Oscillations

DiPS CodeJam 23

Prompt

Given an array of n integers, determine if dividing the list into groups of increasing and decreasing elements yields a list of equal-sized lists. In other words, determine if the "turning points" of the list are spaced out evenly, i.e., the list oscillates between increasing and decreasing "periodically". If an element exists that is neither increasing nor decreasing, assume that the list does not oscillate.

Input Format

The first and only line of input contains a space-separated list of integers.

Output Format

Your output must contain "true" if the list oscillates periodically, and "false" if it cannot.

Sample Input/Output

Input	Output
1 3 5 8 6 4 2 3 5 7 6 4 2 5 7 9 6 4 2	true

Solution

Taking the example $n[0, 3, 7, 5, 2, 3, 6]$:

- Compute the deltas: $n_{\Delta}[3, 4, -2, -3, 1, 3]$
- Compute the sign of the deltas: $n_{\text{sgn }\Delta}[1, 1, -1, -1, 1, 1]$
- Group by equality: $n_{\text{sgn }\Delta \text{ grouped}}[[1, 1], [-1, -1], [1, 1]]$
- Find the lengths of the groups: $n_{\text{length of sgn }\Delta grouped}[2, 2, 2]$
- If $len(\{n : n \in n[]\}) = 1$, then the list matches the condition.

Sample Program

```
import math
import itertools

l=list(map(int, input().strip().split()))

signum = lambda z : int(z/math.fabs(z))

deltas=[l[i+1]-l[i] for i in range(0, len(l)-1)]
```

```
delta_signs=[signum(i) for i in deltas]

lengths_of_groups = [len([*y]) for x, y in itertools.groupby(delta_signs)]

print("true" if len(set(lengths_of_groups))==1 else "false")
```