Load Balancing

SuperComputer Inc. have built a super-fast computer server consisting of N hyper-scalar lightning-fast processors Beta 007. These processors are numbered from 1 to N and are used to process independent jobs. Every new incoming job is assigned to an arbitrary processor. Sometimes, a processor may be assigned too many jobs while other processors have a relatively light load (or even wait idly). In that case, the whole system undergoes rebalancing.

Rebalancing proceeds in rounds. In each round, every processor can transfer at most one job to each of its neighbors on the bus. Neighbors of the processor i are the processors i-1 and i+1 (processors 1 and N have only one neighbor each, 2 and N-1 respectively). The goal of rebalancing is to achieve that all processors have the same number of jobs.

Given the number of jobs initially assigned to each processor, you are asked to determine the minimal number of rounds needed to achieve the state when every processor has the same number of jobs, or to determine that such rebalancing is not possible.

Input file specification

The input file consists of several blocks. Each block begins with a line containing a single number N(1<= N <=9000) - the number of processors. N numbers follow, separated by spaces and/or end of line characters. The i-th number denotes the number of jobs assigned to the i-th processor before rebalancing. There is a blank line after each block. The last block is followed by a single number -1 on a separate line (which should not be processed).

Output file specification

For each block in the input file, output the minimal number of rounds needed to rebalance loads for all the processors. If it is not possible to rebalance jobs so that each processor has the same number of jobs, output -1.

Example

Input file:

```
3
0 99 3

2
49 50

8
16 17 15 0 20 1 1 2

10
0 0 100 0 0 0 0 0 0 0

-1
```

Output file:

```
34
-1
23
```