Problem A
Area Query

You are given a convex polygon with n vertices on the Cartesian plane. Its vertices are numbered from 1 to n in clockwise order. The i-th vertex has coordinates (x_i, y_i).

At the beginning, no diagonals of this polygon exist.

Your task is to process q queries of three following types:

- `A i j`: draw a new diagonal connecting the i-th vertex to the j-th vertex. It is guaranteed that the i-th vertex is not adjacent to the j-th vertex, the diagonal connecting these two vertices does not exist right before this query, and this diagonal does not intersect with any existing diagonals except at endpoints.
- `R i j`: erase the diagonal connecting the i-th vertex to the j-th vertex. It is guaranteed that this diagonal exists right before this query.
- `? i j`: The exisiting diagonals divide the given polygons into smaller pieces. You need to solve the following subproblem: choose to remove some (possibly none) of these pieces so that:
  - The piece containing both the i-th vertex and the $(i \mod n+1)$-th vertex is not removed.
  - The piece containing both the j-th vertex and the $(j \mod n+1)$-th vertex is not removed.
  - The two above pieces remain connected.
  - The total area of removed pieces is as large as possible.

Two pieces A and B are considered connected iff they are not removed, and any of the following conditions holds:

- A and B have a common edge.
- There exists a piece C such that C is not removed and C is considered connected to both A and B.

Please note that, during queries of the third type, no pieces of the polygon are actually removed. You just solve the subproblem and print the largest possible total area of removed pieces. The vertices and edges of the given polygon stay the same at all time.

Input

The input consists of multiple test cases. Each test case is presented as below:

- The first line contains a single integer n $(3 \le n \le 2 \times 10^5)$ – the number of vertices of the given polygon.
- In the next n lines, the i-th one contains two integers x_i and y_i $(0 \le |x_i|, |y_i| \le 10^9)$ – the coordinate of the i-th vertex.
- The next line contains a single integer q $(1 \le q \le 2 \times 10^5)$ – the number of queries.
- In the next q lines, each contains a query in any of the above formats.

The input is terminated by a line containing a single 0 which does not represent a test case.

It is guaranteed that:

- The sum of n over all test cases does not exceed 2×10^5.
- The sum of q over all test cases does not exceed 2×10^5.

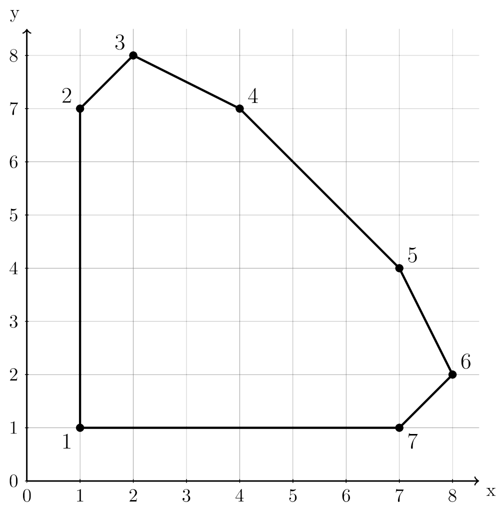
Output

For each query of the third type, print a single number – the largest possible total the area of removed pieces.

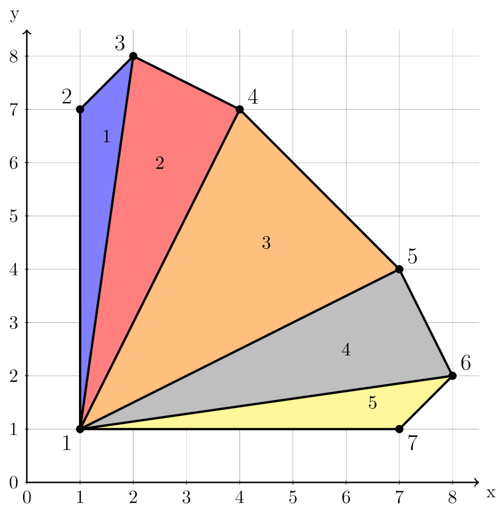
Your answer will be considered correct if its relative or absolute error doesn't exceed 10^{-6}.

Formally: let C be your answer, and J be the jury's answer. The checker program will consider your answer correct, if $\frac{|C-J|}{max(1,|J|)} \le 10^{-6}$.

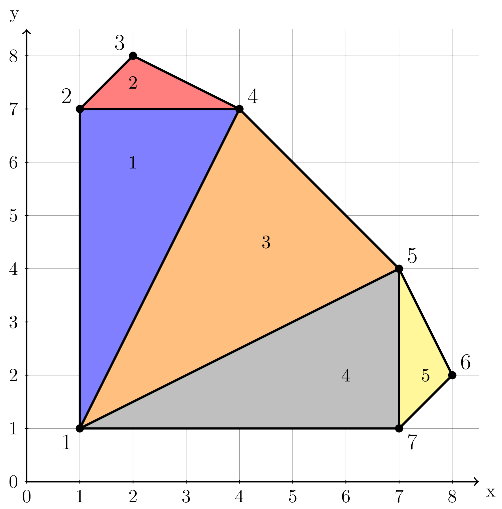
Sample Explanation



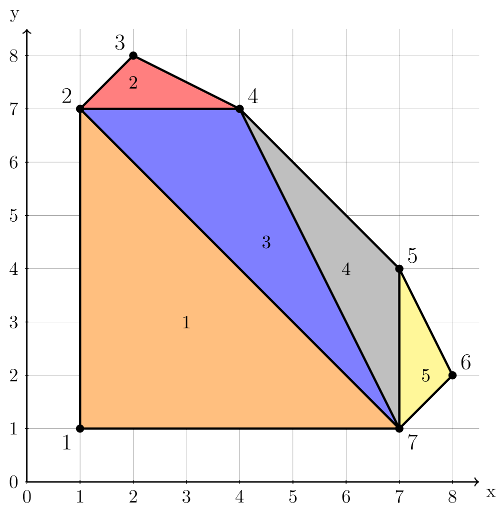
This is the given convex polygon. Initially, no diagonal exists.



This is the polygon with all exisiting diagionals right before the fifth query. The piece containing both the first and the second vertices is numbered 1. The piece containing both the third and the forth vertices is numbered 2. After removing all pieces numbered 3, 4 and 5, the two above pieces stay connected.



This is the polygon with all existing diagionals right before the tenth query. The piece containing both the second and the third vertices is numbered 2. The piece containing both the fifth and the sixth vertices is numbered 5. In order to keep the two above pieces connected, we can not remove any pieces.



This is the polygon with all existing diagionals right before the fifteenth query. The piece containing both the seventh and the first vertices is numbered 1. The piece containing both the forth and the fifth vertices is numbered 4. After removing pieces numbered 2 and 5, the two above pieces stay connected.

Sample Input 1

```
7
1 1
1 7
2 8
4 7
7 4
8 2
7 1
15
A 1 3
A 1 4
A 1 5
A 1 6
? 1 3
R 1 3
A 2 4
R 1 6
A 5 7
? 2 5
R 1 5
R 1 4
A 4 7
A 2 7
? 7 4
0
```

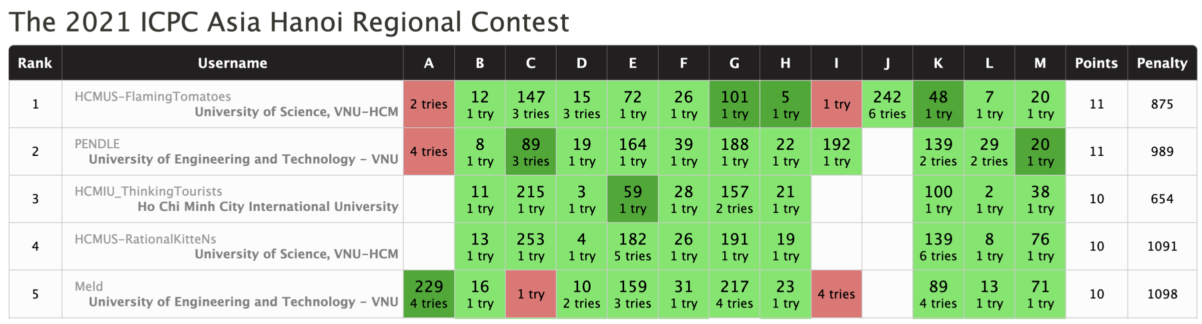
Sample Output 1

```
24.0
0.0
3.0
```

Problem B
Beautiful Scoreboard

In The 2021 ICPC Asia Hanoi Regional Contest, the final scoreboard was deemed beautiful:

- for each problem, at least one team managed to solve it,
- no team was able to solve all the problems,
- each team solved at least one problem.



