	0	0.1126 sec	30.1 KB
Testcase 4	Easy	Hidden case	✗ Wrong Answer	0	0.2247 sec	32.3 KB
Testcase 5	Easy	Hidden case	✓ Success	5	0.3425 sec	45.5 KB
Testcase 6	Easy	Hidden case	✓ Success	5	0.2455 sec	45.4 KB
Testcase 7	Easy	Hidden case	✓ Success	5	0.2626 sec	45.7 KB
Testcase 8	Easy	Hidden case	✓ Success	5	0.2092 sec	45.3 KB
Testcase 9	Easy	Hidden case	✓ Success	5	0.2586 sec	45.7 KB
Testcase 10	Easy	Hidden case	✗ Wrong Answer	0	0.2986 sec	45.6 KB
Testcase 11	Easy	Hidden case	✗ Wrong Answer	0	0.2414 sec	45.7 KB
Testcase 12	Easy	Hidden case	✗ Wrong Answer	0	0.2395 sec	45.2 KB
Testcase 13	Easy	Hidden case	✗ Wrong Answer	0	0.3605 sec	45.3 KB
Testcase 14	Easy	Hidden case	✗ Wrong Answer	0	0.1753 sec	45.2 KB
Testcase 15	Easy	Hidden case	✓ Success	5	0.2268 sec	45.5 KB

Testcase 16	Easy	Hidden case	✓ Success	5	0.2724 sec	45.3 KB
Testcase 17	Easy	Hidden case	✓ Success	5	0.2176 sec	45.7 KB
Testcase 18	Easy	Hidden case	✓ Success	5	0.2489 sec	45.4 KB
Testcase 19	Easy	Hidden case	✓ Success	5	0.3086 sec	44.9 KB
Testcase 20	Easy	Hidden case	✓ Success	5	0.2351 sec	45.3 KB
Testcase 21	Easy	Sample case	✓ Success	0	0.1702 sec	29.7 KB
Testcase 22	Easy	Sample case	✓ Success	0	0.1202 sec	29.4 KB
Testcase 23	Easy	Hidden case	⊗ Wrong Answer	0	0.1162 sec	29.8 KB
Testcase 24	Easy	Hidden case	⊗ Wrong Answer	0	0.098 sec	29.4 KB
Testcase 25	Easy	Hidden case	✓ Success	5	0.113 sec	29.8 KB

No Comments

PDF generated at: 5 Dec 2022 23:08:03 UTC