Problem A. Mango

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 mebibytes

Mango is a cat that lives next door. The name comes from the fact that its color is similar to the color of mango juice. I don't know if Mango likes this name.

Like most cats, Mango is cute. Mango, which many people like, sometimes becomes the subject of humorous jokes such as "The Mango looks delicious" or "That's a cat, not a mango." Recently, Mango has become the hero of a meme about generation of the following recursive sentences.

Let's say we have the initial string M_0 and the rule string S. For any positive integer i, M_i is defined as the string in which each "`$`" character is replaced with the string M_{i-1}.

It is easy to construct the first few such strings. For example, if we say that M_0 is "That's a cat, not a mango", and S is "`That's a "$", not a "$"`", then M_0, M_1, and M_2 look as follows:

- M_0: "`That's a cat, not a mango`".
- M_1: "`That's a "That's a cat, not a mango", not a "That's a cat, not a mango"`".
- M_2: "`That's a "That's a "That's a cat, not a mango", not a "That's a cat, not a mango"", not a "That's a "That's a cat, not a mango", not a "That's a cat, not a mango""`".

Not only M_3 and M_4, but also M_{1000} can be constructed using the same principle. However, even M_5 is already quite long. Still, I wonder, for the string M_k and several pairs (a, b), what is the substring of M_k from the a-th to the b-th character, inclusive. Write a program that finds the answers to these queries.

Input

The first line contains the initial string M_0.

The second contains the rule string S.

The input strings satisfy the following conditions:

- The length of each string is not less than 1 and not more than 10^5.
- All characters in each string have an ASCII code value between 33 and 126, inclusive. Note that whitespace is not included.
- M_0 does not contain any "`$`" (ASCII code 36) characters.
- S contains at least one "`$`" (ASCII code 36) character.

The third line contains two integers k and q ($1 \le k \le 10^5$, $1 \le q \le 10^5$).

Each of the next q lines contains a query: two integers a_i and b_i ($1 \le a_i \le b_i \le 10^{18}$, $b_i - a_i < 10^5$). It is guaranteed that b_i does not exceed the length of M_k.

Output

Print q lines.

On line i, print a total of $b_i - a_i + 1$ characters: the substring of M_k from a_i-th character to b_i-th character inclusive.

It is guaranteed that, in the correct answer, the total number of characters printed (excluding line breaks) will not exceed $5 \cdot 10^5$.

Examples

standard input	standard output
It's_a_cat,_not_a_mango It's_"$",_not_"$" 1 6 1 20 18 35 49 61 29 40 41 50 5 5	It's_"It's_a_cat,_no _not_a_mango",_not _not_a_mango" o",_not_"It' s_a_cat,_n _
Ad_finitum $ 100000 4 1 10 1 2 4 10 5 8	Ad_finitum Ad finitum init
THE_END $_IS_NEVER_$_IS_NEVER_$ 88 5 1 7 3256 3257 67706 67710 111011 111017 999999999999999968 999999999999999993	THE_END IS NEVER THE_END _THE_END_IS_NEVER_THE_END_

Problem B. Koosaga's Problem

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	1024 mebibytes

Jaehyun solved a problem about the maximum cut of graph at Petrozavodsk Winter 2019 camp, so he decided to please the participants of Petrozavodsk training camp with another problem of the same nature.

You are given a simple connected graph $G = (V, E)$ with N vertices and M edges. Find the number of subsets of edges $S \subseteq E$ such that:

- The removal of edges in S makes the graph bipartite.
- $|S| \leq 2$.
- There exists no other subset $T \subseteq E$ such that $|T| < |S|$ and the first two conditions hold.

Note that S can be empty.

Input

The first line of the input contains two integers N and M ($3 \leq N \leq 250\,000$, $N - 1 \leq M \leq 250\,000$).

Then M lines follow. Each of them contains two space-separated integers u_i and v_i ($1 \leq u_i, v_i \leq N$). It describes a bidirectional edge connecting vertex u_i and vertex v_i.

It is guaranteed that the given graph doesn't have any loops or multiple edges, and the graph is connected.

Output

Print the number of subsets of edges that satisfy the given conditions.

