Problem A. Orthogonal Circles

Input file:	`stdin`
Output file:	`stdout`
Time limit:	1 second
Memory limit:	256 megabytes

Two circles are *orthogonal*, if they intersect and for any point of intersection their tangent lines at that point are perpendicular.

Consider a set of n circles on the plane. You are to find a circle orthogonal to all of them.

Input

The input file contains an integer n ($1 \le n \le 10^5$), followed by n triples of integers: x_i, y_i, r_i, denoting the center coordinates and the radii of the circles ($-10^6 \le x_i, y_i \le 10^6$, $1 \le r_i \le 10^6$).

The circles may coincide.

Output

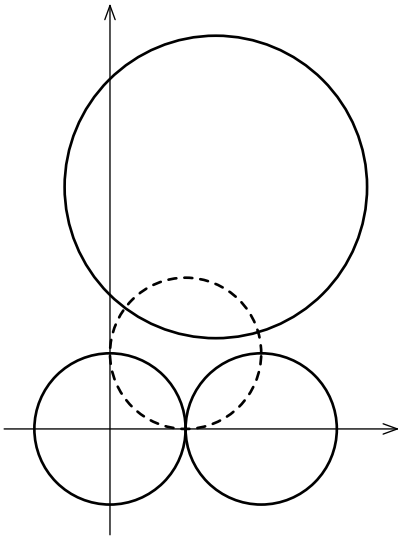
If there exists exactly one circle orthogonal to all the given ones, output its center coordinates and radius as real numbers separated with single spaces. The numbers will be considered correct if they are within 10^{-6} relative or absolute error of the exact answers.

In case there's no such circle, output -1. In case there are many, output -2.

Example

stdin	stdout
3 0 0 5 7 16 10 10 0 5	5.0 5.0 5.0

The example is illustrated by the following picture:



Problem B. Divisibility Tree

Input file:	`divisibility-tree.in`
Output file:	`divisibility-tree.out`
Time limit:	1 second
Memory limit:	256 megabytes

A rooted tree is called a *divisibility tree* when every node of the tree is assigned a positive integer, such that the number assigned to a parent of each node is *strictly* less than the number assigned to the node, and divides it evenly.

Given a rooted tree with numbers assigned to its leaves (nodes with no children), you need to assign numbers to all other nodes in such a way that the tree becomes a divisibility tree.

Input

The first line of the input file contains an integer n, denoting the number of nodes in the tree, $1 \le n \le 1000$.

The next n lines describe the nodes of the tree. The i-th node is described with 2 integers p_i and a_i. The value of p_i is the 1-based index of the parent of this node (or -1 if this node is the root). Additionally, p_i is always less than i. The value of a_i is the number assigned to this node, or -1 if no number is assigned yet. Numbers will be assigned to all leaves, and to no other nodes.

The root of the tree is the first node. All assigned numbers don't exceed 10^9.

Output

Output n positive integers separated by spaces — the numbers assigned to the nodes. If there are several possible solutions, output any one. If there is no solution, output n times -1.

Examples

divisibility-tree.in	divisibility-tree.out
5 -1 -1 1 2 1 -1 3 6 3 8	1 2 2 6 8
3 -1 -1 1 -1 2 2	-1 -1 -1

C. Removing Columns

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

You are given an $n \times m$ rectangular table consisting of lower case English letters. In one operation you can completely remove one column from the table. The remaining parts are combined forming a new table. For example, after removing the second column from the table

```
abcd
edfg
hijk
```

we obtain the table:

```
acd
efg
hjk
```

A table is called good if its rows are ordered from top to bottom lexicographically, i.e. each row is lexicographically no larger than the following one. Determine the minimum number of operations of removing a column needed to make a given table good.

Input

The first line contains two integers — n and m ($1 \le n, m \le$ **100**).

Next n lines contain m small English letters each — the characters of the table.

Output

Print a single number — the minimum number of columns that you need to remove in order to make the table good.

Examples

input
1 10 codeforces
output
0

input
4 4 case care test code
output

```
2
```

input

```
5 4
code
forc
esco
defo
rces
```

output

```
4
```

Note

In the first sample the table is already good.