The 2021 ICPC Asia Hanoi Regional Contest

Rank	Username	A	B	C	D	E	F	G	H	I	J	K	L	M	Points	Penalty
1	HCMUS–FlamingTomatoes **University of Science, VNU–HCM**	2 tries	12 1 try	147 3 tries	15 3 tries	72 1 try	26 1 try	101 1 try	5 1 try	1 try	242 6 tries	48 1 try	7 1 try	20 1 try	11	875
2	PENDLE **University of Engineering and Technology – VNU**	4 tries	8 1 try	89 3 tries	19 1 try	164 1 try	39 1 try	188 1 try	22 1 try	192 1 try		139 2 tries	29 2 tries	20 1 try	11	989
3	HCMIU_ThinkingTourists **Ho Chi Minh City International University**		11 1 try	215 1 try	3 1 try	59 1 try	28 1 try	157 2 tries	21 1 try			100 1 try	2 1 try	38 1 try	10	654
4	HCMUS–RationalKitteNs **University of Science, VNU–HCM**		13 1 try	253 1 try	4 1 try	182 5 tries	26 1 try	191 1 try	19 1 try			139 6 tries	8 1 try	76 1 try	10	1091
5	Meld **University of Engineering and Technology – VNU**	229 4 tries	16 1 try	1 try	10 2 tries	159 3 tries	31 1 try	217 4 tries	23 1 try	4 tries		89 4 tries	13 1 try	71 1 try	10	1098

Figure B.1: Top 5 of The 2021 ICPC Asia Hanoi Regional

Vietnamese ICPC judges strive to design a problemset that results in a beautiful final scoreboard. Your task is to determine whether a given final scoreboard is beautiful.

Input

The first line of the input contains a single integer t $(1 \le t \le 100)$ – the number of test cases. t test cases follow, each is presented as below:

- The first line contains two integers n and m $(1 \le n \le 100, 1 \le m \le 20)$ – the number of teams participated in the competition and the number of problems, respectively.
- In the next n lines, each contains exactly m integers representing one row in the final scoreboard. A 0 indicates that the team did not solve the problem, and a 1 indicates that the team successfully solved the problem.

It is guaranteed that in the scoreboard, if team A appears before team B, then team A must solve at least as many problems as team B.

Output

For each test case, print `YES` if the scoreboard is beautiful according to the given criteria. Otherwise, print `NO`.

Sample Explanation

The sample input contains top 5 of The 2021 ICPC Asia Hanoi Regional, shown in figure 1.

Sample Input 1

```
1
5 13
0 1 1 1 1 1 1 1 0 1 1 1 1
0 1 1 1 1 1 1 1 1 0 1 1 1
0 1 1 1 1 1 1 1 0 0 1 1 1
0 1 1 1 1 1 1 1 0 0 1 1 1
1 1 0 1 1 1 1 1 0 0 1 1 1
```

Sample Output 1

```
YES
```

Problem C
Crop Circle Conundrum

Keke, a mischievous alien, has landed on Earth with a unique mission: to create an artistic crop circle on a farm! He visualizes the farm as an infinite hexagonal grid and wishes to paint certain cells to form his favorite pattern. More precisely, Keke uses a hexagonal coordinate system to divide the farm into hexagonal cells, with the farm's center as cell $(0, 0)$. See figure 1 for the coordinate system. Keke aims to paint a list of cells, with the i-th cell having coordinates (x_i, y_i).

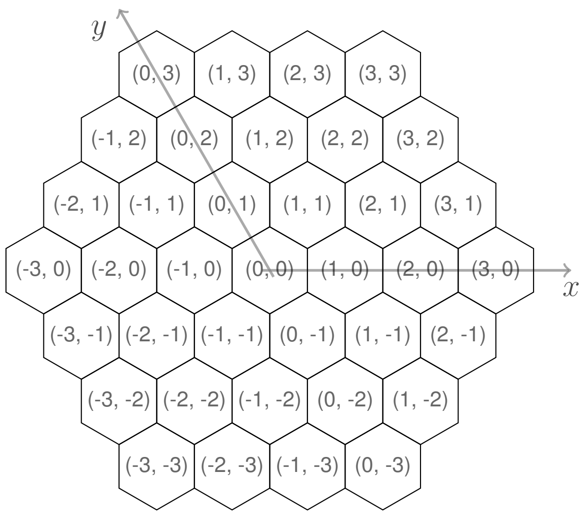


Figure C.1: Hexagonal coordinate system

Keke's spaceship is equipped with a specialized beam-gun capable of painting a star pattern on the farm. The size of a star pattern can vary between 2 and 666 (inclusive). See figure 2 for illustration.

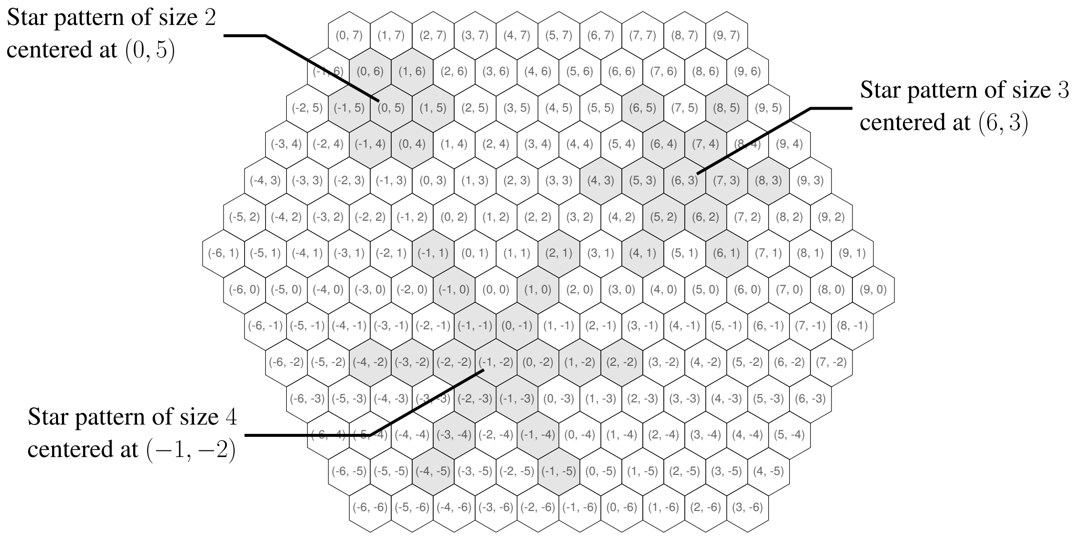

Star pattern of size 2 centered at $(0, 5)$
Star pattern of size 3 centered at $(6, 3)$
Star pattern of size 4 centered at $(-1, -2)$

Figure C.2: Star patterns of size 2, 3 and 4

The beam-gun has a quirky property: if it hits a hexagonal cell twice, the cell reverts to its original state, effectively erasing the paint. Keke can fire the gun-beam up to 666 times. Keke has spent a few months trying to paint his favorite pattern, without success. Please help him!

Input

The first line of the input contains a single integer t $(1 \le t \le 6666)$ – the number of test cases. t test cases follow, each is presented as below:

- The first line contains a single integer n $(0 \le n \le 66)$ – the number of cells in Keke's favorite pattern.
- In the next n lines, each contains two integers x_i and y_i $(0 \le |x_i|, |y_i| \le 6)$ – the coordinate of the i-th cell in Keke's favorite pattern. It is guaranteed that all n cells are distinct.

It is guaranteed that the sum of n over all test cases does not exceed 6666.

Output

For each test case, if there is no solution, print `NO`. Otherwise, print `YES`, followed by:

- a line containing a single integer k $(0 \le k \le 666)$ – the number of times you want to use Keke's beam-gun;
- k lines, the j-th one with three integers: x_j, y_j and r_j $(0 \le |x_j|, |y_j| \le 666, 2 \le r_j \le 666)$, where (x_j, y_j) is the center where the gun-beam should target, and r_j is the size of the star.

If there are multiple correct solutions, you can output any of them.

