CEOI 2021

CENTRAL EUROPEAN OLYMPIAD IN INFORMATICS 2021

ZAGREB, CROATIA | SEPTEMBER 1-5

Day 2

September 4th 2021

Tasks

Task	Time Limit	Memory Limit	Score
Stones	3 seconds	512 MiB	100
Tortoise	3 seconds	512 MiB	100
Wells	10 seconds	1024 MiB	100
Total			300

REPUBLIC OF CROATIA
Ministry of Science and Education

CROATIAN ASSOCIATION OF TECHNICAL CULTURE

CROATIAN COMPUTER SCIENCE ASSOCIATION

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

Task: Tortoise

Wilco the tortoise wants to buy some candy. In order to do so, he will visit the Nakamise Shopping Street in Tokyo.

Tom the hare is a friend who is concerned that Wilco eats too much sugar. In order to decrease the number of candies Wilco is able to buy, Tom is going to buy some candies before him.

There are N locations on the street. Each of them is either a shop or a playground for kids. The distance between adjacent locations is constant. In other words, locations can be pictured as N equally spaced points on a line.

Each shop has some number of candies (possibly zero). Wilco will walk from first to last location, visiting all locations in order. Everytime he reaches a shop he will buy all available candies and put them in his bag.

Tom the hare is moving exactly twice as fast as Wilco. Contrary to Wilco, he can also move in both directions. To avoid suspicion on his mission, Tom will carry at most one candy at a time. Once he buys a candy, he will carry it until he gives it to kids on some playground. He cannot drop it anywhere else, but may drop it at some playground after Wilco has reached the final location. Tom's goal is to minimize the number of candies Wilco is going to buy.

Both of them start at the first location at time 0. Buying and dropping candies takes no time. If at some time both of them are located on the same shop, Tom can buy candy before Wilco (although Tom is always allowed to buy at most one candy). That also means that if the first location is a shop, Tom can buy candy before Wilco at time 0.

What is the total number of candies Wilco will buy under the assumption that Tom's movements and purchases are optimal?

Input

The first line contains the integer N from the task description.

The second line contains N integers $a_1, a_2, \ldots, a_N$ describing the N locations on the street.

For each i, if a_i equals -1, then the i-th location is a playground, otherwise it is a shop and a_i specifies the number of candies in it. It is possible for a shop to have no candies (i.e. a_i can be zero).

At least one location will be a playground.

Output

Output the number of candies Wilco will buy.

Scoring

Subtask	Score	Constraints
1	8	$1 \le N \le 20, \lvert a_i \rvert \le 1$
2	10	$1 \le N \le 300, \lvert a_i \rvert \le 1$
3	30	$1 \le N \le 300, -1 \le a_i \le 10\,000$
4	25	$1 \le N \le 5\,000, -1 \le a_i \le 10\,000$
5	27	$1 \le N \le 500\,000, -1 \le a_i \le 10\,000$

Examples

input	input	input
5 -1 1 1 1 1	8 -1 1 0 0 -1 0 0 3	8 2 -1 2 -1 2 -1 2 -1
output	**output**	**output**
2	1	1

Clarification of the first example: Tom goes to the shop at the second location (at that time Wilco is in-between first and second location), he buys a candy there and brings it to the playground. At the moment he reaches the playground Wilco arrives at the second location. Tom then immediately goes to the shop at the third location, arriving at the same time as Wilco. He buys a candy and brings it back to the playground. At that point Wilco is at the fourth location and Tom is not able to buy anymore candy before Wilco. So in the end Wilco buys candies at the final two shops.

Clarification of the second example: Tom buys a candy at the second location and carries it to the second playground at the fifth location. He then buys a candy at the last location and brings it to the fifth location. At this point Wilco is at the sixth location. He then goes to the last location one more time and reaches just before Wilco. At that point Wilco is in-between seventh and eight location. He buys a candy there. He cannot drop this candy and buy another in time so Wilco will be able to buy a candy at last location.

Clarification of the third example: In the beginning both Tom and Wilco are located at the first location which is a shop. Tom buys one candy before Wilco. Next, Tom drops that candy at the second location and goes to the third location, buys a candy and brings it to one of the playgrounds nearby. Then he comes back just in time as Wilco arrives at the third location so he can buy another candy before him. He brings it to the playground at the fourth location, then goes to the fifth location and buys a candy there. He drops that candy at one of the nearby playgrounds and comes back to the fifth location just in time as Wilco arrives to the fifth location so he buys another candy and brings it to playground at sixth location. He repeats the same for the last shop.

