Problem A. Thomas the Tank Engine

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Thomas is helping The Railway as a freight train, specifically, he has $k-1$ wagons attached to his locomotive in a sequence (i.e. the first wagon is attached to the locomotive, the second wagon — to the first one and so on).

Today Thomas is working on a railway consisting of n stations connected by m two-way tracks. Each track connects two distinct stations. The railway has a special property: every station is located on at most one simple cycle.

The stations are so small and located so close that a single train unit (i.e. a wagon or a locomotive) always occupies a single station. Obviously, two adjacent train units should always occupy two stations that are connected by a track, as well as no station can contain two train units simultaneously.

Initially Thomas (i.e. the locomotive) is located at station v_0 and the wagons are located at stations $v_1, v_2, \ldots, v_{k-1}$. Thomas can move to a station u if:

1. v_0 and u are connected by a track;
2. station u is either empty or contains the last wagon. In other words, $u \neq v_i$ for all $0 \leq i \leq k-2$.

After such a move:

1. the locomotive will move to u, i.e. $v'_0 = u$;
2. all the wagons will move one station towards the locomotive, i.e. $v'_i = v_{i-1}$ for all $1 \leq i \leq k-1$.

Now Thomas is wondering which stations he can potentially reach from his starting position after a sequence of valid moves. Your task is to find all stations that Thomas can reach, i.e. that is possible to move the locomotive to.

Input

The first line of the input contains integers n and m ($1 \leq n, m \leq 10^5$) — the number of railway stations and tracks, respectively.

The following m lines describe the tracks: each of them contains two integers u and v ($1 \leq u, v \leq n; u \neq v$) indicating that there is a two-way track between stations u and v.

The following line contains an integer k ($3 \leq k \leq n$) — the size of the train.

The last line of the input contains k integers $v_0, v_1, \ldots, v_{k-1}$ describing the initial position of the train.

It is guaranteed that all v_i are pairwise distinct and there is a track between v_i and v_{i+1} for every $0 \leq i \leq k-2$.

Output

The first line of the output should contain a single integer p — the number of stations that Thomas can reach.

The second line should contain p integers — the numbers of stations that Thomas can reach, **in increasing order**.

Example

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

Note

Illustration for the sample:

Problem B. Broken line

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

There are n points on the plane, all located on the perimeter of a convex polygon.

Write a program that calculates the length of the shortest broken line (polygonal chain) that connects all of the given points.

Input

The first line contains a single integer n ($2 \le n \le 2\,500$), the number of the points.

Each of the next n lines contains two integers x_i, y_i ($|x_i|, |y_i| \le 250\,000$), the coordinates of the i-th point.

Output

Output a single number, the length of the shortest broken line.

Your answer will be considered correct if its relative or absolute error doesn't exceed 10^{-9}.

Examples

standard input	standard output
5 1 1 1 -1 -1 1 -1 -1 0 -2	6.82842712474619010
3 0 0 3 4 6 8	10.00000000000000000

Problem C. Columns

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

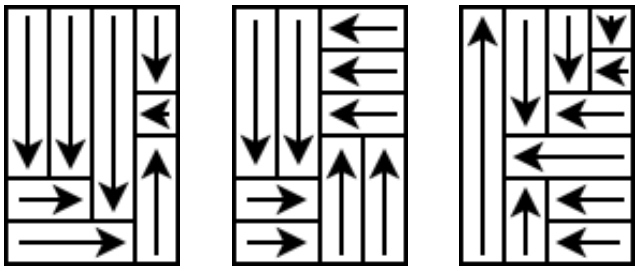
Let's say that a rectangular grid of size $n \times m$ is partitioned into oriented columns if the following conditions hold:

- each cell belongs to exactly one rectangle (let's call these rectangles *oriented columns*);
- each *oriented column* has at least one side of length 1;
- each *oriented column* has the following property: at least one side of length 1 is located on the border of the grid and exactly one of these sides is marked as the start of the *oriented column*.

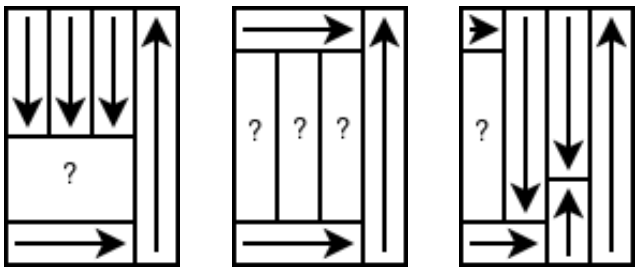
Two partitions of the rectangular grid are different if there are two cells which belong to the same *oriented column* in the first partition but to different *oriented columns* in the second partition or, in case when all *oriented columns* completely coincide, there is an *oriented column* which has a different start side.