Sample Explanation

The following figure illustrates Keke's favorite pattern, with painted cells having a cross:

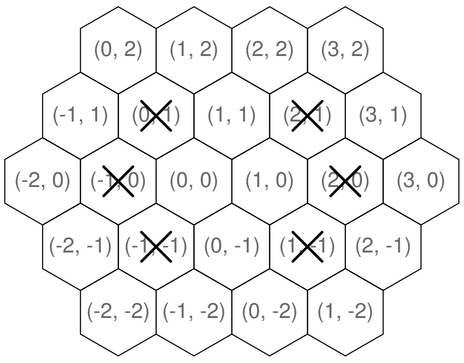
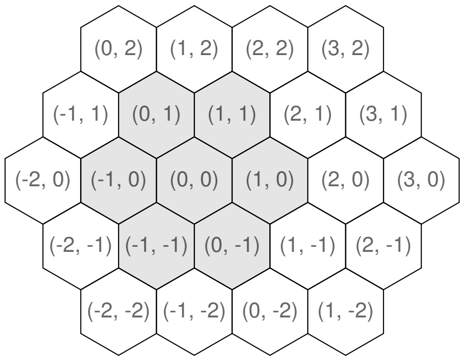
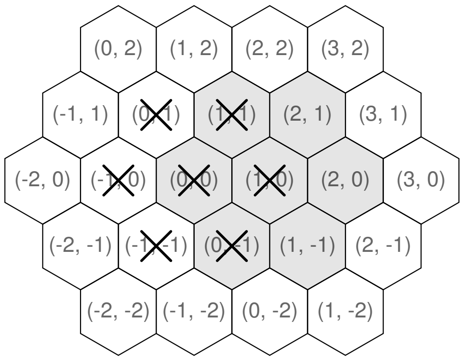


Figure C.3: Keke's favorite pattern

Keke can perform the following operations with his beam-gun:



Step 1. Aim the beam-gun at cell $(0, 0)$ and shoot with star size of 2.



Step 2. Aim the beam-gun at cell $(1, 0)$ and shoot with star size of 2.

Sample Input 1

```
1
6
0 1
-1 0
-1 -1
2 1
2 0
1 -1
```

Sample Output 1

```
YES
2
0 0 2
1 0 2
```

This page is intentionally left blank.

Problem D
Divisive Stone Battle

Bash and Chikapu, as part of their training for the national Pokenom championship, often play strategic games to sharpen their wits.

Today, they gathered some stones in the nearby forest and now are engaging in the following game:

- Initially, they arrange the stones into n piles. The i-th pile contains a_i stones. They also choose an integer k.
- They alternatively take turns. Bash plays first.
- In each turn, a player picks a non-empty pile of stones. Suppose the player chooses the i-th pile. Let c_i be the current number of stones of this pile. Then the player removes some stones from this pile, so that the number of remaining stones is either $c_i - 1$ or $\lfloor \frac{c_i}{k} \rfloor$.
- The player unable to make a move on their turn loses the game.

Both Bash and Chikapu play optimally given their exceptional intelligence. Your task is to determine the winner of the game.

As a reminder, $\lfloor x \rfloor$ is the largest integer which is not greater than x.

Input

The first line of the input contains a single integer t $(1 \leq t \leq 10^5)$, the number of test cases. t test cases follow, each is presented as below:

- The first line contains two integers n and k $(1 \leq n \leq 10^5, 2 \leq k \leq 100)$ denoting the number of piles and the chosen integer, respectively.
- The second line contains n integers $a_1, a_2, \ldots, a_n$ $(1 \leq a_i \leq 10^{18})$ – the number of stones in each pile.

It is guaranteed that the sum of n over all test cases does not exceed 10^5.

Output

For each test case, print on a single line the name of the winner, either `Bash` or `Chikapu`.

Sample Input 1

```
2
1 2
1
4 2
7 7 6 6
```

Sample Output 1

```
Bash
Chikapu
```

This page is intentionally left blank.

Problem E
Eclipse

In today's science class, the whole class was taught about solar eclipses. Since this is a process that rarely occurs, and is not simple or spacious, to illustrate for the students, the teacher wrote a computer program to simulate this phenomenon.

The teacher's program uses computer graphics, displaying a plane on the Oxy coordinate system. On the screen, the teacher drew two **convex** polygons, one represents the sun and one represents the moon. To illustrate the process of a solar eclipse, the teacher rotates the moon polygon around the origin, and when the moon is completely within the sun, the teacher says this is the phenomenon of a solar eclipse!

All the students were very interested in the teacher's illustration and took turns operating the teacher's program. The teacher's program has many advanced features, allowing students to modify the two polygons as they wish as long as they remain convex polygons. However, not every pair of moon and sun polygons can create a solar eclipse phenomenon.

Given a pair of **convex** polygons representing the moon and the sun from the students, your task is to check whether there exists an angel to rotate the moon polygon around the origin so that the solar eclipse phenomenon occurs.

Define $P(X)$ the set of points which are inside the polygon X and on the edges of polygon X.

Polygon A is said to be completely inside polygon B if $P(A) \subset P(B)$.

When rotating polygon A around the origin with angle α, all points belonging to $P(A)$ will be rotated around the origin with angle α.

Input

The first line of the input contains a single integer t $(1 \le t \le 10^4)$ – the number of test cases. t test cases follow, each is presented as below:

- The first line contains an integer n $(3 \le n \le 1000)$ – the number of vertices of the polygon representing the sun.
- In the next n lines, the i-th one contains two integers x_i and y_i $(0 \le |x_i|, |y_i| \le 10^6)$ – the coordinates of the i-th point of the polygon representing the sun.
- The next line contains an integer m $(3 \le m \le 1000)$ – the number of vertices of the polygon representing the moon.
- In the last m lines, the j-th one contains two integers The j-th line of the following m lines contains a pair of integers x_j and y_j $(0 \le |x_j|, |y_j| \le 10^6)$ — the coordinates of the j-th point of the polygon representing the moon.

It is guaranteed that:

- All polygons are convex.
- Points of each polygon are listed in the counterclockwise order.

- No three consecutive points of each polygon are collinear.
- If there exists an angle α so that the solar eclipse phenomenon occurs, there will also exist an angle β so that **all** angels $\gamma \in [\beta; \beta + 10^{-6}]$ have this property. Angels are measured in radian.
- The sum of $n \times m$ over all test cases does not exceed 10^6.

Output

For each test case, print “YES” (without quotes) if there exists a rotation angle for the moon polygon such that the solar eclipse phenomenon occurs, otherwise print “NO” (without quotes).

Sample Explanation

In this section, figures 1, 2 and 3 illustrate the first, second and third test cases, respectively.

In the first test case, the triangle ABC represents the sun, and the **shaded** quadrilateral DEFG represents the moon. The quadrilateral with **dotted** edges, D′E′F′G′, is obtained by rotating the quadrilateral DEFG around the origin O, and it completely lies inside the triangle ABC. Thus, the answer for this test case is YES.

