Problem A. Chess

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 megabytes

A legend says that, as a reward for inventing chess, the creator asked his ruler for 1 grain of rice on the first square, 2 on the second, 4 on the third, and on each of the following twice the amount of the preceding square. The ruler at first laughed it off as a meager prize for such a remarkable invention, but soon (around the 3rd rank, where he needed more than millions of grains) realized his mistake.

Paul invented a new version of chess, that uses N squares and assigned each square a value the same way. However, for convenience, he used all the numbers modulo some positive integer M.

You have recovered a sorted sequence of these numbers $\{A_i\}$, and you want to find which M was used.

Input

First line contains a single number $1 \leq N \leq 2 \cdot 10^5$, the number of squares on the chessboard. The next line contains N sorted numbers $0 \leq A_i \leq 10^{18}$, representing the values written on squares. It is guaranteed that a solution exists.

Output

Output a single line with a single integer M, the modulus that was used for the original calculation. If there are multiple solutions, print the smallest one.

Examples

standard input	standard output
5 1 2 3 4 8	13
2 1 2	3
8 1 1 1 2 2 2 4 4	7
24 1 2 4 8 16 32 64 128 256 512 1024 2048 4096 8192 12149 15537 16384 24298 31074 32768 38775 44387 47193 48596	49999

Note

In the first test case, the numbers written on squares, in order, are 1, 2, 4, 8 and 3 ($2 \cdot 8 \bmod 13$).

In the second test case, the smallest possible solution is 3, but other numbers, like 7 or 11, would also work.

In the third case, note that the numbers can repeat and it is possible for N to be larger than M.

Fourth case sees a well-known prime in action.

Problem B. Triangle in a Triangle

Input file:	`standard input`
Output file:	`standard output`
Time limit:	4 seconds
Memory limit:	64 megabytes

This year's public viewing of SWERC will take place on the famous Triangular Plaza, which is delimited by three perfectly straight roads. Due to weather concerns you have been hired to design a rain cover for the event. This cover should of course be triangular (as is tradition for the Triangular Plaza) and cover as much of the Plaza as possible. The only other restriction is that the rain cover's corners must be fixed on 3 of the lampposts that are standing along the roads. Can you figure out the maximal area of the Plaza you can cover?

Input

The first three lines of input contain pairs of integers, the x- and y-coordinates of the 3 corners A, B, C of the Plaza. It is guaranteed that $0 \leq x, y \leq 10^6$. The next line contains a single integer $n \geq 1$, the number of lampposts on the line segment $\overline{AB}$. The following line contains n integers d_i in ascending order, where d_i is the distance between the corner A and the i-th lamppost. The next two lines are of the same format but for the line segment $\overline{BC}$ and the corner B. The last two lines are of the same format but for the line segment $\overline{CA}$ and the corner C. It is guaranteed that the lampposts are on the segment between two corners. A lamppost may however be in a corner. In total there will be less than 10^6 lampposts. (In case you are worried about precision: it is also guaranteed that all triangles of lampposts that have an area of at least 70% of the solution will have angles that are all > 20 degrees).

Output

Print a single floating point number, the maximal area of the Plaza that can be covered. The answer is considered correct if it is within a 10^{-6} absolute or relative error margin.

Example

standard input	standard output
0 0 6 0 6 7 2 1 5 2 2 6 1 5	12.0000000

Problem C. Candy division

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	64 megabytes

Your older sister has three kids. Whenever you go to visit her, you bring along a bag of candies for the kids. There are n candies in the bag. You want to give all the candies to the kids, but you also want to teach them a little math along the way. Therefore, you gave them not just the bag of candies but also one simple rule: each of the kids must take an integer fraction of candies in the bag. In other words, the amount of candies each kid takes must be a divisor of n.

Formally, in order to divide all the candies the kids have to find three *positive* integers a_1, a_2, a_3 such that $n = a_1 + a_2 + a_3$ and each a_i divides n.

Input

The first line of the input contains a single integer t – the number of test cases to follow. Each of the following t lines of the input contains the integer n. You may assume that $1 \leq t \leq 100$ and $1 \leq n \leq 10^{18}$.

Output

Output t lines. The k-th line will solve the k-th test case and will contain three integers a_i as specified above. If there are multiple solutions you may select an arbitrary one. If there are no solutions, output the word `'IMPOSSIBLE'` instead (quotes only for clarity).