Examples

standard input	standard output
3 2 1 2 2 3	1
4 6 1 2 1 3 1 4 2 3 2 4 3 4	3
5 9 1 2 1 3 1 4 2 3 2 4 2 5 3 4 3 5 4 5	0
12 16 1 2 2 3 3 4 4 1 5 6 6 7 7 8 8 9 9 10 10 11 11 12 12 5 1 5 2 7 3 9 4 11	2

Problem C. Gardening Game

Input file:	*standard input*
Output file:	*standard output*
Time limit:	15 seconds
Memory limit:	1024 mebibytes

Whiteking is going to play a local decorating game. The area he wants to decorate is in the form of an $N \times N$ grid in which the unit grid is represented by 1×1 square tiles, the coordinates of the top left tile are $(1, 1)$, and the coordinates of the bottom right tile are (N, N). Thus the coordinates of the tile refer to the coordinates of the lower right corner of the tile. From this, it can be seen that the tile located at (x, y) occupies a square area corresponding to $[x - 1, x] \times [y - 1, y]$. Each tile has its own beauty value, initially all tiles have a beauty value of 0.

The local decorating game is a game where the player earns points using the following three actions.

- Divide the grid into different zones by drawing a straight line horizontally or vertically. Initially, there is only one area of size $N \times N$. The lines are infinite in both directions: for example, if you draw a straight line horizontally and another one vertically, the initial area will be divided into a total of four areas.
- Select a tile and decorate the area it belongs to. As a result, the beauty of all tiles in the decorated area is increased by a given value.
- Select a rectangle on the grid and earn points equal to the beauty of the most beautiful tile in it.

Whiteking wants to know in advance how many points he will earn for every action of the third type. So he will show you an ordered list of actions he is going to perform. The actions will be given in the following format:

- "`1` a b": If a is 0, draw the straight line $x = b$, and if a is 1, draw $y = b$.
- "`2` a b X": Select the tile located at (a, b) decorate the area it belongs to, increasing the beauty of all tiles in it by X.
- "`3` a b c d": Select a rectangle with (a, b) as the top left tile and (c, d) as the bottom right tile, and earn points equal to the beauty of the most beautiful tile in it.

Help Whiteking find what will be the score for every action of the third type.

Input

The first line contains two integers N and Q ($1 \le N \le 10^5$, $1 \le Q \le 3 \cdot 10^5$).

Each of the next Q lines describes an action in the format shown in the problem statement.

For an action of type `1`, $0 \le a \le 1$ and $1 \le b \le N - 1$.

For an action of type `2`, $1 \le a, b \le N$ and $-10^9 \le X \le 10^9$.

For an action of type `3`, $1 \le a \le c \le N$ and $1 \le b \le d \le N$.

There are at most 25 000 actions of type 2, and at least 1 action of type 3.

Output

For every action of the third type, print the score that Whiteking will get for that action.

Example

standard input	standard output
3 7 3 1 1 3 3 2 1 3 -3 3 1 1 3 3 1 0 1 2 1 1 4 3 2 2 3 3 3 1 1 3 3	0 -3 -3 1

Problem D. Non-Decreasing Subarray Game

Input file:	*standard input*
Output file:	*standard output*
Time limit:	3 seconds
Memory limit:	256 mebibytes

Yuto and Platina are going to play a Non-Decreasing Subarray Game. The game is played on an array A of length N.

Yuto first says an integer, and after that, Platina says an integer. The numbers selected by the players should be in the interval from L to R, inclusive. Let the two selected integers be a and b, ordered in such a way that $a \le b$. Then the score obtained in the game is the number of pairs (i, j) such that $a \le i \le j \le b$ and the interval $[i, j]$ forms a non-decreasing subarray in array A.

We say that $[i, j]$ forms a non-decreasing subarray when, for each x and y such that $i \le x \le y \le j$, it is true that $A[x] \le A[y]$.

Yuto wants the score to be minimized, and Platina wants the score to be maximized. This game is so much fun that we are going to play it T times. All games will use the same array A, but different games might use different values of L and R.

Assuming that both players are playing optimally, find the number of points they will get in each of the games played.

Input

The first line contains two integers N and T ($1 \le N, T \le 500\,000$): the length of the array and the number of games played, respectively.

In the second line, the array values $A[1]$, $A[2]$, $A[3]$, $\ldots$, $A[N]$ are given ($1 \le A[i] \le 10^9$).

Each of the next T lines describes a game by two positive integers L_i and R_i ($1 \le L_i \le R_i \le N$): the values of L and R to use for this game.

Output

For each game, print the score in this game on a separate line.