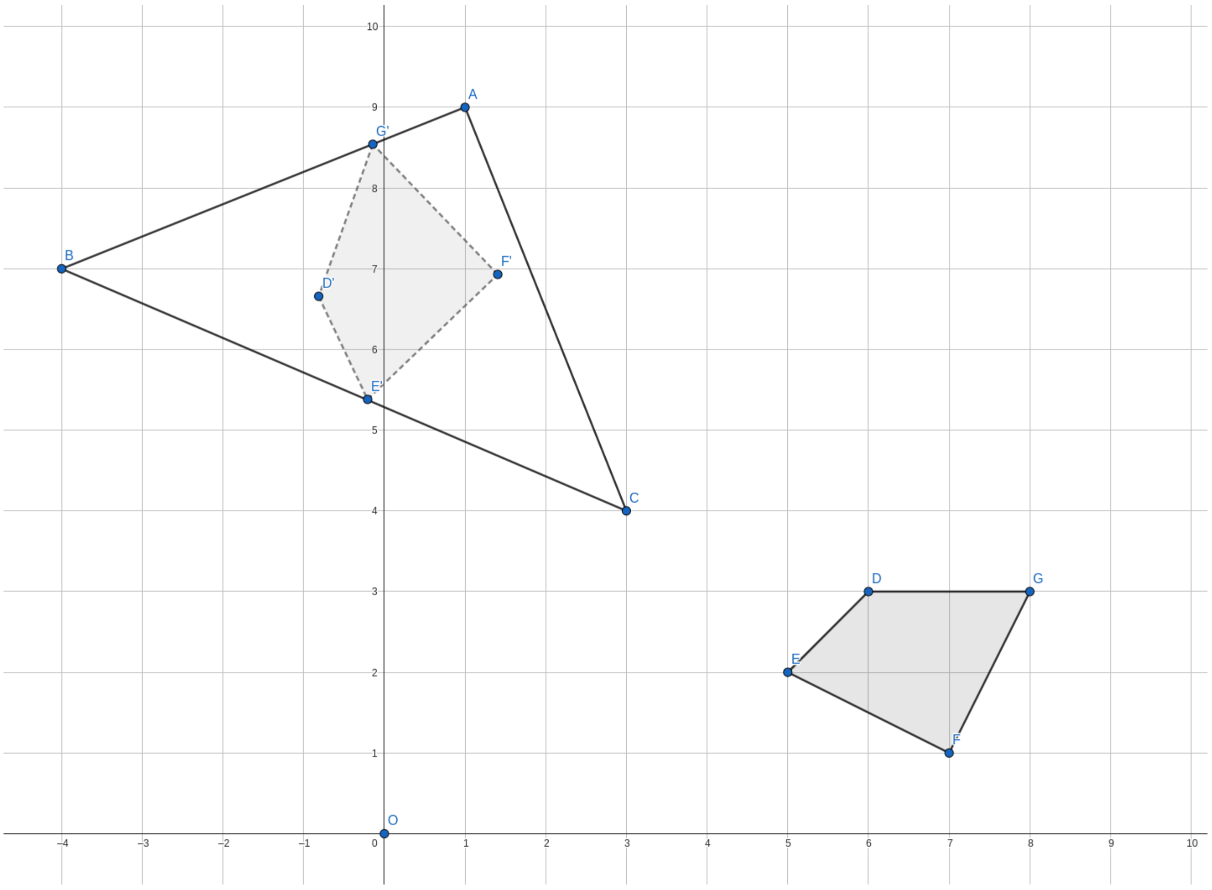


Figure E.1: Illustration for the first test case

In the second test case, the quadrilateral `ABCD` represents the sun, and the **shaded** triangle `EFG` represents the moon. It is impossible to rotate the triangle `EFG` around the origin so that it completely lies inside the quadrilateral `ABCD`. Thus, the answer is `NO`.

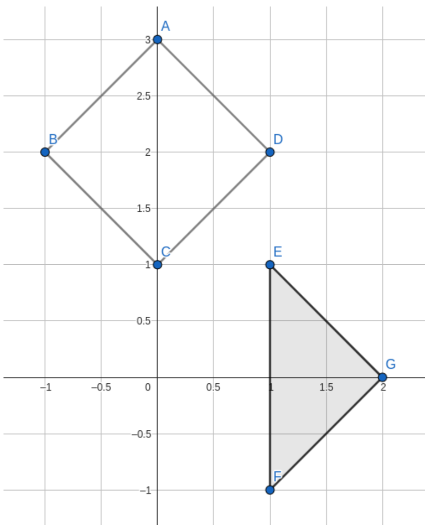


Figure E.2: Illustration for the second test case

In the third test case, the quadrilateral `ABCD` represents the sun, and the **shaded** triangle `IJK` represents the moon. The triangle with **dotted** edges, `I′J′K′`, is the image of the triangle `IJK`, under the rotation of angle $\frac{\pi}{4}$ about the origin; which completely lies inside the quadrilateral `ABCD`. Please note that there are other angles that cause the eclipse phenomenon, including $\frac{3\cdot\pi}{4}$, $\frac{5\cdot\pi}{4}$ and $\frac{7\cdot\pi}{4}$.

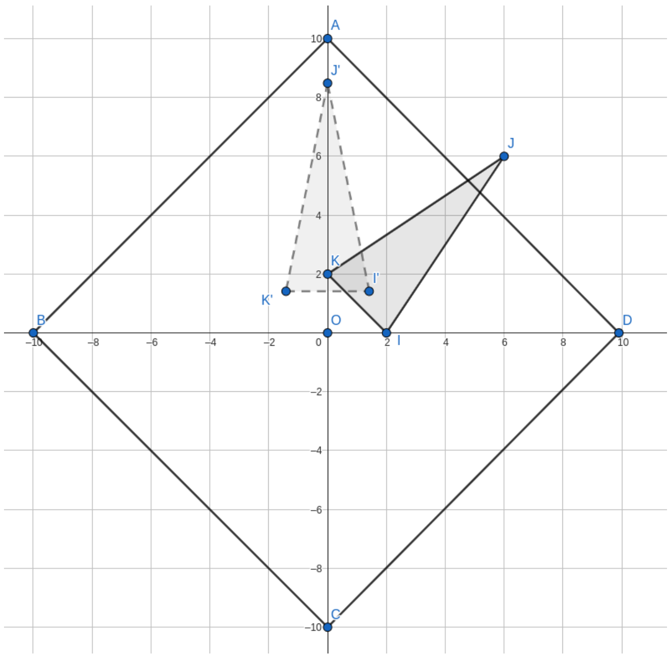


Figure E.3: Illustration for the third test case

Sample Input 1	Sample Output 1

```
3
3
1 9
-4 7
3 4
4
6 3
5 2
7 1
8 3
4
0 3
-1 2
0 1
1 2
3
1 1
1 -1
2 0
4
0 10
-10 0
0 -10
10 0
3
2 0
6 6
0 2
```
	```
	YES
	NO
	YES
	```

Problem F
Flipping Substrings

Given two strings a and b. Your task is to determine whether it is possible to transform string a into string b, using only the following operation multiple (possibly zero) times: Select a substring of a with **odd length** and reverse it.

Recall that a substring is a contiguous sequence of characters within a string.

Input

The first line of the input contains a single integer t – the number of test cases. t test cases follow, each consists of 2 lines: the first line contains string a, and the second line contains string b.

It is guaranteed that the total length of all strings in the input does not exceed 10^5, and all given strings contain lowercase English letters only.

Output

For each test case, print a single line containing `YES` if it is possible to transform string a into string b using the given operation only, and `NO` otherwise.

Sample Explanation

In the first test case, you can transform string a into string b using the following sequence of operations (note that it is not the one with least number of steps):

- Reverse substring `han`. a becomes `nahoi`.
- Reverse the whole string. a becomes `iohan`.
- Reverse substring `han`. a becomes `ionah`.

Sample Input 1

```
3
hanoi
ionah
ac
wa
hue
hue
```

Sample Output 1

```
YES
NO
YES
```

Problem G
Glamorous Garment Gameshow

It's official! The VNOI (Vietnamese National Outfit Innovation) organization has been selected to host the 2023 Glamorous Garment Gameshow. Promising to be a dazzling event, it will be broadcasted live to millions of viewers in Vietnam. The competition features $2 \times n$ talented contestants, evaluated by esteemed Vietnamese celebrity designers in a high-stakes, multi-stage format. The i-th contestant has a calculated **strength index** s_i (known as *chỉ số sức mạnh* in Vietnamese), determined by factors such as social media popularity, fan base, and even mouse movement speed, among others.

The preliminary stage consists of two rounds. In each round, contestants are divided into n pairs, engaging in intense one-on-one design battles. To maintain excitement, contestants who face off in the first round won't meet again in the second, ensuring an engaging spectacle. Mathematically, let a_i and b_i be the opponents of the i-th contestant in the first and second rounds, respectively. The following conditions must be met for all $1 \le i \le 2 \times n$:

- $a_i \neq i$;
- $b_i \neq i$;
- $a_i \neq b_i$;
- $i = a_{a_i} = b_{b_i}$.

The organizers aim to create compelling matchups by pairing contestants with minimal differences in their strength indices. In other words, they seek to minimize

$$\max_{i=1\ldots 2n} \{\max(|s_i - s_{a_i}|, |s_i - s_{b_i}|)\}$$

Given the strength indices of all contestants, your task is to organize the contestants into pairs for the two rounds of the preliminary stage, aiming to minimize the differences in strength indices among all pairs.

Input

The input consists of multiple test cases. Each test case is presented in two lines:

- The first line contains a single integer n ($2 \le n \le 2 \times 10^5$).
- The second line contains $2 \times n$ integers $s_1, s_2, \ldots, s_{2\times n}$ ($1 \le s_i \le 10^9$) – the strength indices of the contestants. Note that the contestants are numbered from 1 to $2 \times n$.

The input is terminated by a line containing a single 0 which does not represent a test case.

It is guaranteed that the sum of n over all test cases does not exceed 2×10^5.

Output

For each test case, output two lines:

- The first line contains $2 \times n$ integers $a_1, a_2, \ldots, a_{2\times n}$ $(1 \leq a_i \leq 2 \times n)$ – the indices of the contestants' opponents in the first round.
- The second line contains $2 \times n$ integers $b_1, b_2, \ldots, b_{2\times n}$ $(1 \leq b_i \leq 2 \times n)$ – the indices of the contestants' opponents in the second round.

If there are multiple optimal solutions, you can output any of them.

