Problem A. Altitude

Input file:	standard input
Output file:	standard output
Time limit:	1 second
Memory limit:	512 megabytes

In a new computer game "Take Them All" the player controls a drone that flies forward and has to pass through all the checkpoints on its way. Each checkpoint is located at some altitude (possibly negative), and the player has to adjust the altitude of the drone in order to pass through this checkpoint.

You are given a sequence of n **distinct** integers $a_1, a_2, \ldots, a_n$. The i-th checkpoint is located at altitude a_i. The level is cyclic: the checkpoint with index n is followed by the checkpoint with index 1 again. The triple of indices (not necessarily distinct) i, j and k is *tricky* if $a_i < a_j > a_k$.

The *length* of the triple i, j and k is defined as the minimum number of segments between neighbouring checkpoints that the drone needs to fly through to get from i to j and then from j to k. Formally, we denote d_1 as $j - i$ if $j > i$ and $n - (i - j)$ if $j \le i$ and d_2 as $k - j$ if $k > j$ and $n - (j - k)$ if $k \le j$. Then, the length of the triple i, j and k is equal to $d_1 + d_2$.

The goal of this task is to find a tricky triple of the minimum possible length.

Input

The first line of input contains a single integer n ($2 \le n \le 100\,000$) — the number of checkpoints.

The second line contains the sequence of checkpoint altitudes $a_1, a_2, \ldots, a_n$ ($-10^9 \le a_i \le 10^9$). It is guaranteed that all values a_i are distinct.

Output

Output three integers i, j and k ($1 \le i, j, k \le n$) that define the tricky triple of the minimum possible length. If there are several optimal answers, print any of them.

Examples

standard input	standard output
2 20 16	2 1 2
4 2 0 1 6	3 4 1
5 16 10 20 1 6	2 3 4

Note

Please note that triple 1, 4, 3 is not a correct answer for the second sample. This triple is tricky, but its length is bigger than the length of the triple 3, 4, 1.

Problem B. Blocking Buffer

Input file:	standard input
Output file:	standard output
Time limit:	1 second
Memory limit:	512 megabytes

New microarchitecture for parallel programming has a pipe buffer to organize the data exchange between two execution threads. The buffer is circular and can contain no more than l bytes. At the beginning of the program execution the buffer is empty. The first thread sometimes tries to write to the buffer, and it always uses blocks of size w bytes. The second thread sometimes reads from the buffer in blocks of size r bytes each. All write operations put the new data at the end of the pipe buffer and all read operations take the data from the beginning of the pipe buffer, so empty positions always form one continuous segment.

Both write and read operations are blocking. It means that, if the first thread tries to write the block of data to the buffer but there is not enough space there, the thread stops and waits until there will be at least w empty bytes in the buffer. Similarly, if the second thread tries to read the block of data from the buffer but there are less than r bytes of data left, the thread stops and waits until there will be enough new data in the buffer.

In this particular problem, a *deadlock* is a situation when the first thread is blocked because there is not enough space in the buffer to fit a new block of size w, and the second thread is blocked because there is not enough data in the buffer to read a new block of size r.

Your goal is to determine whether there exists such a sequence of reads and writes from two threads that will result in a deadlock. Note that write operations are only performed by the first thread and read operations are only performed by the second thread.

Input

The only line of input contains three integers l, r and w ($0 < l, r, w \le 10^{18}$, $r \le l$, $w \le l$).

Output

If there exists a sequence of reads and writes that result in a deadlock, print "DEADLOCK" (without quotes) in the only line of the output. Otherwise, print "OK" (without quotes).

Examples

standard input	standard output
5 3 4	DEADLOCK
5 2 3	OK

Note

One of the ways to obtain the deadlock in the first sample is:

1. The first thread writes a block of size 4 to the buffer.
2. The second thread reads a block of size 3. There is 1 byte of data left in the buffer.
3. The first thread writes a block of size 4. The buffer is full now.
4. The second thread reads a block of size 3. There are 2 bytes of data left in the buffer.
5. The second thread tries to read a block of size 3, but there is not enough data in the buffer, so the thread is blocked.
6. The first thread tries to write a block of size 4, but there it not enough free space in the buffer, so the thread is also blocked. The deadlock just happened.

Problem E. Economy Printing

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	512 megabytes

Helen has a large document consisting of 10^9 pages numbered from 1 to 10^9. She is given the list of indices of pages $p_1, p_2, \ldots, p_n$ she needs to print. Helen likes nature and cares a lot about the trees, so she wants to print exactly one copy of each of these pages and no copies of any other pages. In order to achieve this she composes the printing instruction using tokens of four types:

1. Single index "i". This token asks to print the page number i.
2. Range "i_1-i_2". This token asks to print all pages from i_1 to i_2 inclusive.
3. Even range "i_1%i_2". Here, both i_1 and i_2 should be even. This token asks to print the pages with even indices from i_1 to i_2 inclusive.
4. Odd range "i_1#i_2". Here, both i_1 and i_2 should be odd. This token asks to print the pages with odd indices from i_1 to i_2 inclusive.

