Objective

The Battle Dev uses a large server with many processors managing the thousands of processes generated by candidates' codes. The problem is that the operating system that the organizers chose is not very clever in scheduling processes and there is an increasing risk of illegal competitor access to resources. It seems that someone is actively trying to exploit this security flaw, you have to prevent it otherwise the contest will be biased and it will never happen again...

To simplify, it will be considered that each process **i** consumes an amount of resources **C(i)** ≥ 0 and that the amount of resource available is worth V. Processes are run 2 by 2. The operating system has a great predictive algorithm analyzing the candidates' code. It finds a minimum amount of V resource that works provided it complies with the exclusion list. The exclusion list is an exhaustive list of processes that cannot be run simultaneously. The OS of course respects this list. There's just one problem... instead of giving you V that would allow you to scale the infrastructure capacity in real time. It only gives you the list of exclusions, small error of the programmer, large consequence.

But all is not lost. In fact, you can find V from the exclusion list and as the resource is expensive, you are looking for the minimum value of V that may explain the list provided by your algorithm. This implies :

- V is greater than or equal to all C(i)
- for each pair of processes (i,j) (i and j being different) comprised in the exclusion list, we have C(i)+C(j)>V
- for each pair of processes (i,j) (i and j being different) not comprised in the exclusion list, we have C(i)+C(j)≤V

Data format

Input

Row 1: **N** the number of processes where 1 ≤ **N** ≤ 1000.
Row 2: **N** integers separated by spaces, these integers will be called **P(i)** with **i** ranging from 0 to **N**-1 and indicate how many processes cannot be run at the same time as the process **i**, by construction we have 0 ≤ **P(i)** ≤ **N**..
Rows **i** from 3 to **N** + 2: **P (i-3)** integers separated by spaces representing process numbers that cannot be run at the same time as the process **(i-3).** If **P (i-3)**=0, the row **i** will be empty.

Output

The minimum value of the amount of resource that may explain the situation or -1 if there no such amount.

Example

Entry

```
4
1 3 2 2
1
0 2 3
1 3
1 2
```

Output

```
3
```

Comments

Processes are numbered from 0 to 3.

If we set C(0) = 1, C(1) = 3, C(2) = 2, C(3) = 2 and if V=3, then
- Processes 0 and 1 give a consumption of 4 that is greater than 3
- Processes 1 and 2 give a consumption of 5 that is greater than 3
- Processes 1 and 3 give a consumption of 5 that is greater than 3
- Processes 2 and 3 give a consumption of 5 that is greater than 3

On the other hand, you can run together the process pairs (0,2) or (0,3) that are not on the exclusion list.

So we've checked all the cases, V=3 is a resource value that explains the situation, it remains to be seen that it is minimal.

V cannot be equal to 1. If it were the case the value of each process would be 0 or 1. Furthermore :

- C(0)+C(1) is excluded so strictly greater than 1, thus C(0)=1 and C(1)=1
- C(2)+C(3) is excluded so strictly greater than 1, thus C(2)=1 and C(3)=1

Consequently (0,2) should be in the exclusion list and it's not. So V cannot be 1.

V cannot be equal to 2 as we have:

- C(0)+C(1) is excluded so strictly greater than à 2 so one of the process is equal to 2 and the other is at least equal to 1.
- C(2)+C(3) is excluded so strictly greater than à 2 so one of the process is equal to 2 and the other is at least equal to 1.

So not process is equal to 0.
So the process among (2,3) that is equal to 2 cannot be associated with any other process.
Yet processes 2 and 3 can both be associated with process 0.
So V can not be equal to 2.

<u>Input</u>

4
1 1 1 1
1
0
3
2

<u>Output</u>

-1

<u>Comments</u>

(0,1) and (2,3) are excluded therefore C(0)+ C(1) > V and C(2) + C(3) > V. Therefore C(0)+C(1)+C(2)+C(3)>2*V

(0,2) and (1,3) are not excluded therefore C(0) + C(2) ≤ V and C(1) + C(3) ≤ V therefore C(0)+C(1)+C(2)+C(3) ≤ 2*V
So V doesn't exist.