Sample Explanation

The given strength indices of $2 \times n = 4$ contestants are $s = [1, 2, 3, 4]$

In the first round:

- Contestants 1 and 2 are paired, exhibiting a strength difference of $|s_1 - s_2| = |1 - 2| = 1$.
- Contestants 3 and 4 are paired, displaying a strength difference of $|s_3 - s_4| = |3 - 4| = 1$.

In the second round:

- Contestants 1 and 3 are paired, showcasing a strength difference of $|s_1 - s_3| = |1 - 3| = 2$.
- Contestants 2 and 4 are paired, demonstrating a strength difference of $|s_2 - s_4| = |2 - 4| = 2$.

This arrangement yields a maximum difference of $\max(1, 1, 2, 2) = 2$.

It's proven to be the arrangement that minimizes the differences among all possible configurations.

Sample Input 1

```
2
1 2 3 4
0
```

Sample Output 1

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

Problem H
Harmonious Hue Palace

In the historic city of Hue, Vietnam, the revered king wants to construct a grand palace. The city is laid out in an $n \times n$ grid. The rows are numbered from 1 to n from north to south, and the columns are numbered from 1 to n from east to west. The cell on the i-th row and the j-th column of the grid is denoted as (i, j). Each cell radiates with either harmonious or ordinary energy.

The palace must be a rectangle within this grid, with its sides parallel to the city's borders. Also, to ensure fengshui alignment, among its 4 corners, exactly three must be harmonious and the other must be ordinary. Your task is to find a location to build the palace. The king is not patient, so you have to be quick!

Input

The first line of the input contains a single integer t $(1 \le t \le 10^5)$ – the number of test cases. t test cases follow, each is presented as below:

- The first line contains an integer n $(2 \le n \le 5000)$ – the size of the grid.
- The second line contains a string s – the **Base64-compressed** representation the grid, obtained by the following steps:
  - First, the grid is represented as matrix a of size $n \times n$, where $a_{i,j}$ equals to 0 if cell (i, j) is harmonious or 1 if this cell is ordinary.
  - The elements of a are then listed row-first to obtain zero-indexed binary string b of length $n \times n$. Formally, $b_{(i-1)\times n+(j-1)} = a_{i,j}$ for every $1 \le i, j \le n$.
  - b is then divided into continuous binary substrings of length 6. If the last substring has less than 6 characters, repeat adding `0` to the end of this string until its length equals 6. Let c_i be the i-th obtained substring.
  - For each string c_i obtained during the previous steps, denote $c_i = c_{i,0}c_{i,1}c_{i,2}c_{i,3}c_{i,4}c_{i,5}$. Then we calculate the value $d_i = \sum_{j=0}^{5} 2^j \times c_{i,j}$. It can be seen that $0 \le d_i \le 63$.
  - Finally, the string s is constructed as below: The i-th character of the string s equals to the d_i-th character of the following string (which has 64 characters, numbered from 0 to 63):

```
ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/
```

See the sample explanation for better understanding.

It is guaranteed that the sum of n^2 over all test cases does not exceed 2.5×10^7.

Output

For each test case, if no appropriate placement for the palace exists, output `NO` on a single line.

Otherwise, output `YES` on the first line. On the second line, output four integers x_1, y_1, x_2 and y_2 ($1 \le x_1, y_1, x_2, y_2 \le n$, $x_1 \ne x_2$, $y_1 \ne y_2$), where (x_1, y_1) and (x_2, y_2) denote two opposite corners of the palace.

If there are multiple correct solutions, you can output any of them.

Sample Explanation

For the first testcase, the given matrix a is:

$$\begin{matrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{matrix}$$

The Base64-compressed representation of the grid is obtained by the following steps:

- $b = $ `001010101`
- $c_0 = $ `001010`, $c_1 = $ `101000`
- $d_0 = 2^2 + 2^4 = 20$, $d_1 = 2^0 + 2^2 = 5$
- $s_0 = $ `U`, $s_1 = $ `F`

In this testcase, another valid placement of the palace is to place the top-left corner at cell $(1, 1)$ and the bottom-right corner at $(2, 3)$.

For the second testcase, the given matrix a is:

$$\begin{matrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 1 & 1 \end{matrix}$$

For all placements of the palace, there are always at least two ordinary cells, violating the fengshui alignment.

For the third testcase, the given matrix a is:

$$\begin{matrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 \end{matrix}$$

Sample Input 1

```
3
3
UF
3
VH
4
/zO
```

Sample Output 1

```
YES
1 3 2 1
NO
YES
4 2 3 1
```

Problem I
It's Time To D-D-D-D-Duel!

The names of the characters have been redacted due to copyright concerns.

Ka•ba S•to, CEO of *Ka•baCorp* and Japan's number one Duelist is once again preparing to defeat his arch-rival, *M•to Y•gi*. Having tasted defeats after defeats mostly thanks to Y•gi's massive plot armor, Ka•ba finally decided to drop his noble façade and try underhanded tactics himself.

He demanded that both players duel using the newly developed *Duel Disk++*, which allows for automatic deck shuffling. However, as the inventor of *Duel Disk++*, Ka•ba knows exactly how the shuffling mechanisms work. Assuming the player's deck contains n cards. After one shuffle, the device will move the card currently at the i-th position to the p_i-th position (positions are numbered from 1 to n from the top of the deck), where p is a permutation of integers from 1 to n that is unique to the device. From prior experience, Ka•ba knows that Y•gi always sets up his deck in a particular order, and that Y•gi will always shuffle the deck using the device exactly k times at the start of the duel. In order to win, Ka•ba needs the i-th card in Y•gi's initial setup to end up at the a_i-th position after k shuffles. To ensure that, Ka•ba will secretly change the permutation p of Y•gi's *Duel Disk++* before the duel.

Given n integers $a_1, a_2, \ldots, a_n$ and the number of shuffles k, help Ka•ba find the number of permutations p that achieves the task.

Input

The first line contains two integers n and k ($1 \le n \le 10^5$, $1 \le k < 10^{100}$) — the number of cards in Y•gi's deck and the number of times Y•gi will shuffle his starting deck, respectively.

The second line contains n distinct integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le n$) – the desired configuration of Y•gi's deck.

Output

Output a single integer – the number of permutations p that Ka•ba can choose to fulfill his objective, modulo $998\,244\,353$.

Sample Explanation

In the first sample, Ka•ba can choose p to be either $[4, 1, 5, 2, 3]$ or $[4, 1, 3, 2, 5]$.

In the second and third samples, it can be proven that no matter the choice of p, the deck will always return to its original state after 120 shuffles. Therefore, there are $6! = 720$ permutations that achieve the desired deck configuration in sample 2, and 0 such permutation in sample 3.

Sample Input 1

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

Sample Output 1

```
2
```

Sample Input 2

```
6 120
1 2 3 4 5 6
```

Sample Output 2

```
720
```

Sample Input 3

```
6 120
1 2 3 4 6 5
```

Sample Output 3

```
0
```

Sample Input 4

```
10 15
4 8 5 1 3 10 9 2 7 6
```

Sample Output 4

```
465
```

Problem J
Jom and Terry

Recently, the mouse Terry has come to live in a cheese factory. Day after day, Terry has eaten a considerable amount of cheese that the factory has produced. With Terry's presence, the cheese supply has decreased significantly. Jom the cat has been tasked with guarding the cheese stock, preventing Terry from stealing any pieces of cheese. The first thing Jom did was to set traps at all positions where there was cheese. This caused a considerable hindrance to Terry's food-gathering process.

Not giving up, Terry devised a plan to safely eat the cheese inside the stockroom. After investigating the traps, Terry found there are n traps in the stockroom. Each trap has a special structure represented by a tube. At each end of the trap, Jom has placed one of two items:

- A piece of cheese with a positive mass.
- A special food glue that allows sticking together two pieces of cheese with masses x and y units, forming a new piece of cheese with a mass of $x + y$. Each glue can only be used once.

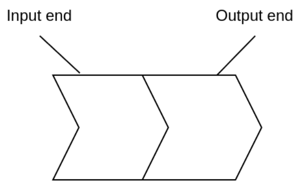


Figure J.1: Mouse trap

For each tube, Terry calls one of its end "the input end" and the other "the output end". Terry can cleverly move from "the input end" to "the output end" safely. However, moving in the opposite direction will trigger Jom's trap.

The following illustration is an example of traps placement by Jom the cat, with $n = 3$ traps:

- Trap 1 with a piece of cheese with mass 1 at the input end, and a glue at the output end.
- Trap 2 with a glue at the input end, and a piece of cheese with mass 2 at the output end.
- Trap 3 with a piece of cheese with mass 4 at the input end, and a piece of cheese with mass 3 at the output end.

