4910 CD Stacks

Kazimir, the robot, has a new job tidying stacks of CDs. Several stacks of CDs are placed evenly near the perimeter on top of a round table. The stacks contain different numbers of CDs and they are about to be put away to storage. Kazimir's pointy-haired boss is fickle: each stack must contain the same number of CDs.

Kazimir has to rebalance the stacks. But, as befits a robot, Kazimir can only move CDs between neighboring stacks. Thus, Kazimir is allowed to take any number of CDs from any stack and move these CDs to one of the two immediately adjacent stacks (we refer to this process as one move).

Help Kazimir rebalance the stacks so that each stack ends up with the same number of CDs. Of course, Kazimir is a lazy robot and seeks a solution with the fewest possible moves.

Input

The input file contains several test cases, each of them as described below.

The first line contains a positive integer n, the number of stacks ($n \leq 500,000$).

The second line contains n positive integers, separated by white space. Each integer indicates the number of CDs on one stack, in order, starting with stack 1 and ending with stack n which is adjacent to stack 1 (the total number of CDs is $< 2^{31}$).

For example, four stacks with a total of twelve CDs could be the first sample below, and four stacks with a total of seven CDs could be the second one.

Output

For each test case, write to the output as described below.

The first line contains either a positive integer p, the minimum number of moves needed for rebalancing the stacks, or the string '`IMPOSSIBLE`' if it is not possible to rebalance the stacks.

If it is possible to rebalance, the first line is followed by another p lines, each of which contains three positive integers, separated by single blanks, and describes one move: how many CDs were transferred in this move, from which stack, and to which stack.

Note for the Sample 1:

Kazimir first moves one CD from stack 1 to stack 2, then two CDs from stack 4 to stack 3, and finally one CD from stack 3 to stack 2.

Sample Input

```
4
4 1 2 5
4
```

```
1 3 2 1
```

Sample Output

```
3
1 1 2
2 4 3
1 3 2
IMPOSSIBLE
```