Given the size of the grid, calculate the number of ways to partition the grid into oriented columns modulo 1 000 000 007.

The following examples are correct partitions (each arrow is oriented from the start of the corresponding *oriented column*):



The following examples are incorrect partitions (incorrect rectangles are marked with ?):



Input

The single line contains two integer numbers n and m ($1 \le n \le m \le 1000$) — the size of the rectangular grid.

Output

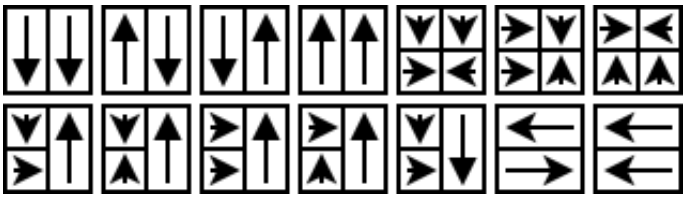
Output a single integer — the number of ways to partition the grid into oriented columns modulo 1 000 000 007.

Example

standard input	standard output
2 2	56

Note

Some of the possible partitions for the sample test:



Problem D. Domino sets

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

A board game company makes unusual domino tiles — 1×2 rectangular wood pieces where each side can have from 0 to 10^9 holes. These domino tiles are sold in n-sets — for each pair of integers i and j such that $0 \le i \le j \le n$, an n-set contains exactly one tile with i holes on one side and j holes on the other side. The 6-set is the most popular one and it consists of 28 tiles.

Recently, the company got a huge order for m-sets of dominoes and their client is ready to buy as many sets as the company can make. Unfortunately, their wood supplier went bankrupt, so the company can only use tiles that are already in stock.

As one of the employees has noticed, they can also drill additional holes in the existing tiles to get the missing tile types. However, they don't know a way to get rid of the existing holes. For example, the tile "1-2" can be transformed into "3-3", but not the other way around.

What is the maximum number of m-sets of dominoes that the company can sell to the client?

Input

The first line contains two integers separated by space — m ($1 \le m \le 10^9$) and v (the number of tile types available to the company, $1 \le v \le 1\,024$).

Each of the next v lines describes one type of tiles and contains 3 space-separated integers — i, j and k ($0 \le i, j \le m, 1 \le k$), where i and j denote the number of holes on each side of a tile and k is the number of tiles of this type.

The total number of tiles in stock does not exceed $8 \cdot 10^{18}$.

Output

Print the maximum number of m-sets the company can make from the existing tiles.

Examples

standard input	standard output
1 3 1 1 3 0 0 21 1 1 10	10
5 1 1 3 88	0

Note

In the first example there are 21 "0-0" tiles and 13 "1-1" tiles. It is possible to drill additional holes in "0-0" tiles to get ten "0-1" tiles. After this operation there are enough tiles to make ten 1-sets.

Problem E. Zebra

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 seconds
Memory limit:	256 megabyte

An infinite square grid is drawn on a plane. Each grid cell has side length 1 and all its corners are located at integer coordinates. All cells are painted either black or white in the following way:

- if two cells share a vertical side (its endpoints are (x, y) and $(x, y + 1)$), their colors are the same;
- if two cells share a horizontal side, their colors are different;
- the cell with corners in $(0, 0)$, $(1, 0)$, $(1, 1)$, $(0, 1)$ is white.

You are given a polygon with vertices at integer coordinates. Calculate the area of the inner part of the polygon that is colored in white.

Input

The first line contains a single integer n ($3 \le n \le 10^5$) — the number of vertices of the polygon. Each of the next n lines contains two space-separated integers x_i and y_i ($-10^9 \le x_i, y_i \le 10^9$), denoting the coordinates of the i-th vertex. The vertices are given in the same order as they appear on the polygon perimeter.

Output

Print the area of the white part of the polygon. The answer will be accepted if its absolute or relative error does not exceed 10^{-6}.