In the second sample you may remove the first and third column.

In the third sample you have to remove all the columns (note that the table where all rows are empty is considered good by definition).

Let strings s and t have equal length. Then, s is lexicographically larger than t if they are not equal and the character following the largest common prefix of s and t (the prefix may be empty) in s is alphabetically larger than the corresponding character of t.

D. Eels

time limit per test: 3 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Vasya is a big fish lover, and his parents gave him an aquarium for the New Year. Vasya does not have a degree in ichthyology, so he thinks that filling a new aquarium with eels is a good idea. Unfortunately, eels are predators, so Vasya decided to find out how dangerous this idea was.

Getting into one aquarium, eels fight each other until exactly one fish remains. When two eels fight, the big one eats the smaller one (if their weights are equal, then one of them will still eat the other). Namely, let n eels be initially in an aquarium, and the i-th of them have a weight of x_i. Then $n-1$ battles will occur between them, as a result of which, only one eel will survive. In a battle of two eels with weights a and b, where $a \leq b$, eel of weight a will be eaten and disappear from the aquarium, and eel of weight b will increase its weight to $a+b$.

A battle between two eels with weights a and b, where $a \leq b$, is considered *dangerous* if $b \leq 2a$. For a given set of eels, *danger* is defined as the maximum number of dangerous battles that can occur among these eels if they are placed in one aquarium.

Now Vasya is planning, which eels he wants to put into an aquarium. He has some set of eels (initially empty). He makes a series of operations with this set. With each operation, he either adds one eel in the set, or removes one eel from the set. Vasya asks you to calculate the danger of the current set of eels after each operation.

Input

The first line of input contains a single integer q ($1 \leq q \leq 500\,000$), the number of operations that Vasya makes. The next q lines describe operations. Each operation has one of two types :

- + x describes the addition of one eel of weight x to the set ($1 \leq x \leq 10^9$). Note that in the set there can be several eels of the same weight.
- - x describes the removal of one eel of weight x from a set, and it is guaranteed that there is a eel of such weight in the set.

Output

For each operation, output single integer, the danger of the set of eels after this operation.

Examples

input
2 + 1 - 1
output
0 0

input
4 + 1 + 3 + 7 - 3
output

```
0
0
1
0
```

input

```
9
+ 2
+ 2
+ 12
- 2
- 2
+ 4
+ 1
+ 1
- 12
```

output

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

Note

In the third example, after performing all the operations, the set of eels looks like $\{1, 1, 4\}$. For this set of eels, there are several possible scenarios, if all of them are placed in one aquarium:

- The eel of weight 4 eats the eel of weight 1, and then the second eel of weight 1. In this case, none of the battles are dangerous.
- The eel of weight 1 eats the eel of weight 1, and this battle is dangerous. Now there are two eels in the aquarium, their weights are 4 and 2. The big one eats the small one, and this battle is also dangerous. In this case, the total number of dangerous battles will be 2.

Thus, the danger of this set of eels is 2.

E. Cats Transport

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Zxr960115 is owner of a large farm. He feeds m cute cats and employs p feeders. There's a straight road across the farm and n hills along the road, numbered from 1 to n from left to right. The distance between hill i and $(i - 1)$ is d_i meters. The feeders live in hill 1.

One day, the cats went out to play. Cat i went on a trip to hill h_i, finished its trip at time t_i, and then waited at hill h_i for a feeder. The feeders must take all the cats. Each feeder goes straightly from hill 1 to n without waiting at a hill and takes all the **waiting** cats at each hill away. Feeders walk at a speed of 1 meter per unit time and are strong enough to take as many cats as they want.

For example, suppose we have two hills $(d_2 = 1)$ and one cat that finished its trip at time 3 at hill 2 $(h_1 = 2)$. Then if the feeder leaves hill 1 at time 2 or at time 3, he can take this cat, but if he leaves hill 1 at time 1 he can't take it. If the feeder leaves hill 1 at time 2, the cat waits him for 0 time units, if the feeder leaves hill 1 at time 3, the cat waits him for 1 time units.

Your task is to schedule the time leaving from hill 1 for each feeder so that the sum of the waiting time of all cats is minimized.

