

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 $d[l \dots r]$, 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 $d[l \dots r]$ 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* / *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 $d[l \dots r]$, output *reverse* / *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 .

$d[l \dots r]$ 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.