Printing instruction is composed using any number of tokens of any types, separated by commas. Note that each page is printed any time it matches a token, but Helen wants each page from her list to be printed only once, and she doesn't want to print any page not from her list. As Helen also wants to save some space on her screen, she asks you to find the correct printing instruction of minimum possible length. The length of the instruction is equal to the **total number of characters** that are used to write it down.

Input

The first line of the input contains a single integer n ($1 \le n \le 200\,000$) — the number of pages Helen wants to print.

The second line contains a list of n distinct indices of pages p_i ($1 \le p_i \le 10^9$).

Output

Print one line containing the shortest possible instruction which will print all the pages from the list exactly once and won't print any other pages. If there are several optimal answers, print any one of them.

Example

standard input	standard output
15 1 18 20 5 14 11 3 16 17 8 4 10 6 15 21	11,21,20,14-18,4%10,1#5

Note

The length of the answer to the given example is 23.

Problem G. Great Guest Gathering

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	512 megabytes

Pizza Factory has decided to host a great guest gathering for promotional purposes and has invited n their most valuable guests. The chef has prepared n plates, each having exactly one pizza on it. All pizzas are of different types. The chef doesn't know the preferences of guests, so he wants everyone to be able to taste pizza of each type.

In order achieve that, he divided each pizza into n equal slices, and now he wants to shuffle them into n "festival" pizzas, each containing one slice of each type. He has one additional small plate that can hold only one slice of any pizza.

One chef's move can be either:

- Taking a slice of pizza from some plate and placing it on the free space on another plate. Note that each plate can contain no more than n slices simultaneously.
- Taking a slice of pizza from some plate and placing it on the additional small plate, if this plate is free.
- Taking a slice of pizza from the additional small plate and placing it on the free space on another plate.

Your task is to help chef come up with the shortest possible list of moves he has to make in order to prepare n "festival" pizzas.

Input

The first line of the input contains a single integer n ($2 \le n \le 100$).

Output

Print the shortest list of moves that is required to make n "festival" pizzas.

Print each move on a separate line of output as three integers a, b and c ($0 \le a, c \le n$, $1 \le b \le n$). A line containing "a b c" means that the chef moves a slice belonging to pizza b from plate a to plate c, where plate 0 denotes the additional small plate.

Example

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

Note

In the first sample, any win should be sufficient for the team "Russia".

Consider the fourth sample.

Team	Points	Total Wins	Prime Time Wins	Goal Differential	Goals Scored
Sweden	6	3	3	6	6
NA	2	1	1	0	1
Finland	2	1	1	2	4
Russia	0	0	0	-8	0

If the Russian team wins 11 to 0 it qualifies to the knockout stage. If the team "Russia" team wins 10 to 0, then three teams use tiebreaking rules for places 2–4. The total number of wins and the total number of wins in prime time are equal for all these teams. By goal differential, the team "NA" takes the fourth place. After that, tiebreaking rules are applied again to determine the second place. Team "Finland" is better as it won the game between the two teams (the fact that team "Russia" scored more is ignored).

Consider the fifth sample.

Team	Points	Total Wins	Prime Time Wins	Goal Differential	Goals Scored
Finland	6	3	3	30	30
Sweden	2	1	1	-9	1
Russia	2	1	1	-10	1
NA	0	0	0	-11	0

The only outcome that gives team "Russia" a chance to advance to the knockout stage is win of team "NA" over team "Sweden" in regular time with score 1 to 0. In this case, all three teams will have identical results, and team "Russia" has a chance to get to the knockout stage by random.

Problem I. Interesting Interactive Idea

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	512 megabytes

This is an interactive problem.

The jury has chosen two integers x_0 and m ($0 \le x_0 < m \le n$). You are given a number n: the upper limit for m.

Your task is to guess both integers x_0 and m.

In order to do that, your program will ask questions in the form "a_i" ($1 \le a_i \le n$) where $i = 1, 2, \ldots$ is the index of the question.

After each question, the jury calculates $x_i = (x_{i-1} + a_i) \mod m$ and tells you the result of comparison between x_i and x_{i-1} ('>', '<' or '=').

The last question must be in the form "`0` x_0 m".

Your program must guess the numbers x_0 and m by asking no more than 2016 questions.

Input

The first line of the input contains a single integer n ($1 \le n \le 10^{18}$). After that, for each new question in the form "a_i", the input will contain a line with a single character: the result of comparison between x_i and x_{i-1}. If $x_i > x_{i-1}$ then the result is '>', if $x_i < x_{i-1}$ then the result is '<', otherwise the result is '='.