Example

standard input	standard output
3 1 3 12	IMPOSSIBLE 1 1 1 4 6 2

Problem D. Effective network

Input file:	`standard input`
Output file:	`standard output`
Time limit:	4 seconds
Memory limit:	256 megabytes

Quite recently, in a very close galaxy cluster, there was a bunch of planets. As the galaxy expands to a vast space, the only effective way of traveling between them is to use the network of teleporters. Due to technological problems, one can only travel between certain pairs of teleporters, using so-called links. This means that in order to travel from planet A to planet B, one may need to first teleport from A to some intermediate planet before reaching B. The distance between planets A and B is the minimum number of links required to travel between them. It is guaranteed that one can travel between each pair of planets using a finite number of links, but this number may be quite large for some pairs.

You are an employee of Teleport GmbH and were tasked to set up terms and conditions for the GA Travel Card such that the customers are happy. In order to do this, you need to select a non-empty subset of planets, called promoted planets, and designate all the links between them (and only those) to be included in the GA. We will call these free links. Thanks to customer survey you know that there are three conditions you need to fulfill:

- There should be at least two promoted planets.
- It should be possible to travel between any two promoted planets A and B using only free links
- The distance between any two promoted planets A and B, using only free links, cannot be larger than $R - K$, where R is the number of promoted planets, and K is a fixed constant representing the demands of the customers.

Determine whether it is possible to find a set of promoted planets, and if so, return one set that satisfies the conditions. If there are multiple solutions, return any of them.

Input

First line contains integers $1 \leq N \leq 5000$, $N - 1 \leq M \leq \min(\frac{N(N-1)}{2}, 30000)$, $1 \leq K \leq N$ – the number of planets, the number of links, and the height of the demands, respectively.

Next M lines contain description of links. Each link is represented by two integers, $1 \leq u_i, v_i \leq N$, meaning that there is a link between planets u_i and v_i. It is guaranteed that no pair occurs more than once, and also that $u_i \neq v_i$.

It is guaranteed that one can travel between each pair of planets using a finite number of links.

Output

If there is a solution, output two lines: the first containing a single integer R – the number of promoted planets. On the second line, print R space separated integers, specifying the indices of promoted planets.

If there is no set of planets that would satisfy the conditions, output a single line containing the integer 0.

Examples

standard input	standard output
7 9 3 1 3 1 4 1 5 2 5 3 4 6 3 6 1 4 6 1 7	4 1 3 4 6
5 4 3 1 3 1 2 3 4 3 5	0
5 6 1 1 3 1 2 3 4 3 5 2 3 5 4	3 1 3 5

Note

In the first sample case, we select a set of four planets that are all pairwise connected. As $R = 4$ and $K = 3$, the conditions are satisfied.

In the second test case, there is no subset of planets satisfying the conditions.

In the third case note that the pair of planets 1 and 5 would not be a solution, as one cannot travel between them without interchange at a non-promoted planet.

Problem E. Collection

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	64 megabytes

Alan recently started collecting the cards of the well known "Famous Computer Scientists" card game by the ACM. As buying cards is starting to get expensive, he would like to start trading. To do this, he needs to know the number of duplicate cards in his collection. Write a program to help him calculate this number.

Input

The first line contains a single integer n, the number of cards in Alan's collection. The second line contains n integers c_i, where c_i is the id of the i-th card. It holds that $1 \leq n \leq 200'000$ and $0 \leq c_i \leq 10^9$.

Output

Print a single integer denoting the number of duplicates, that is the number of cards he can safely trade whilst still having the same number of unique cards.

Examples

standard input	standard output
4 127 0 0 1	1
6 1 1 1000000 1 1 1	4

Problem F. Mattress Run

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	64 megabytes

You are a big fan of a new hotel chain called Almost Complimentary Motels (ACM). You have recently noticed that this year this chain has launched a loyalty program for its most valuable customers. If you spend a certain number of nights or stays with the hotels from that chain, you will get "Diamond Spire" status with very valuable benefits, such as a guaranteed upgrade to the best room (known as "Billy's suite") on any stay.

In order to qualify for this status, you need to either accumulate 50 qualifying nights or 25 qualifying stays in a calendar year. Now, this year you are a little bit short on both of those criteria, and in order to meet them by the end of the year, you decide to make a mattress run. Around your town there are quite many hotels belonging to the ACM chain, most of them are pretty bad, and normally you wouldn't need to stay anywhere close to your home anyway. However in this case your plan is to make several reservations that will get you your status (by either stays or nights) that would cost you as little as possible.

