Closest Numbers

Sorting is often useful as the first step in many different tasks. The most common task is to make finding things easier, but there are other uses, as well.

Challenge

Given a list of unsorted integers, $A = \{a_1, a_2, \ldots, a_N\}$, can you find the pair of elements that have the smallest absolute difference between them? If there are multiple pairs, find them all.

Input Format

The first line of input contains a single integer, N, representing the length of array A.
In the second line, there are N space-separated integers, $a_1, a_2, \ldots, a_N$, representing the elements of array A.

Output Format

Output the pairs of elements with the smallest difference. If there are multiple pairs, output all of them in ascending order, all on the same line (consecutively) with just a single space between each pair of numbers. If there's a number which lies in two pair, print it two times (see the sample case #3 for explanation).

Constraints

- $2 \le N \le 200000$
- $-10^7 \le a_i \le 10^7$
- $a_i \ne a_j, where\ 1 \le i < j \le N$

Sample Input #1

```
10
-20 -3916237 -357920 -3620601 7374819 -7330761 30 6246457 -6461594 266854
```

Sample Output #1

```
-20 30
```

Explanation

(30) - (-20) = 50, which is the smallest difference.

Sample Input #2

```
12
-20 -3916237 -357920 -3620601 7374819 -7330761 30 6246457 -6461594 266854 -520 -470
```

Sample Output #2

```
-520 -470 -20 30
```

Explanation

(-470) - (-520) = 30 - (-20) = 50, which is the smallest difference.

Sample Input #3

```
4
5 4 3 2
```

Sample Output #3

```
2 3 3 4 4 5
```

Explanation

Here, the minimum difference will be *1*. So valid pairs are *(2, 3)*, *(3, 4)*, and *(4, 5)*. So we have to print *2* once, *3* and *4* twice each, and *5* once.