Task: Stones

When Ankica finally caught Branko, he refused to buy her newspapers and demanded they should play a different game because the last one was rigged. Ankica *innocently* suggested another game involving stones, but Branko was rightly suspicious, and decided to completely change its rules.

The game involves N piles of stones with the i-th pile having a_i stones, and players take turns removing some number of stones from a pile. The player that removes the last stone wins the game.

The catch is that the pile from which a player must remove a certain number of stones in a particular turn is forced by the other player.

More precisely, with turns indexed by increasing integers starting from 1, the game proceeds as follows:

- **The odd-numbered turns** start by Branko pointing to a non-empty pile of stones. Ankica then proceeds to remove at least one (and at most all) stones from the said pile.
- **The even-numbered turns** start by Ankica pointing to a non-empty pile of stones. Branko then proceeds to remove at least one (and at most all) stones from the said pile.

Branko found a bunch of stones, formed some piles and they began playing. As a professional gamer, Ankica quickly realized that the starting configuration of stones is winning for her, i.e. that she can win no matter how Branko plays the rest of the game.

Could you win the game in Ankica's situation?

Interaction

This is an interactive task. Your program must communicate with a program made by the organizers which takes the role of Branko. Of course, your program should take the role of Ankica and ensure she wins the game.

Your program should first read the initial state of the game from the standard input. The initial state is given in two lines. The first line contains a single integer N from the task description, while the second line contains N space-separated positive integers $a_1, a_2, \ldots, a_N$ from the task description.

Now, the game begins. Remember, your program plays as Ankica, meaning that it should act differently depending on whether the current turn is odd-numbered or even-numbered.

During an odd-numbered turn:

- Your program should first read a single integer k. If all piles are empty at this point, k will be equal to -1, and you should terminate the program as this means that the game is over and you have lost. Otherwise k ($1 \leq k \leq N$) denotes that you (Ankica) must take some number of stones from the k-th pile. It is guaranteed that the k-th pile is not empty at this point. Let the current number of stones in the k-th pile be equal to s_k.
- Your program should then output a single integer x ($1 \leq x \leq s_k$), denoting the number of stones you wish to remove from the k-th pile, and *flush* the output afterwards.

During an even-numbered turn:

- Your program should first write a single integer l and *flush* the output. If all piles are empty at this point, l must be equal to -1, and you should terminate the program as this means that the game is over and you have won. Otherwise l ($1 \leq l \leq N$) denotes that you force Branko to take take some number of stones from the l-th pile. The l-th pile must not be empty at this point. Let the current number of stones in the l-th pile be equal to s_l.
- Your program should then read a single integer y ($1 \leq y \leq s_l$), denoting the number of stones Branko removed from the l-th pile.

It is guaranteed that the initial state of the game ensures that you (Ankica) can win the game regardless of the way Branko plays.

Note: You can download the sample source code from the judging system that correctly interacts with the program made by the organizers (including the output *flush*), and solves the first example.

Scoring

Let $M = \max(a_1, a_2, \ldots, a_N)$.

Subtask	Score	Constraints
1	12	$1 \leq N, M \leq 7$
2	13	$1 \leq N \leq 12, 1 \leq M \leq 500$
3	15	$1 \leq N, M \leq 500$, and $a_i = a_j$ for all $1 \leq i, j \leq N$.
4	60	$1 \leq N, M \leq 500$

Interaction Example 1

Output	Input	Comment
	`1`	
	`4`	There is only one pile consisting of 4 stones
	`1`	Branko has no choice but to force Ankica to take the stones from the first pile.
`4`		Ankica takes all of the stones from the first pile.
`-1`		There are no stones left and Ankica wins the game.

Interaction Example 2

Output	Input	Comment
	`3`	
	`1 1 5`	There are three piles consisting of 1, 1 and 5 stones
	`3`	Branko forces Ankica to take at least one stone from the third pile
`5`		Ankica takes all of the stones from the third pile.
`1`		Ankica forces Branko to take at least one stone from the first pile.
	`1`	Branko takes the only stone from the first pile.
	`2`	Branko forces Ankica to take at least one stone from the second pile.
`1`		Ankica takes the only stone from the second pile.
`-1`		There are no stones left and Ankica wins the game