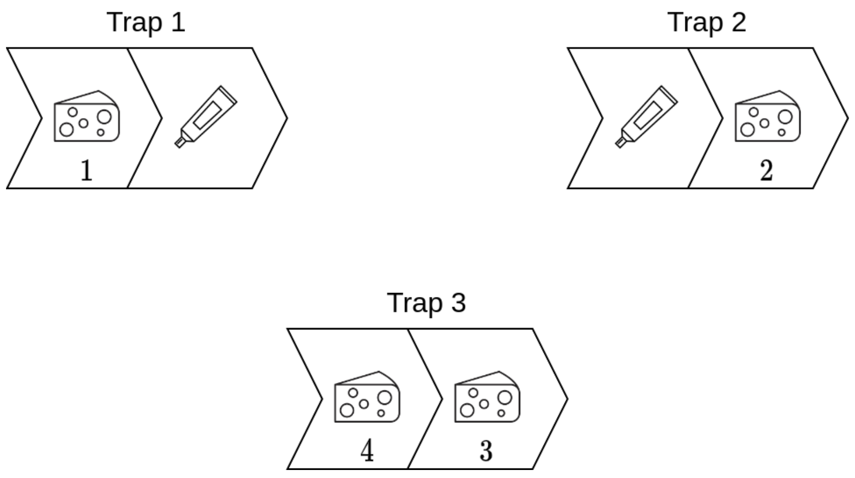


Figure J.2: An example of traps placement by Jom the cat.

With this information, Terry now has a plan. Terry has prepared a bag large enough to carry all the cheese in the stockroom. First, Terry will move to trap p_1, then to trap p_2, and so on, until trap p_n, where $p_1, p_2, \ldots, p_n$ is a permutation of integers from 1 to n that Terry will choose. When moving to trap i, Terry crawls into the trap, first picking up the item at the input end of the trap, moving through the trap, and then picking up the item at the output end.

- When Terry picks up a piece of cheese, Terry will put the cheese into the bag.
- When Terry picks up the food glue, Terry will take the **last** 2 pieces of cheese placed into the bag in the order they were added, stick them together using the glue, and put the newly combined cheese back into the bag. If there are fewer than two pieces of cheese remaining in the bag, Terry will throw away the glue.

The plan is nearly perfect, but Terry also needs an escape plan if Jom appears. Terry calculates that the time to move through all n traps is very fast. However, when returning to the burrow, due to carrying a very heavy bag, the travel time will be much longer. If Jom appears at that time, instinctively, Terry will take the **last** piece of cheese put into the bag, leave the bag with the remaining cheese, and quickly run back to the burrow. Of course, if there is no cheese in the bag, Terry will not take any cheese back to the burrow.

Given the description of items placed in n traps, help Terry find the sequence of movements $p_1, p_2, \ldots, p_n$ so that in case Terry has to escape from Jom, the mass of cheese Terry can take back to the burrow is as **large as possible**.

In the aforementioned example of trap placement, one possible plan to traverse the traps is $p = [3, 1, 2]$.

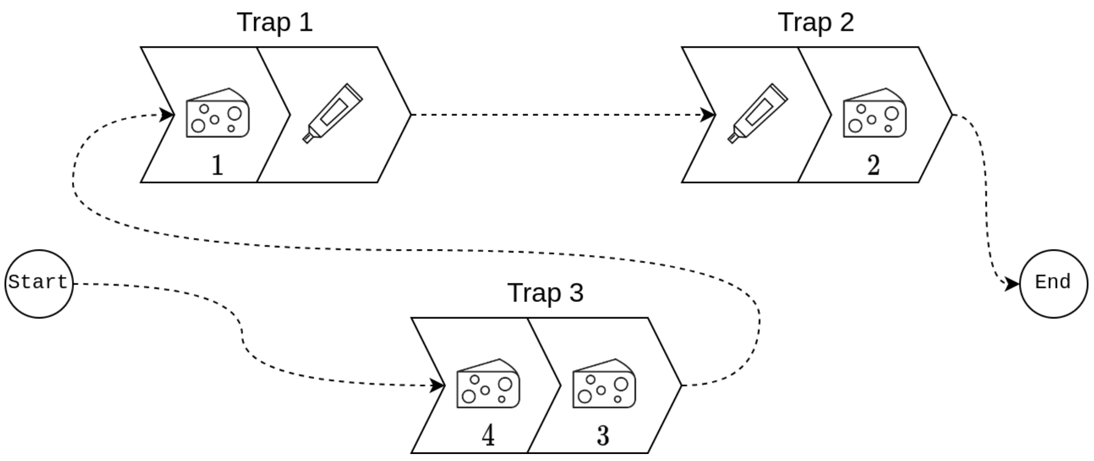
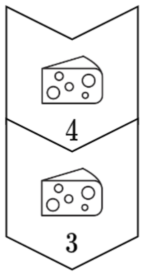
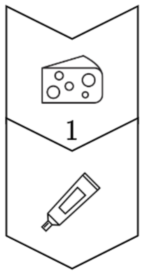
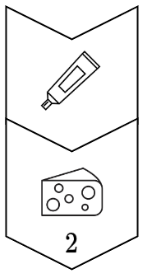


Figure J.3: A potential plan with the trap traversal order of $p = [3, 1, 2]$.

Following this plan and visiting all the traps, Terry's bag will contain pieces of cheese with masses of $[8, 2]$. The events and the corresponding state of Terry's cheese bag are explained as follows:

	Events	**Terry's cheese bag**
Enter Trap 3		
	Add a piece of cheese with mass of 4 to the bag.	$[4]$
	Add a piece of cheese with mass of 3 to the bag.	$[4, 3]$
Enter Trap 1		
	Add a piece of cheese with mass of 1 to the bag.	$[4, 3, 1]$
	Use the glue to attach the last two pieces of cheese. Obtain a new piece of cheese with a mass of $3 + 1 = 4$.	$[4, 4]$
Enter Trap 2		
	Use the glue to attach the last two pieces of cheese. Obtain a new piece of cheese with a mass of $4 + 4 = 8$.	$[8]$
	Add a piece of cheese with mass of 2 to the bag.	$[8, 2]$

However, if Jom appears, Terry will escape with only a piece of cheese with a mass of 2. A more advantageous plan would be $p = [2, 3, 1]$.