Before you begin, there are several special rules from ACM chain that you need to know:

- You only collect qualifying nights and qualifying stays on special rates that may be unavailable some hotels on some check-in/check-out dates.
- Unlike many other chains, you may not make consecutive reservations in one hotel, i.e. the check out day for one reservation cannot coincide with the check in day for another reservation in the same hotel.
- Any two reservations should not overlap by nights, regardless of whether the hotel is the same.
- A stay with check-in day b and check-out day e gives you $e - b$ qualifying nights credits and always 1 qualifying stay credit.

Find the strategy to qualify you with minimal total cost. If there are many possible strategies you may choose either of them.

Input

On the first line you are given 3 integers: Y — the number of days remaining in the current year ($2 \le Y \le 365$), N — the number of nights required for qualification ($1 \le N \le 50$), S — the number of stays required for qualification ($1 \le S \le 25$).

The second line contains two integers: H — the number of hotels ($1 \le H \le 50$), M — the number of available rates ($1 \le M \le 5000$).

The next M lines describe all the available qualifying rates in the following way; each line contains 4 integers: h — the ID of the hotel ($1 \le h \le H$), b — the day of check-in, e — the day of check-out ($1 \le b < e \le Y$), c — the total cost of the stay on this rate ($1 \le c \le 1000000$).

Output

If achieving the status is impossible, output a single line with the word "`IMPOSSIBLE`". Otherwise, output a mattress run with minimal possible cost. The first line should contain either "`NIGHTS`" or "`STAYS`", stating which qualification criteria you are intending to meet with your mattress run. The second line should contain one integer K — the number of bookings you are going to make. The third line should contain K 1-based indices (as they are listed in the input) of the fares you are planning to use for the bookings. Those fares should follow in the chronological order of their use.

Examples

standard input	standard output
5 3 2 2 5 1 1 2 5 1 2 3 4 1 1 3 20 1 3 5 15 2 3 4 10	STAYS 2 2 5
5 3 3 1 4 1 1 2 5 1 2 3 5 1 3 4 5 1 4 5 5	IMPOSSIBLE

Problem G. Affine

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	64 megabytes

We call a function f from $\mathbb{R}^m$ to $\mathbb{R}^2$ affine if it maps straight lines to straight lines. Formally, f is affine if and only if $f(\alpha \cdot \vec{x} + (1-\alpha) \cdot \vec{y}) = \alpha \cdot f(\vec{x}) + (1-\alpha) \cdot f(\vec{y})$ for all $\alpha \in \mathbb{R}$ and $\vec{x}, \vec{y} \in \mathbb{R}^m$. (Note that this is a weaker condition than linearity: $f(x_1, x_2) = 3 \cdot x_1 + 2 \cdot x_2 + 1$ is affine but not linear.)

You are given a polygon P in the plane. Determine the smallest integer $m \geq 0$ such that there is an affine function f that maps the m-hypercube $[0,1]^m$ to P, or determine that no such m exists.

Input

The first line of the input contains the integer N ($1 \leq N \leq 100'000$), the number of given boundary points of the polygon. The i-th of the next N lines contains two integers X_i and Y_i ($-10^8 \leq X_i, Y_i \leq 10^8$), the coordinates of the i-th given boundary point of the polygon.

The counterclockwise boundary of the polygon P is obtained by connecting the given points by line segments. (X_i, Y_i) is connected to (X_{i+1}, Y_{i+1}) for $1 \leq i < N$, and (X_N, Y_N) is connected to (X_1, Y_1). All given points are distinct and the boundary of the polygon P neither intersects nor touches itself.

Output

Print `INFINITY`, if there is no m such that P is an affine image of the m-hypercube. Otherwise, print a single non-negative integer, the smallest m such that P is an affine image of the m-hypercube.

Examples

standard input	standard output
4 0 0 1 0 1 1 0 1	2
6 0 0 2 0 3 1 3 3 1 3 0 2	3
2 -100000000 29611633 100000000 29611633	1

Problem H. Compass

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	64 megabytes

The university of Algoland is located in a single huge building. It is great. It is the best building in the world. It only uses the best angles: the right angle! But the building is so huge that it is also very confusing to new students, because it is very easy to get lost.