Input

The first line of the input contains three integers n, m, p $(2 \le n \le 10^5, 1 \le m \le 10^5, 1 \le p \le 100)$.

The second line contains $n - 1$ positive integers $d_2, d_3, ..., d_n$ $(1 \le d_i < 10^4)$.

Each of the next m lines contains two integers h_i and t_i $(1 \le h_i \le n, 0 \le t_i \le 10^9)$.

Output

Output an integer, the minimum sum of waiting time of all cats.

Please, do not write the `%lld` specifier to read or write 64-bit integers in C++. It is preferred to use the `cin`, `cout` streams or the `%I64d` specifier.

Examples

input
4 6 2 1 3 5 1 0 2 1 4 9 1 10 2 10 3 12
output
3

Problem F. $\sqrt{Nim}$

Input file:	`nim.in`
Output file:	`nim.out`
Time limit:	2 seconds
Memory limit:	256 megabytes

Square Root Nim game is played using the following rules. Two players have a pile of n stones. They alternatively take stones from the pile. Each turn if the pile contains k stones the player can take from 1 to $\lfloor\sqrt{k}\rfloor$ stones, inclusive, from it. For example, if there are 10 stones, the player can take 1, 2 or 3 stones. If there are no stones left, the player who must make a turn loses.

Given n, find out whether the first player to make a turn would win if both players play optimally.

Input

The input file contains one integer number n ($1 \le n \le 10^{12}$).

Output

Output "`WIN`" if the first player will win, or "`LOSE`" if the first player will lose.

Example

nim.in	nim.out
3	WIN
5	LOSE

G. QueryreuQ

time limit per test: 1 second
memory limit per test: 1024 megabytes
input: standard input
output: standard output

A string is **palindrome**, if the string reads the same backward and forward. For example, strings like "a", "aa", "appa", "queryreuq" are all palindromes.

For given empty string S, you should process following two queries :

1. Add a lower case alphabet at the back of S.
2. Remove a character at the back of S.

After processing a query, you should count the number of **palindrome substring** in S. For string S and integers $i,\ j$ $(1 \le i \le j \le |S|)$, $S[i,\ j]$ represents a substring from ith to jth character of S. You should print out the number of integer pairs $(i,\ j)$ where $S[i,\ j]$ is palindrome.

Input

Input consists of two lines.

In the first line, Q, the number of queries is given. $(1 \le Q \le 10,000)$

In the second line, the query is given as string of length Q. ith character K_i denotes the ith query.

K_i is '-' or lower case alphabet ('a', 'b', ..., 'z') (without quotes).

If the character is '-', you should remove a character at the back of S. If the character is lower case alphabet, you should add a character K_i at the back of S.

It is guaranteed that length of S is always positive after the query.

Output

Print out Q space-separated integers in the first line. i-th integer should be the answer of the ith query.

Example

input
17 qu-uer-ryr-reu-uq
output
1 2 1 2 3 4 3 4 5 7 5 7 9 11 9 11 13

H. Matrix, The

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

...Neo finally understood how The Matrix is working.

The Matrix is the square grid $n \times n$; each cell of the grid contains 0 or 1. Next additional limitations are working for The Matrix:

- Each column contains at least one 1; ones in the column are "continuous", i.e. no zeros between any two ones in the column can be found.
- Each line number of ones is between a and b, inclusively

The Matrix changes with time following next law: lets define state of The Matrix as sequence of n^2 elements ($a_{1,1}, ..., a_{1,n}, a_{2,1}, ..., a_{n,n}$), obtained from The Matrix by writing all its lines one by one. Then all states, corresponding to correct (i.e. conforming with limitations above) instances of The Matrix, are ordered lexicographcally and in time t The Matrix have t-th state in the resulting list.

Neo is sure that he can reconstruct the Matrix in any moment of time. Can you do it?

Input

First line of the input contains four integers n, a, b and q ($1 \le n \le 10$, $1 \le a \le b \le n$, $1 \le q \le 1000$) — dimension of The Matrix, parameters of The Matrix and number of queries, respectively. i-th of the next q lines contains one integer t_i — some moment of time ($1 \le t_i \le 10^{18}$).