Example

standard input	standard output
8 5 7 10 3 1 9 5 5 2 1 5 2 2 5 8 1 8 3 5	4 1 4 7 3

Problem E. Observer Game

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	256 mebibytes

Yuto and Platina are trying to play a new game called the guarding game. The game is played on an $N \times M$ rectangular grid ($N \leq M$).

The game always starts with Yuto, and the two take turns placing observers in the desired empty position on their turn.

In this game, "safe state" means that there is at least one observer in every $K \times K$ square that is completely contained within the grid.

At the moment the game is in a safe state, the game ends, and the player who played the most recent turn wins.

The players tried T games with different parameters. When both are playing their best, let's predict who will win for every game!

Input

The first line gives the number of games, T, to be played ($1 \leq T \leq 10^5$).

Then T lines follow each containing three integers N, M and K ($1 \leq N \leq 3000$, $N \leq M \leq 10^5$, $1 \leq K \leq N$), representing the dimensions of the grid and the size of the square, respectively.

Output

For each case, print the winner's name: either "`Yuto`" or "`Platina`".

Example

standard input	standard output
2 1 2 1 3 3 2	Platina Yuto

Problem F. Rhythm Game

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	256 mebibytes

The famous artist Karuna is playing the rhythm gane.

The artist is trying to hit the notes in a song. The song is a sequence of N notes.

The scoring system used in this game is as follows:

- At beginning of the song (before first note), the score is 0 and combo bonus is 0.
- Each note has its own cost. The cost of i-th note is A_i.
- The combo bonus value is equal to 0 if Karuna misses current note, or C_j if Karuna hits this note and there are j notes in a row which Karuna hits.
- If Karuna hits the i-th note and the combo length after that is j, the value of $A_i \cdot C_j$ is added to the score.
- If Karuna misses the note, the length of the combo is reset to 0. If it was non-zero before (in other words, if Karuna hit the previous note), then the combo ending score P is added to the score.
- If Karuna hits the last not in the song, the combo ending score P is added to the score as well.

Karuna's skills allow him to hit no more than K notes during the song. For every note, he may choose to hit it or to miss it, as long as he hits no more than K notes in total.

Given all the parameters, tell the maximum score Karuna can get.

Input

The first line of the input contains three integers N, K and P ($1 \leq N, K \leq 2000$, $-10^9 \leq P \leq 10^9$): the number of notes in the song, the maximum number of notes Karuna can hit and combo break score, respectively.

The second line contains N integers separated by spaces. The i-th number represents the score A_i for hitting the i-th note ($0 \leq A_i \leq 10^5$).

The third line contains N integers separated by spaces. The j-th number represents the score C_j for a combo of length j ($-10^5 \leq C_j \leq 10^5$, and for all $1 \leq j \leq N-1$, it is guaranteed that $C_j \geq C_{j+1}$).

Output

Print one integer: the maximum score Karuna can get in the Rhythm Game.

Example

standard input	standard output
5 5 1 5 4 3 2 1 5 4 3 2 1	57

Problem G. Solo Tree Game

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	256 mebibytes

Whiteking and Blackking are about to play a stone game with long wooden boards.

In this game, you use N wooden boards. The i-th wooden board has the shape of a two-dimensional stripe: a rectangle of size $1 \times A_i$. The game starts with a white stone on the first space of all wooden boards and a black stone on the last space.

On each turn, the king must move one of his colored stones. When moving, the king must move the stone to another space of the same board, but cannot jump over the opponent's stone or move to the same space. Kings take turns, and the king who cannot move in his turn is defeated.

For example, if a 6-long wood board has a white stone in cell 3 and a black stone in cell 5, the white stone can be moved to one of cells 1, 2 and 4, and the black stone can be moved to one of cells 4 and 6.

Assuming that kings are playing optimally, determine the game result.

Input

The first line of the input contains one integer N ($1 \le N \le 10^5$): the number of long wooden boards.

The second line contains N integers $A_1, A_2, A_3, \ldots, A_N$ ($2 \le A_i \le 10^9$): the lengths of the long wooden boards.

The third line contains the name of the king to move first: either "`Whiteking`" or "`Blackking`".

Output

On the first line, print the name of the king who wins. Note that the first letter is always capitalized.

Examples

standard input	standard output
2 3 3 Whiteking	Blackking
2 3 5 Whiteking	Whiteking

Problem H. Stone Catch Game

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	256 mebibytes

Yuto and Platina are going to play a stone tag game. The game is played on a two-dimensional coordinate plane, and unusually, only the part where both the x and y coordinates are from 0 to 10^9 inclusive is used as the game board.