Output

Your program must output each question on a separate line. Each question except the last one must consist of a single integer a_i ($1 \le a_i \le n$). The last question must be in the form "`0` x_0 m" ($0 \le x_0 < m \le n$).

Example

standard input	standard output
2 > < 	1 1 1 0 1 2

Note

The pipe from your program to the interactor program and the pipe back have limited size. Your program must read from the standard input to avoid deadlock. Deadlock condition is reported as "`Time Limit Exceeded`".

To flush the standard output stream, use the following statements:

In C, use `fflush(stdout);` in C++, use `cout.flush();` in Java, use `System.out.flush();` in Python, use `sys.stdout.flush()`.

If your program receives an `EOF` (end-of-file) condition on the standard input, it MUST exit immediately with exit code 0. Failure to comply with this requirement may result in "`Time Limit Exceeded`" error.

Problem J. Juice Degustation

Input file:	standard input
Output file:	standard output
Time limit:	1 second
Memory limit:	512 megabytes

Vasya plans a juice degustation party at his house. There are n different kinds of juice from Vasya's favorite parts of the world, such as Loire Valley in France, Napa Valley in California and Valle Central in Chili. He has bought c_i liters of juice of kind i.

To store the juice till the beginning of the party, Vasya needs to buy some special barrels. Each barrel is large enough to fit any amount of juice. Then, he distributes the different kinds of juice between the barrels following the special rules below:

1. All the juice of all kinds should be distributed into barrels.
2. Each barrel may contain juice of no more than two different kinds.
3. Any two barrels should have the same total amount of juice.

Note that the amount of juice in the barrels might be non-integer. Some barrels are allowed to contain only one kind of juice. Each kind of juice can be used in any number of barrels. Two different kinds of juice can be mixed in a barrel in any proportion.

The barrels are expensive, therefore Vasya wants to buy the minimum number of them in order to be able to store the juice following the above rules. Help him find this number.

Input

The first line of the input contains an integer n ($1 \le n \le 20$) — the number of different kinds of juice Vasya likes.

The second line contains n integers $c_1, c_2, \ldots, c_n$ ($1 \le c_i \le 10^9$), the i-th of them shows how many liters of i-th kind of juice Vasya has.

Output

Print one integer: the minimum number of barrels Vasya has to buy in order to be able to store all his juice following the above rules.

Examples

standard input	standard output
3 1 1 1	2
5 1 2 1 1 1	3
1 100	1

Note

In the first sample, Vasya has 3 kinds of juice, 1 liter of each kind. He can put all the juice of the first kind in the first barrel, all the juice of the second kind in the second barrel, and add 0.5 liters of juice of the third kind in each of the barrels.

Problem K. Knights of the Old Republic

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	512 megabytes

A long time ago in a galaxy far, far away a group of Jedi from the Resistance forces had to destroy the Death Star. The Death Star consists of n guarded command centers and m bidirectional pathways connecting them. Each pathway connects two (not necessarily distinct) command centers. A pair of command centers may be connected by two or more pathways. The goal of the Resistance is to capture all command centers. Note that it is **not** necessary to capture all pathways.

For any command center i, you are given the number of resources b_i required to transfer one Jedi there (thus, for sending k Jedi you need to spend $b_i \cdot k$ resources). To capture the command center i, the Resistance forces need to gather at least a_i Jedi there. To capture the pathway j, they need to simultaneously gather at least c_j Jedi cumulatively at the endpoints of this pathway. All captures happen without any casualties: none of the Jedi die and they all can fight for other command centers and pathways.

After capturing a pathway or a command center, Jedi can move through it for free. To capture a command center Jedi need to be either sent there or walk there from other command centers through already captured pathways. To capture a pathway, Jedi do not have to unite in one command center: they can attack the pathway from both endpoints (but only if the total number of Jedi at both endpoints is at least c_i).

Find the minimum number of resources required to capture all the command centers.

Input

The first line of the input contains two integers n and m ($1 \le n, m \le 300\,000$) — the number of command centers and pathways respectively.

Each of the next n lines contains two integers a_i and b_i ($0 \le a_i, b_i \le 1\,000\,000$) providing the minimum number of Jedi that can capture the i-th command center and the amount of resources required to send one Jedi directly to the i-th command center.

Next m lines describe the pathways. Each of them contains three integers s_i, f_i and c_i ($1 \le s_i, f_i \le n$, $0 \le c_i \le 1\,000\,000$) — the numbers of command centers connected by the i-th pathway and the minimum number of Jedi required to capture the i-th pathway.

Output

Print one integer: the minimum number of resources required to capture all the command centers of the Death Star.

Example

standard input	standard output
3 2 10 5 20 10 10 3 1 2 22 2 3 200	140