Output

For i-th request print n lines, each containing n characters — The Matrix in the time t_i. If total number of correct instances of the Matrix is less than t_i, print "`No such matrix.`" without quotes. Separate answers on different requests with newline. Follow the sample for clarify.

Example

input
3 2 3 5 1 2 16 34 35

output
011 011 101 011 011 110 101 111 101 111 111 111

```
No such matrix.
```

I. Politics

time limit per test: 4 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

There are n cities in the country.

Two candidates are fighting for the post of the President. The elections are set in the future, and both candidates have already planned how they are going to connect the cities with roads. Both plans will connect all cities using $n - 1$ roads only. That is, each plan can be viewed as a tree. Both of the candidates had also specified their choice of the capital among n cities (x for the first candidate and y for the second candidate), which may or may not be same.

Each city has a potential of building a port (one city can have at most one port). Building a port in i-th city brings a_i amount of money. However, each candidate has his specific demands. The demands are of the form:

- $k\ x$, which means that the candidate wants to build exactly x ports in the subtree of the k-th city of his tree (the tree is rooted at the capital of his choice).

Find out the maximum revenue that can be gained while fulfilling all demands of both candidates, or print `-1` if it is not possible to do.

It is additionally guaranteed, that each candidate has specified the port demands for the capital of his choice.

Input

The first line contains integers n, x and y ($1 \le n \le 500$, $1 \le x, y \le n$) — the number of cities, the capital of the first candidate and the capital of the second candidate respectively.

Next line contains integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 100\,000$) — the revenue gained if the port is constructed in the corresponding city.

Each of the next $n - 1$ lines contains integers u_i and v_i ($1 \le u_i, v_i \le n$, $u_i \neq v_i$), denoting edges between cities in the tree of the first candidate.

Each of the next $n - 1$ lines contains integers u'_i and v'_i ($1 \le u'_i, v'_i \le n$, $u'_i \neq v'_i$), denoting edges between cities in the tree of the second candidate.

Next line contains an integer q_1 ($1 \le q_1 \le n$), denoting the number of demands of the first candidate.

Each of the next q_1 lines contains two integers k and x ($1 \le k \le n$, $1 \le x \le n$) — the city number and the number of ports in its subtree.

Next line contains an integer q_2 ($1 \le q_2 \le n$), denoting the number of demands of the second candidate.

Each of the next q_2 lines contain two integers k and x ($1 \le k \le n$, $1 \le x \le n$) — the city number and the number of ports in its subtree.

It is guaranteed, that given edges correspond to valid trees, each candidate has given demand about each city at most once and that each candidate has specified the port demands for the capital of his choice. That is, the city x is always given in demands of the first candidate and city y is always given in the demands of the second candidate.

Output

Print exactly one integer — the maximum possible revenue that can be gained, while satisfying demands of both candidates, or `-1` if it is not possible to satisfy all of the demands.

Examples

input

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

output

```
9
```

input

```
5 1 1
3 99 99 100 2
1 2
1 3
3 4
3 5
1 3
1 2
2 4
2 5
2
1 2
3 1
2
1 2
2 1
```

output

```
198
```

input

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

output

```
-1
```

Note

In the first example, it is optimal to build ports in cities 2, 3 and 4, which fulfills all demands of both candidates and gives revenue equal to $2 + 3 + 4 = 9$.

In the second example, it is optimal to build ports in cities 2 and 3, which fulfills all demands of both candidates and gives revenue equal to $99 + 99 = 198$.

In the third example, it is not possible to build ports in such way, that all demands of both candidates are specified, hence the answer is `-1`.

J. Primes on Interval

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

You've decided to carry out a survey in the theory of prime numbers. Let us remind you that a prime number is a positive integer that has exactly two distinct positive integer divisors.

Consider positive integers a, $a + 1$, ..., b ($a \le b$). You want to find the minimum integer l ($1 \le l \le b - a + 1$) such that for any integer x ($a \le x \le b - l + 1$) among l integers x, $x + 1$, ..., $x + l - 1$ there are at least k prime numbers.

Find and print the required minimum l. If no value l meets the described limitations, print -1.