On the game board, there are one white stone and N black stones. Initially, the white stone is located at $(0, 0)$, and the i-th black stone is located at (x_i, y_i). Initially and during the game, two or more stones may occupy the same position.

On Yuto's turn, if the white stone is at (x, y), he moves it to either $(x + 1, y)$ or $(x, y + 1)$.

On Platina's turn, she picks any one black stone she wants, and if it is at (x, y), moves it to either $(x-1, y)$ or $(x, y - 1)$.

Players take turns, and Yuto moves first. Yuto wins if the white stone escapes the game board, and Platina wins if the white stone and some black stone occupy the same position before that. In particular, if there is a black stone initially at $(0, 0)$, Platina wins before the game even starts.

Both players play the game perfectly, but the playing field is so large that we want to know in advance who will win. Find out who will win the game.

Input

The first line of the input contains an integer N, the number of black stones ($1 \le N \le 3 \cdot 10^5$).

The i-th of the next N lines contains two integers x_i and y_i: the position of i-th black stone ($0 \le x_i, y_i \le 10^9$).

Output

Print who will win the game. Note that the first letter of the name is uppercase.

Examples

standard input	standard output
1 0 1	Yuto
2 2 3 3 2	Platina
2 0 2 2 1	Platina

Problem I. Selecting Points and Segments

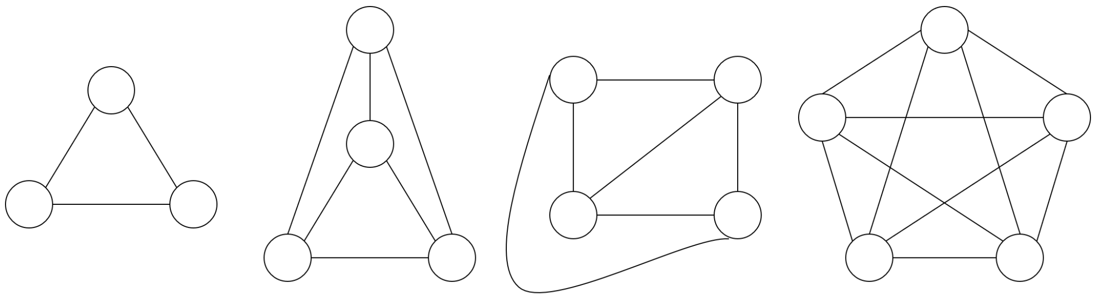
Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	256 mebibytes

Your task is to select N points on the coordinate plane and draw M straight segments connecting them such that there will be exactly K finite faces. Here, *faces* are the regions into which the plane is divided by the segments. One of the regions is infinite and should be ignored.

More formally, your configuration must satisfy the following conditions:

- The x and y coordinates of each point must be integers from 1 to 79.
- All points must have different positions.
- There must not be multiple line segments connecting two points.
- Two different line segments must not intersect except at an endpoint.
- Points other than the endpoints of the segment must not be on the segment.

In the figure below, (a) is a case in which one face is created with 3 points and 3 line segments.
(b) is a case in which 3 faces are made with 4 points and 6 line segments.
(c) is an incorrect output because there are curves, and (d) is incorrect because there are intersecting line segments.



(A) (B) (C) (D)

Input

On the first line, there are three positive integers, N, M, and K, representing the number of points, the number of segments, and the number of faces to be created, respectively ($3 \le N \le 3000$, $0 \le M$, $0 \le K$).

It is guaranteed that, for the given N, M, and K, a solution exists.

Output

On the first N lines, the print coordinates of selected points. The i-th of those lines must contain two integers x_i and y_i ($1 \le x_i, y_i \le 79$): the coordinates of i-th point.

Then print M lines describing segments. Each of these lines must containin two integers between 1 and N: the indices of points connected by a segment.

If there is more than one possible solution, print any one of them.