Examples

standard input	standard output
3 1 0 1 1 0 2	0.250000000
5 1 1 1 3 2 3 3 2 3 1	1.500000000

Problem F. Fractal NV

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

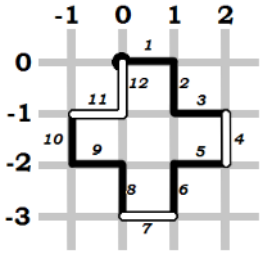
Little Arthur loves drawing fractals on an enormous square tile sheet whose grid lines are numbered with integers.

Arthur's favorite figure is the Fractal NV which is constructed in the following way.

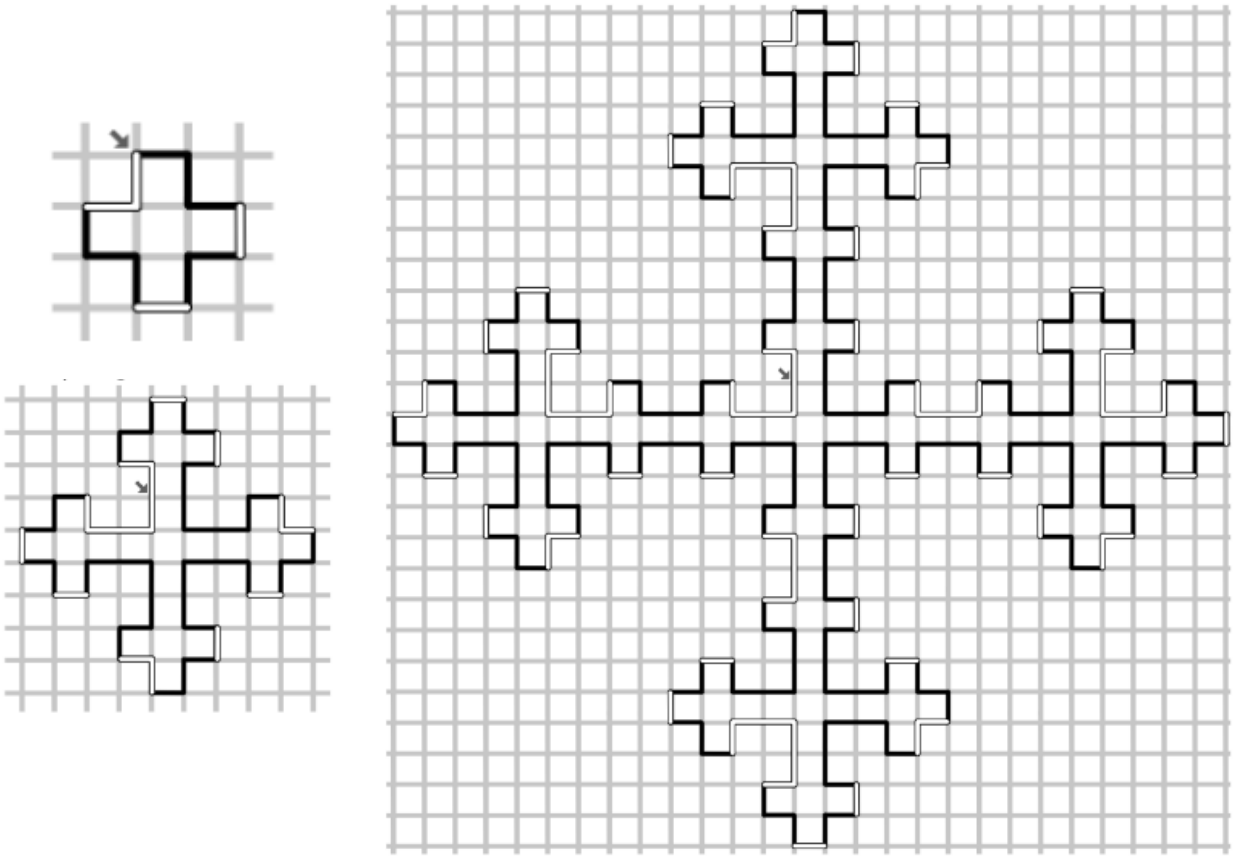
1) Initially, the 1-st level figure is drawn (resembling a + sign) whose upper-left corner has coordinates $(0, 0)$, and some of the figure's sides (segments) are colored.

2) The n-th level figure is drawn using the $(n-1)$-th level figure. At first, the base $(n-1)$-th level figure is drawn, then four new instances of the $(n-1)$-th level figure are appended to the base — two of them are flipped horizontally and appended to the leftmost and rightmost sides of the base figure and two are flipped vertically and appended to the topmost and bottommost sides, respectively. After that, the segments where the appended figures touch the base figure are removed.