Task: Wells

On the beautiful mountain of Velebit there are N shelters. Exactly $N-1$ pairs of shelters are connected by a hiking path such that it is possible to travel between any pair of shelters using these paths.

Vila the Fairy likes hiking very much and it takes her exactly one day to traverse a hiking path connecting two shelters. She can use her magical abilities to appear at any shelter at the beginning of a day, and then spend the next $K-1$ days hiking in such a way that she never visits the same shelter more than once. Thus, during her hike, Vila visits exactly K shelters.

Vila gets thirsty while hiking so she would like some shelters to have water wells. During any possible hike of hers she wants to visit exactly one shelter with a well.

Your task is to determine whether it is possible to find a subset of shelters at which to put wells to satisfy Vila's peculiar wishes. In addition, you need to calculate the number of such subsets modulo 10^9+7.

Formally, given a tree of N vertices and a positive integer K, determine if there is a subset of vertices such that any path containing exactly K vertices has exactly one vertex from the subset. Additionally, you are asked to find the number of such subsets modulo 10^9+7.

Input

The first line contains two integers N and K ($2 \le K \le N$) from the task description.

The next $N-1$ lines describe the hiking paths. The i-th of these lines contains two space-separated integers a_i and b_i ($1 \le a_i, b_i \le N$), representing a hiking path between shelters a_i and b_i.

It is guaranteed that these paths form a tree.

Output

On the first line, output `"YES"` if there exists a subset of shelters satisfying Vila's conditions and `"NO"`, otherwise.

On the second line output the number of possible subsets of shelters satisfying Vila's conditions modulo 10^9+7.

Scoring

Subtask	Score	Constraints
1	30	$2 \le K \le N \le 200$
2	20	$2 \le K \le N \le 10\,000$
3	20	$2 \le K \le N \le 500\,000$
4	30	$2 \le K \le N \le 1\,500\,000$

If your program outputs the correct first line, but does not provide the correct second line, that test case will be scored with 60% of points allocated for the subtask it is part of.

The score for each subtask equals the smallest score obtained by one of its test cases.

Examples

input

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

output

```
YES
2
```

input

```
8 3
7 3
1 3
7 8
5 1
4 6
7 2
3 6
```

output

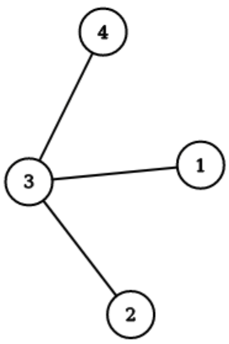
```
NO
0
```

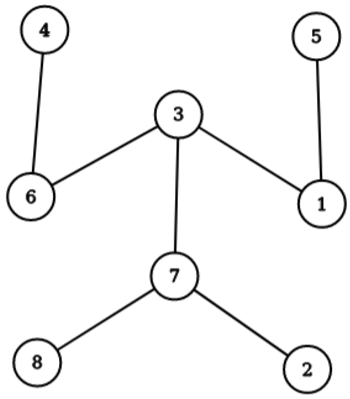
input

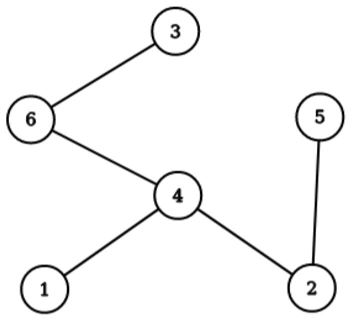
```
6 5
4 1
4 2
3 6
5 2
4 6
```

output

```
YES
10
```







Clarification of the first example: The valid subsets of shelters are $\{3\}$ and $\{1, 2, 4\}$.

Clarification of the third example: There is only one path of length 5, and that path contains nodes 3, 6, 4, 2, and 5. Exactly one of these nodes must be in a sought subset, and it makes no difference whether node 1 is in the subset.

Therefore, the valid subsets of shelters are $\{3\}$, $\{1, 3\}$, $\{6\}$, $\{1, 6\}$, $\{4\}$, $\{1, 4\}$, $\{2\}$, $\{1, 2\}$, $\{5\}$, $\{1, 5\}$.