

Maximum Perimeter Triangle

Given an array of stick lengths, use **3** of them to construct a **non-degenerate triangle** with the maximum possible perimeter. Return an array of the lengths of its sides as **3** integers in non-decreasing order.

If there are several valid triangles having the maximum perimeter:

1. Choose the one with the *longest maximum side*.
2. If more than one has that maximum, choose from them the one with the *longest minimum side*.
3. If more than one has that maximum as well, print any one them.

If no non-degenerate triangle exists, return $[-1]$.

Example

sticks = [1, 2, 3, 4, 5, 10]

The triplet (1, 2, 3) will not form a triangle. Neither will (4, 5, 10) or (2, 3, 5), so the problem is reduced to (2, 3, 4) and (3, 4, 5). The longer perimeter is $3 + 4 + 5 = 12$.

Function Description

Complete the *maximumPerimeterTriangle* function in the editor below.

maximumPerimeterTriangle has the following parameter(s):

- *int sticks[n]*: the lengths of sticks available

Returns

- *int[3]* or *int[1]*: the side lengths of the chosen triangle in non-decreasing order or -1

Input Format

The first line contains single integer *n*, the size of array *sticks*.

The second line contains *n* space-separated integers *sticks[i]*, each a stick length.

Constraints

- $3 \leq n \leq 50$
- $1 \leq sticks[i] \leq 10^9$

Explanation

Sample Case 0:

There are **2** possible unique triangles:

1. (1, 1, 1)

2. **(1, 3, 3)**

The second triangle has the largest perimeter, so we print its side lengths on a new line in non-decreasing order.

Sample Case 1:

The triangle **(1, 2, 3)** is degenerate and thus can't be constructed, so we print **-1** on a new line.