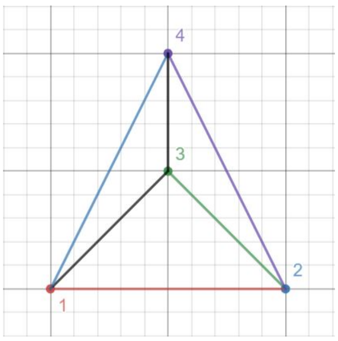
Examples

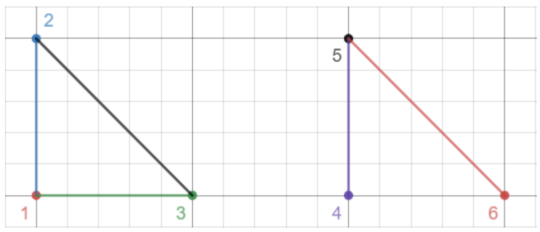
standard input	standard output
4 6 3	1 1 3 1 2 2 2 3 1 2 1 3 1 4 2 3 2 4 3 4
6 5 1	1 1 1 2 2 1 3 1 3 2 4 1 1 2 1 3 2 3 4 5 5 6

Note

The left picture shows 3 faces made with 4 points and 6 line segments.

The right picture shows 1 face made with 6 points and 5 line segments.





Problem J. Setting Maps

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	1024 mebibytes

I intend to install maps on the trail. The trail can be thought of as a graph consisting of N vertices, numbered from 1 to N, and M directed edges. There are no self-loops and no multiple directed edges.

The trail has a starting point S and a destination E. For better convenience, I want to put maps in such a way that all routes from S to E go through at least K maps.

Only one map can be installed per vertex (including starting an ending points), and the cost of installing a map at vertex v is C_v.

I want to complete the map installation at the lowest possible cost. Determine if the maps can be installed to meet the conditions, and if it is possible, print out in which vertices the maps should be installed so that the total cost is minimal possible.

Input

In the first line of the input, the number of vertices N, the number of edges M, and the minimum number of maps K are given ($2 \leq N \leq 200$, $1 \leq M \leq 500$, $1 \leq K \leq 5$).

In the second line, the starting point S and the destination point E are given ($1 \leq S, E \leq N$, $S \neq E$).

The next line contains N integers $C_1, C_2, \ldots, C_N$: the costs of installing a map in vertices $1, 2, \ldots, N$ ($1 \leq C_i \leq 10^7$).

The next M lines describe edges. The j-th of these lines contains two integers u_j and v_j: the source and destination of the j-th edge ($1 \leq u_j, v_j \leq N$, $u_j \neq v_j$).

It is guaranteed that the trail contains no self-loops and no multiple directed edges.

Output

If it is not possible to install maps to satisfy the conditions, -1 must be printed on the first line.

If map installation is possible, on the first line, print the number of vertices P at which the map should be installed so that the total cost is minimal possible. On the second line, print the labels of these P vertices, separated by spaces, in any order. If there are several possible answers, print any one of them.

Examples

standard input	standard output
3 2 5 1 3 1 60 35 1 2 2 3	-1
7 11 1 1 7 100 5 7 16 11 12 100 1 2 1 3 1 4 1 5 2 3 2 6 3 6 4 3 4 7 5 7 6 7	3 5 6 4

Problem K. Determinant

Input file:	*standard input*
Output file:	*standard output*
Time limit:	8 seconds
Memory limit:	512 mebibytes

The determinant is one of the important objects covered in linear algebra.

For a natural number n, S_n is the set of all permutations: functions from $\{1, 2, \ldots, n\}$ to $\{1, 2, \ldots, n\}$ such that all n values $f(1), f(2), \ldots, f(n)$ are different.

For $f \in S_n$, $inv(f)$ is the number of inversions in permutation f: the number of pairs (i, j) such that $i < j$ but $f(i) > f(j)$.

Consider matrix A of size $N \times N$. Let $a_{i,j}$ be the element at row i and column j. The *determinant* of A is:

$$\det(A) = \sum_{f \in S_n} (-1)^{inv(f)} \prod_{i=1}^{n} a_{i,f(i)}.$$

We have an $N \times N$ matrix A where each element is an integer. Let's run Q of the following queries.

When the integer x is given, print the value of the determinant of $A - xI$, where I is an $N \times N$ unit matrix.

Since the value can be too large, print the answer modulo prime number 998 244 353.

Input

The first line contains two integers N and Q ($1 \le N \le 500$, $1 \le Q \le 250\,000$).

The next N lines describe matrix A. The i-th of these lines contains N integers, and the j-th of these integers represents $a_{i,j}$ ($0 \le a_{i,j} < 998\,244\,353$).

Then Q lines follow, each containing one query: an integer x ($0 \le x < 998\,244\,353$).

Output

For each query, print the answer on a separate line.

Example

standard input	standard output
3 6 2 4 5 6 3 8 1 6 3 10 9 5 8 3 1	407 470 402 495 260 110