The rector of Algoland's university, a former professor in physics, had a great idea to prevent students from getting lost in the future: He bought an incredibly strong magnet with the intention of placing it somewhere inside the building and using it as an emergency meeting point. On the first day of the semester, every student gets a free compass. With that compass the student can always tell the direction towards the magnet (the magnet is so strong that it completely dominates the earth's magnetic field). If a student gets lost, she can follow the following simple procedure to get to the emergency meeting point at the magnet's location:

- Move straight into the direction towards the magnet until you either reach the magnet, or bump into a wall.
- If your path is blocked by a wall, follow the direction alongside the wall that brings you closer towards the magnet, until you either reach the end of the wall or your path becomes orthogonal to the compass direction.
- If the wall is perfectly orthogonal to the compass direction, or you end up in a corner, you are stuck. Scream as loud as you can!

In the corner case where you want to walk parallel to a wall at the exact coordinate of a wall, you are not stopped by the wall. We assume here that you are infinitesimally small (which is true in proportion to the size of the building).

The rector now wants your help to place the magnet inside the building in such a way that every student can reach it (following the procedure above) no matter where inside the building the student gets lost. Actually, you just have to decide if this is possible or not.

Input

The first line contains an integer N, the number of points, $4 \leq N \leq 10^3$.

Each of the following N lines contains two coordinates x and y ($0 \leq x \leq 10^3$ and $0 \leq y \leq 10^3$). Each line describes a corner point of the single wall that delimits the inside from the outside of the building. The points are given in clockwise order and all the angles are guaranteed to be 90 degrees.

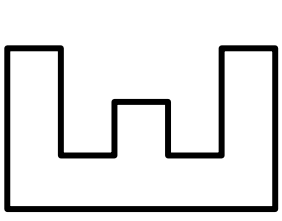
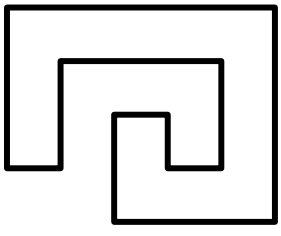
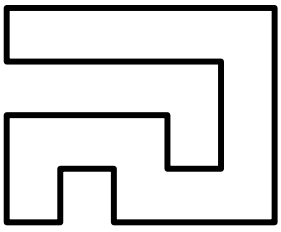
Output

Print a single line containing a single word: either `SAFETY` if it is possible to place the magnet such that one is always able to find it, or `DANGER` otherwise.

Examples

standard input	standard output
12 1 0 1 3 2 3 2 1 3 1 3 2 4 2 4 1 5 1 5 3 6 3 6 0	SAFETY
12 0 4 5 4 5 0 2 0 2 2 3 2 3 1 4 1 4 3 1 3 1 1 0 1	DANGER
14 6 4 6 0 3 0 3 1 2 1 2 0 1 0 1 2 4 2 4 1 5 1 5 3 1 3 1 4	SAFETY

Note

Problem I. The Secret

Input file:	`standard input`
Output file:	`standard output`
Time limit:	6 seconds
Memory limit:	64 megabytes

The random integer y is either zero or one, and the random integer x is between 1 and N. The value y is a secret. We are given an interval $[a, b] \subseteq [0, 1]$, and we need $\Pr[y = 1] \in [a, b]$ to hold at all times in order to guarantee the confidentiality of the secret. (I.e. we want that $a \leq \Pr[y = 1] \leq b$.)

The values $\Pr[x = i]$ and $\Pr[y = 1|x = i]$ are known for all i between 1 and N. Unfortunately, the dependency between x and y means that someone who learns x might be able to compute good bounds on $\Pr[y = 1]$. Fortunately, the value of x is not yet known. Your goal is to censor the value of x such that nobody can learn too much about the value of y.

To censor x means to partition the set $\{1, 2, \ldots, N\}$ into equivalence classes $A_1, \ldots, A_M$ such that, as soon as x is determined, only the index i of the unique equivalence class A_i that contains x becomes known (instead of the precise value of x).

You therefore have to find a partition $\bigcup_{i=1}^{M} A_i = \{1, 2, \ldots, N\}$ such that for all i between 1 and M, the confidentiality of the secret is maintained, i.e. $\Pr[y = 1|x \in A_i] \in [a, b]$.

(Hint: Recall that $\Pr[A|B] \cdot \Pr[B] = \Pr[A \wedge B]$, where $A \wedge B$ means that both A and B are true, and $\Pr[x \in A] = \sum_{i \in A} \Pr[x = i]$.)

In case there are multiple solutions, compute an arbitrary solution that maximizes the number of equivalence classes that contain only a single element.