Input

A single line contains three space-separated integers a, b, k ($1 \le a, b, k \le 10^6$; $a \le b$).

Output

In a single line print a single integer — the required minimum l. If there's no solution, print -1.

Examples

input
2 4 2
output
3

input
6 13 1
output
4

input
1 4 3
output
-1

K. Eugeny and Play List

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Eugeny loves listening to music. He has n songs in his play list. We know that song number i has the duration of t_i minutes. Eugeny listens to each song, perhaps more than once. He listens to song number i c_i times. Eugeny's play list is organized as follows: first song number 1 plays c_1 times, then song number 2 plays c_2 times, ..., in the end the song number n plays c_n times.

Eugeny took a piece of paper and wrote out m moments of time when he liked a song. Now for each such moment he wants to know the number of the song that played at that moment. The moment x means that Eugeny wants to know which song was playing during the x-th minute of his listening to the play list.

Help Eugeny and calculate the required numbers of songs.

Input

The first line contains two integers n, m ($1 \le n, m \le 10^5$). The next n lines contain pairs of integers. The i-th line contains integers c_i, t_i ($1 \le c_i, t_i \le 10^9$) — the description of the play list. It is guaranteed that the play list's total duration doesn't exceed 10^9 ($\sum_{i=1}^{n} c_i \cdot t_i \le 10^9$).

The next line contains m positive integers $v_1, v_2, ..., v_m$, that describe the moments Eugeny has written out. It is guaranteed that there isn't such moment of time v_i, when the music doesn't play any longer. It is guaranteed that $v_i < v_{i+1}$ ($i < m$).

The moment of time v_i means that Eugeny wants to know which song was playing during the v_i-th munite from the start of listening to the playlist.

Output

Print m integers — the i-th number must equal the number of the song that was playing during the v_i-th minute after Eugeny started listening to the play list.

Examples

input
1 2 2 8 1 16
output
1 1

input
4 9 1 2 2 1 1 1 2 2 1 2 3 4 5 6 7 8 9
output
1 1

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

L. Hidden Bipartite Graph

time limit per test: 4 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Bob has a simple undirected connected graph (without self-loops and multiple edges). He wants to learn whether his graph is bipartite (that is, you can paint all vertices of the graph into two colors so that there is no edge connecting two vertices of the same color) or not. As he is not very good at programming, he asked Alice for help. He does not want to disclose his graph to Alice, but he agreed that Alice can ask him some questions about the graph.

The only question that Alice can ask is the following: she sends s — a subset of vertices of the original graph. Bob answers with the number of edges that have both endpoints in s. Since he doesn't want Alice to learn too much about the graph, he allows her to ask no more than 20000 questions. Furthermore, he suspects that Alice might introduce false messages to their communication channel, so when Alice finally tells him whether the graph is bipartite or not, she also needs to provide a proof — either the partitions themselves or a cycle of odd length.

Your task is to help Alice to construct the queries, find whether the graph is bipartite.

Input

The first line contains a single integer n ($1 \le n \le 600$) — the number of vertices in Bob's graph.

Interaction

First, read an integer n ($1 \le n \le 600$) — the number of vertices in Bob's graph.

To make a query, print two lines. First of which should be in the format "? k" ($1 \le k \le n$), where k is the size of the set to be queried. The second line should contain k space separated distinct integers $s_1, s_2, \ldots, s_k$ ($1 \le s_i \le n$) — the vertices of the queried set.

After each query read a single integer m ($0 \le m \le \frac{n(n-1)}{2}$) — the number of edges between the vertices of the set $\{s_i\}$.

You are not allowed to ask more than 20000 queries.

If $m = -1$, it means that you asked more queries than allowed, or asked an invalid query. Your program should immediately terminate (for example, by calling exit(0)). You will receive `Wrong Answer`; it means that you asked more queries than allowed, or asked an invalid query. If you ignore this, you can get other verdicts since your program will continue to read from a closed stream.

After printing a query do not forget to print end of line and flush the output. Otherwise, you will get `Idleness limit exceeded`. To do this, use:

- `fflush(stdout)` or `cout.flush()` in C++;
- `System.out.flush()` in Java;
- `flush(output)` in Pascal;
- `stdout.flush()` in Python;
- see documentation for other languages.