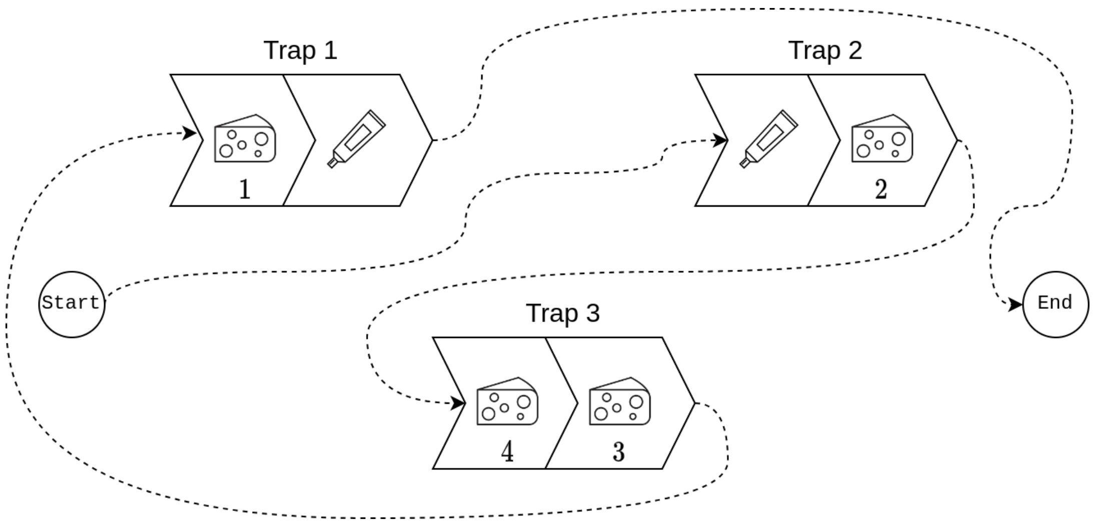
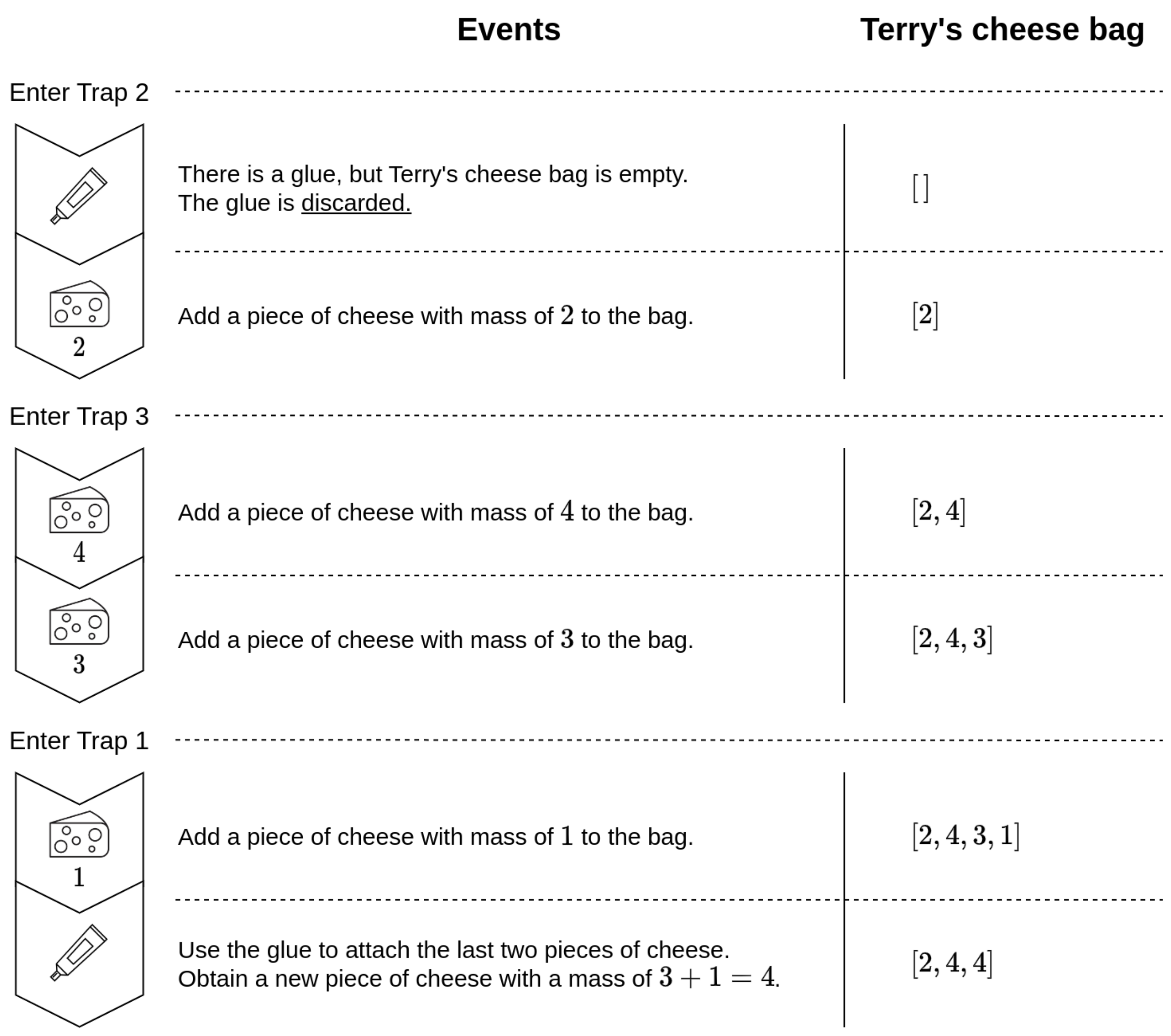


Figure J.4: An optimal plan with $p = [2, 3, 1]$.

Executing this plan would fill Terry's cheese bag with pieces of masses $[2, 4, 4]$ at the end. Thus, when Jom appears, Terry can escape with a piece weighing 4.



	Events	Terry's cheese bag
Enter Trap 2		
	There is a glue, but Terry's cheese bag is empty. The glue is <u>discarded.</u>	$[]$
2	Add a piece of cheese with mass of 2 to the bag.	$[2]$
Enter Trap 3		
4	Add a piece of cheese with mass of 4 to the bag.	$[2, 4]$
3	Add a piece of cheese with mass of 3 to the bag.	$[2, 4, 3]$
Enter Trap 1		
1	Add a piece of cheese with mass of 1 to the bag.	$[2, 4, 3, 1]$
	Use the glue to attach the last two pieces of cheese. Obtain a new piece of cheese with a mass of $3 + 1 = 4$.	$[2, 4, 4]$

It can be proven that this is the optimal plan for the example.

Input

The input consists of multiple test cases. Each test case is presented as below:

- The first line contains a positive integer n $(1 \leq n \leq 2 \times 10^5)$ – the number of traps that Jom has placed in the stockroom.

  In the next n lines, the i-th one contains two integers a_i and b_i $(0 \leq a_i, b_i \leq 10^9)$ describing the two ends of the i-th trap as below:

  – If $a_i > 0$, the input end of the trap contains a piece of cheese with a positive mass of a_i.
  – If $a_i = 0$, the input end of the trap contains food glue.

The number b_i describes the output end of this trap in the same way.

The input is terminated by a line containing a single 0 which does not represent a test case.

It is guaranteed that the sum of n over all test cases does not exceed 2×10^5.

Output

For each test case, print n integers $p_1, p_2, \ldots, p_n$ representing the order of trap movements in Terry's plan so that if Terry has to flee from Jom after visiting all n traps, the mass of cheese Terry can take back to the burrow is **largest possible**.

If there are multiple optimal solutions, you can output any of them.

Sample Explanation

In the first test case, there are no food glues in the stock room. Hence, the last piece of cheese put into the bag is the one at the output end of the last visited trap. Therefore, by visiting the traps in the order $p = [2, 1]$, the last piece of cheese put into the bag has a mass of 3.

In the second test case, at the output end of the second trap, there is food glue. By choosing the visiting order $p = [1, 2]$, Terry can use that food glue and stick two pieces of cheese with a mass of 1 together to get a piece of cheese with a mass of 2.

In the third test case, if Terry chooses the order $p = [1, 2]$, the food glue at the input end of the second trap will be used to stick two pieces of cheese at the first trap together, but then Terry will have to leave a piece of cheese with a mass of 1 at the end, so it's better to choose the order $p = [2, 1]$, where the last piece of cheese put into Terry's bag has a mass of 2.

In the fourth test case, by moving in the order $p = [3, 1, 2]$, at the end Terry will put a piece of cheese with a mass of $4 + 4 + 4 = 12$ into the bag. Another sequence of movements yielding this final piece of cheese is $p = [1, 3, 2]$.

Sample Input 1	Sample Output 1

```
2
2 3
1 2
2
1 1
1 0
2
2 2
0 1
3
4 4
0 0
4 4
0
```
	```
	2 1
	1 2
	2 1
	1 3 2
	```

This page is intentionally left blank.

Problem K
Kingdom Profit Kerfuffle

There is a kingdom with n cities and m one-way roads connecting these cities, where cities are numbered from 1 to n and roads are numbered from 1 to m. The i-th road is from the u_i-th city to the v_i-th city. **There can be multiple roads connecting the same pair of cities, but no road connects a city to itself.** In this kingdom, there are two pivotal cities: the capital is the first city, and the central city is the n-th city. It is known that there is at least one path from the capital to the central city.

The king wishes to tax the citizens as much as possible, so he begins to construct toll booths on each road. It is known that each toll booth constructed on the i-th road yields a profit of a_i dollars for the king. Importantly, there can be more than one toll booths on a road.

Understandably, this causes discontent among the citizens, prompting them to protest against this indiscriminate toll booth construction. To pacify the populace, the king decided to build amusement parks. Constructing an amusement park on the i-th road costs the king b_i dollars. Similar to toll booths, multiple amusement parks can be built on a road.

To ensure the citizens do not protest further, the king will build toll booths and amusement parks such that for every path from the capital to the central city, the difference between the number of toll booths and amusement parks does not exceed a certain value. Specifically, for a path from the capital to the central city that goes through roads with indices $r_1, r_2, \ldots, r_k$ in that order, the following condition must be satisfied:

$$\sum_{j=1}^{k} \mathtt{B}_{r_j} - \sum_{j=1}^{k} \mathtt{P}_{r_j} \le c$$

where

- $\mathtt{B}_{r_j}$ is the number of toll booths the king will construct on the r_j-th road.
- $\mathtt{P}_{r_j}$ is the number of amusement parks the king will build on the r_j-th road.
- c represents the tolerance level of the citizens as surveyed by the king.