Input

The first line of the input contains two space-separated integers A and B such that $a \cdot 10^6 = A$ and $b \cdot 10^6 = B$. The second line of the input contains the single integer N ($1 \leq N \leq 10^6$).

The i-th of the next N lines of the input contains integers X_i and Y_i such that $0 < \Pr[x = i] \cdot 10^6 = X_i$ and $\Pr[y = 1|x = i] \cdot 10^6 = Y_i$.

Output

If there is no solution, print the single integer -1.

Otherwise, the first line of the output should contain the size M of the partition. The i-th of the following M lines should contain an integer K_i, the number of elements of the i-th equivalence class A_i, followed by K_i numbers, the elements of A_i.

You may print the equivalence classes and their elements in any order.

Examples

standard input	standard output
450000 550000 6 100000 449999 100000 550001 100000 400000 100000 600000 300000 500000 300000 500000	4 2 1 2 2 3 4 1 5 1 6
500000 500000 5 200000 500000 200000 500000 200000 500000 200000 500000 200000 500000	5 1 1 1 2 1 3 1 4 1 5
228503 520839 1 1000000 379204	1 1 1

Problem J. Box Hedge

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	64 megabytes

As a reward of your great performance at SWERC in Paris, you're offered a job to take care of the garden in Versailles.

The palace complex of Versailles is arranged as follows: The main building is the center of everything. On one side, there is the garden and on the other side there is the village of Versailles. Those two are separated by a straight street of infinite length.

Now to the garden itself. Most of it are flowerbeds and you have full control over them. At certain points, however, there are static attractions (monuments, fountains, small huts, bird cages, etc.). You can't change the positions of those.

You want to plant a box hedge that surrounds all attractions and contains the minimum area possible (you are very smart and figured that less area means less work).

Let's define the task formally. The garden consists of a grid of 1×1 tiles with coordinates (x, y), such that x is any integer and y is any non-negative integer (below the tiles at $y = 0$ there is the street). Two tiles are adjacent if they share an edge. We say tile a is S-connected to tile b if they both lie in S and are either adjacent or a is connected to a tile in S adjacent to b. You need to find a set of tiles, called box hedge, such that the following conditions are fulfilled:

- The box hedge is connected, i.e. all tiles of the box hedge are pairwise box-hedge-connected.
- The box hedge touches the street, i.e. at least one tile of the box hedge has $y = 0$.
- All attractions are surrounded. We define outside tiles as tiles S-connected to a tile strictly above the highest tile of the box hedge, where S are all tiles not part of the box hedge. All tiles which are neither part of the box hedge nor outside are called surrounded.

The circumference of the box hedge is the number of edges separating outside tiles with the box hedge.

The size of the box hedge is the number of tiles contained in the box hedge.

The area is the number of surrounded tiles.

Find a box hedge satisfying the conditions listed above that minimizes the circumference as first priority, its size as second priority and the area as third priority.

Input

The first line contains an integer N, the number of attractions $1 <= N <= 10^6$.

Each of the following N lines contains two coordinates x and y ($-10^9 \leq x \leq 10^9$ and $0 \leq y \leq 10^9$).

It is guaranteed that no two attractions have the same coordinate.

Output

Print a single line containing three numbers: the minimum circumference, size and area.

Example

standard input	standard output
3 2 1 -1 2 2 4	18 16 14

Note

The optimal box hedge looks like this (where "B" means "box hedge", "A" means "attraction", "." means "inside, but not attraction", "S" means "street" and "V" means "village"):

```
      BBB
      BAB
   BBBB.B
   BA...B
   B...AB
   B....B
SSSSSSSSSSS
VVVVVVVVVVV
VVVVVVVVVVV
```

Problem K. ACM

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	64 megabytes

Programming contests are fun! So fun in fact, that you find yourself browsing online through hundreds of programming-related tasks, checking if they are about competitive programming. As quite a few of those tasks were not what you were looking for, you were wondering if you could write a program that classifies the problems for you!

You figured out that the following classification works every time: If the task description mentions "ACM", then the task is about competitive programming and will thus be very fun to solve. Otherwise, the task is not.

Input

You are given the task description, consisting of only uppercase letters of the Latin alphabet (`A` through `Z`). It is guaranteed that there will be at least one and at most 10^6 characters.

Output

Print "`Fun!`" if the statement contains `ACM`, and "`boring...`" otherwise.

Examples

standard input	standard output
SWISSSUBREGIONALACMICPC	Fun!
XXXAXCXMXXX	boring...