When you know the answer, you need to print it.

The format of the answer depends on whether the graph is bipartite or not.

If the graph is bipartite, print two lines. The first should contain the letter "Y" (short for "YES") followed by a space, and then a single integer s ($0 \le s \le n$) — the number of vertices in one of the partitions. Second line should contain s integers $a_1, a_2, \ldots, a_s$ — vertices belonging to the first partition. All a_i must be distinct, and all edges in the main graph must have exactly one endpoint in the set $\{a_i\}$.

If the graph is not bipartite, print two lines. The first should contain the letter "N" (short for "NO") followed by a space, and then a single integer l ($3 \le l \le n$) — the length of one simple cycle of odd length. Second line should contain l integers $c_1, c_2, \ldots, c_l$ — the vertices along the cycle. It must hold that for all $1 \le i \le l$, there is an edge $\{c_i, c_{(i \mod l)+1}\}$ in the main graph, and all c_i are distinct.

If there are multiple possible answers, you may print any of them.

Hacks format For hacks, use the following format:

The first line contains two integers n and m ($1 \le n \le 600, 0 \le m \le \frac{n(n-1)}{2}$) — the number of vertices and edges of the graph, respectively.

Each of the next m lines contains two integers u_i and v_i ($1 \le u_i, v_i \le n$) mean that there is an edge between u_i and v_i. There must not be any multiple edges, no loops, and the graph must be connected.

For example, you need to use this test to get the first sample:

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

Examples

input
4 4 0 1 1 1 0

output
? 4 1 2 3 4 ? 2 1 2 ? 2 1 3 ? 2 1 4 ? 2 2 4 ? 2 3 4 Y 2 1 2

input
4 4 3

output

```
? 4
1 4 2 3
? 3
1 2 4
N 3
2 1 4
```

Note

In the first case, Alice learns that there are 4 edges in the whole graph. Over the course of the next three queries, she learns that vertex 1 has two neighbors: 3 and 4. She then learns that while vertex 2 is adjacent to 4, the vertex 3 isn't adjacent to 4. There is only one option for the remaining edge, and that is $(2, 3)$. This means that the graph is a cycle on four vertices, with $(1, 2)$ being one partition and $(3, 4)$ being the second. Here, it would be also valid to output "3 4" on the second line.

In the second case, we also have a graph on four vertices and four edges. In the second query, Alice learns that there are three edges among vertices $(1, 2, 4)$. The only way this could possibly happen is that those form a triangle. As the triangle is not bipartite, Alice can report it as a proof. Notice that she does not learn where the fourth edge is, but she is able to answer Bob correctly anyway.

M. Red and Black Tree

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

You have a weighted tree, consisting of n vertices. Each vertex is either painted black or is painted red. A red and black tree is called *beautiful*, if for any its vertex we can find a black vertex at distance at most x.

The distance between two nodes is the shortest path between them.

You have a red and black tree. Your task is to make it beautiful in the minimum number of color swap operations. In one color swap operation, you can choose two vertices of different colors and paint each of them the other color. In other words, if you choose a red vertex p and a black vertex q, then in one operation you are allowed to paint p black and paint q red.

Print the minimum number of required actions.

Input

The first line contains two integers n and x ($2 \le n \le 500$; $1 \le x \le 10^9$). The next line contains n integers, each of them is either a zero or one. If the i-th number equals 1, then vertex i of the tree is black, otherwise vertex i is red. Next $n - 1$ lines contain the tree edges. The j-th line contains integers u_j v_j w_j ($1 \le u_j, v_j \le n$; $u_j \ne v_j$; $1 \le w_j \le 10^9$) which means that the tree has an edge of weight w_j between vertices v_j and u_j.

Assume that the tree vertices are numbered from 1 to n.

Output

Print a single integer — the minimum number of required swap operations.

If it is impossible to get a beautiful tree at any number of operations, print -1.

Examples

input
3 2 1 0 0 1 2 2 2 3 2
output
1

input
4 2 0 1 0 0 1 2 2 2 3 2 3 4 2
output
-1