D. Two progressions

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

An arithmetic progression is such a **non-empty** sequence of numbers where the difference between any two successive numbers is constant. This constant number is called common difference. For example, the sequence 3, 7, 11, 15 is an arithmetic progression. The definition implies that any sequences whose length equals 1 or 2 are arithmetic and all sequences whose length equals 0 are non-arithmetic.

You are given a sequence of **different** integers $a_1, a_2, ..., a_n$. You should either split it into two arithmetic progressions or find out that the operation is impossible to perform. Splitting assigns each member of the given sequence to one of two progressions, but the relative order of numbers does not change. Splitting is an inverse operation to *merging*.

Input

The first line contains a positive integer n $(2 \le n \le 30000)$, n is the length of the given sequence. The second line contains elements of the given sequence $a_1, a_2, ..., a_n$ ($- 10^8 \le a_i \le 10^8$). The elements of the progression are different integers.

Output

Print the required arithmetic progressions, one per line. The progressions can be positioned in any order. Each progression should contain at least one number. If there's no solution, then print "No solution" (without the quotes)in the only line of the input file. If there are several solutions, print any of them.

Examples

input

```
6
4 1 2 7 3 10
```

output

```
1 2 3
4 7 10
```

input

```
5
1 2 3 -2 -7
```

output

```
1 2 3
-2 -7
```

Note

In the second sample another solution is also possible (number three can be assigned to the second progression): 1, 2 and 3, -2, -7.