In other words, for all paths from the capital to the central city, the king wants to ensure that the number of toll booths does not exceed the number of amusement parks by more than c units. Please note that we count all paths, including paths which pass through the same city or road multiple times. In this case, the elements of the sequence $r_1, r_2, \ldots, r_k$ in the above expression may not be distinct.

Naturally, the king still wants to maximize his profits. Given the kingdom's map, the citizens' tolerance level c, and the list of construction costs as well as profits for building structures in the kingdom, help the king calculate the maximum possible profit when optimally constructing these facilities, or indicate that there exists a construction plan that allows the king to earn more than 10^{18} dollars.

As a reminder, a path from the capital to the central city can be represented as a sequence of road indices $r_1, r_2, \ldots, r_k$ where:

- $u_{r_1} = 1$.
- $v_{r_j} = u_{r_{j+1}}$ for all $1 \le j < k$.
- $v_{r_k} = n$.

Input

The first line of the input contains a single integer t $(1 \le t \le 20\,000)$ – the number of test cases. t test cases follow, each is presented as below:

- The first line contains three integers n, m, and c ($2 \le n \le 1000$, $1 \le m \le 1000$, $1 \le c \le 10^6$) representing the number of cities and roads in the kingdom, and the citizens' tolerance level, respectively.
- In the next m lines, the i-th one contains four integers u_i, v_i, a_i, and b_i ($1 \le u_i, v_i \le n$, $u_i \ne v_i$, $0 \le a_i, b_i \le 10^6$) describing the i-th road, which is from the u_i-th city to the v_i-th city. It also indicates the profit earned when constructing a toll booth on the road as a_i and the cost of building an amusement park on the road as b_i. It is guaranteed that there is at least one path going from the capital to the central city.

It is guaranteed that:

- The sum of n^2 over all test cases does not exceed 10^6.
- The sum of m^2 over all test cases does not exceed 10^6.

Output

For each test case:

- Print -1 if there exists a construction plan that allows the king to earn more than 10^{18} dollars.
- Otherwise, print an integer representing the maximum profit the king can earn by optimally constructing facilities.

Sample Explanation

For the first test case, we can build 2 toll booths on the first road, granting the king 6 profit.

For the second test case, note that building only 1 toll booth on the first road will anger the population as the path $1, 3, 1, 3, 1, 3, 1$ sees 4 toll booths but no amusement parks. It turns out that the king cannot profit at all in this case.

For the fourth test case, we can put 10^{18} toll booths on the fourth road as it is not a part of any paths from the capital to the central city. It is clearly that the king can earn more than 10^{18} dollars in this case.

Sample Input 1

```
4
2 1 2
1 2 3 8
2 4 3
1 2 6 10
1 2 1 3
2 1 2 7
2 1 0 3
6 8 1
3 6 4 8
1 5 0 6
1 3 0 3
5 4 1 3
6 3 5 10
5 4 5 7
5 6 3 10
4 5 5 8
4 4 5
1 2 4 5
1 2 3 1
2 4 3 3
3 1 9 6
```

Sample Output 1

```
6
0
3
-1
```

This page is intentionally left blank.

Problem L
Linking Bits

What should you do with a lot of bits? Link them together, of course!

Today at class, Minh learnt about binary representation of integers and he was excited to practice his knowledge.

Initially, Minh has a graph of m vertices, numbered from 0 to $m-1$, with no edges. Minh then writes down the binary representation of every integer from 1 to n. After writing down the binary representation of the integer x, Minh adds an edge between every pair of nodes (i, j), satisfying that both the i-th and the j-th bits of x are 1.

More formally, for each integer x from 1 to n:

- Let $x_{m-1}x_{m-2}\ldots x_0$ be the m least significant bits of x, where x_0 is the least significant one. We may add leading zeroes so that the binary representation of x contains at least m bits.
- For all pairs of indices (i, j) $(0 \le i < j \le m-1)$ such that $x_i = x_j = 1$, Minh adds an edge connecting the i-th and the j-th vertices.

After finishing the graph with all the satisfied edges, Minh wonders if it is connected. Please help him to answer the question.

Input

The first line of the input contains a single integer t $(1 \le t \le 10^3)$ – the number of test cases. t test cases follow, each is presented as below:

- The first line contains a single integer m $(1 \le m \le 10^3)$.
- The second line contains a string demonstrating the binary representation of n $(0 \le n < 2^{10^3})$. It is guaranteed that this string does not contain leading zeroes.

Output

For each test case, output a single line containing `YES` if the graph is connected, and `NO` otherwise.

Sample Explanation

In the first test case, $m = 3$ and $n = 4$:

- No edges are added when Minh writes down the binary representation of 1 - `001`, 2 - `010` and 4 - `100`.
- Edge $(0, 1)$ is added when Minh writes down the binary representation of 3 - `011`.

The resulting graph is not connected since there is no path from vertex 2 to vertex 0 and 1.

For the second sample, with $m = 3$ and $n = 5$:

- No edges are added when Minh writes down the binary representation of 1 – `001`, 2 – `010` and 4 – `100`.
- Edge $(0, 1)$ is added when Minh writes down the binary representation of 3 – `011`.
- Edge $(0, 2)$ is added when Minh writes down the binary representation of 5 – `101`.
- Edge $(1, 2)$ is added when Minh writes down the binary representation of 6 – `110`.
- Edges $(0, 1)$, $(0, 2)$ and $(1, 2)$ are added when Minh writes down the binary representation of 7 – `111`.

The resulting graph is this case is connected.

Sample Input 1

```
2
3
100
3
111
```

Sample Output 1

```
NO
YES
```

Problem M
Monopoly

Monopoly is perhaps one of the most popular tabletop games of all time. Recently, thanks to many content creators rediscovering and playing the game, it has once again become a common bonding activity among young couples.

After a long day at work, our lovely couple *Haros* and *Iwys* cannot wait to blow off steam with some *Monopoly* action! However, as they both have to attend a meeting tomorrow morning, a **long and intimate** *Monopoly* battle is sadly not an option. Therefore, Haros and Iwys decide to go for a **quick and intense** *Monopoly Turbo* game instead.

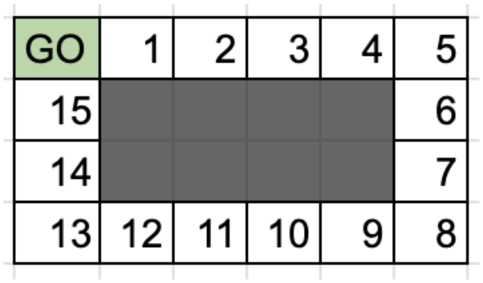


Figure M.1: A Monopoly Turbo board with $n = 15$

In *Monopoly Turbo*, there is no dice or chance involved. Instead, the game board consists of a *GO* space and n property spaces placed along the perimeter of the board. The properties are numbered from 1 to n clockwise. The i-th property costs a_i dollars to purchase. Haros and Iwys initially have x and y dollars, respectively. They take turns playing the game, with Haros making the first move. In a player's turn, they will start from *GO* and choose to travel either in clockwise or counter-clockwise direction, until they land on **the first** unclaimed property. They then have to purchase the property and claim it for themselves. The game ends when a player cannot make a purchase, either by not having enough fund left or having no property left to buy. The player who cannot make the move loses the game.

As experienced *Monopoly* players, Haros and Iwys immediately find the optimal way to play the game, and it has gotten a bit boring, so they decided to write a program that automatically plays for them and output the winner. Let's help them finish it before the meeting!

Input

The first line of the input contains a single integer t — the number of test cases ($1 \le t \le 200$). t test cases follow, each is presented two lines:

- The first line contains three integers n, x and y ($1 \le n \le 200$, $1 \le x, y \le 10^9$) — the number of property spaces and the amount of money Haros and Iwys initially have, respectively.
- The second line contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^9$) – the price of these properties.

Output

For each test case, print on a single line the name of the winner, either `Haros` or `Iwys`.

Sample Explanation

In the first test case, Haros can win by traveling in the clockwise direction and buying the property number 1, which costs \$7. Then, Iwys must choose either property 2 or 4, which both cost \$5. Haros can then purchase property 3, leaving Iwys with the other \$5 property which she cannot afford (as she only has \$4 left after her first turn).

In the second test case, as both players can afford every property, the game ends in Haros' fourth turn when there is no property left, thus making Iwys the winner.

In the third test case, Iwys can win by always choosing the cheaper available property on her turns.

Sample Input 1

```
3
4 9 9
7 5 1 5
6 10 10
1 1 1 1 1 1
7 405 6
103 1 102 2 101 3 100
```

Sample Output 1

```
Haros
Iwys
Iwys
```