Edges on the first level figure are numbered as in the picture below:



For example, if Arthur colors the 4-th, 7-th, 11-th and 12-th edges, the following first, second and third level figures will be constructed (an arrow indicates a point with coordinates $(0, 0)$):



Arthur is tired of boring picture drawing and it would be much more interesting for him if he would travel in his imagination along the edge of the figure, increasing its level every now and then. Initially he draws the 1-st level

figure and imagines he is at the point with coordinates $(0, 0)$. Then he imagines a number k_1 and starts walking clockwise along the edge of the figure. As he walks, he counts how many colored segments he has passed. After he has passed k_1 colored segments he stops and writes down his coordinates. He then imagines a new number k_2 and again starts walking clockwise until k_2 colored edges have been passed, stops and after that writes down his coordinates. Arthur repeats this process with k_3 and so on.

Moreover, at the beginning and after each stop Arthur can decide to increase his figure by k levels, that is, if he currently has the n-th level figure then it gets replaced by the $(n + k)$-th level figure. After the figure is enlarged he continues to walk along the figure from the same point where he stopped (or $(0, 0)$ if he enlarged the figure at the beginning).

Print all the coordinates which Arthur has written down!

Input

The first line contains a string of length 12 consisting of characters "+" and "-" that describes the coloring of the first level figure. If the i-th symbol $(1 \le i \le 12)$ is "+" then the i-th edge starting from point $(0, 0)$ clockwise is colored, otherwise it's not.

The second line contains a single integer n — the number of operations $(n \le 10^4)$.

Each of the next n lines contains two integers t and k:

- if $t = 1$, Arthur walks along the figure until he passes k colored segments and writes down his coordinates;
- if $t = 2$, he increases the level of his figure by k.

It is guaranteed that the sum over k of all operations where $t = 1$ doesn't exceed 10^{16}, and also the sum over k of all operations where $t = 2$ doesn't exceed 10^{16}.

Output

For each query where $t = 1$ output two integers x and y in a separate line — the coordinates Arthur has written down.

Example

standard input	standard output
---+--+---++ 5 1 2 2 1 1 6 2 1 1 8	0 -3 0 0 -2 6

Note

The sample corresponds to the pictures given in the problem statement.

Problem G. Pluses and minuses

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

There are n rows of circles drawn on a board so that they form a triangle. There are n circles in the first row (topmost), $n - 1$ circles in the second row and so on. The n-th row has only one circle. Each circle that is not in the first row is connected with the two closest circles above it. At the beginning each circle in the first row has either "+" or "−" written in it. Then the rows are filled one by one in the following way. If the circle is connected with two circles with equal signs, we write "+" in it, otherwise we write "−" in that circle.

For each row you know the symbol that is written in its first (leftmost) circle. Restore the original sequence of symbols that was written in the first row.

Input

The first line contains a single integer n ($2 \le n \le 5 \cdot 10^6$) – the number of rows.

The second line contains n characters "+" or "−" without spaces, the i-th character is the sign written in the first circle of the i-th row.

Output

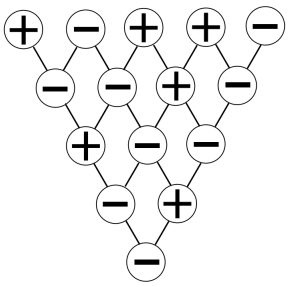
Output n characters — the signs written in the first row.

Examples

standard input	standard output
5 +-+--	+-++-
7 ---+---	-++-+++

Note

Illustration for the first sample:



Problem H. Towers of Hanoi

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 seconds
Memory limit:	256 megabytes

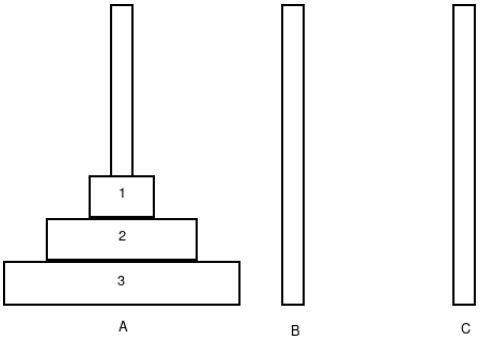
Towers of Hanoi is a well-known mathematical puzzle. It consists of three rods, denoted as A, B and C, and n disks numbered from 1 to n from smallest one to largest one which can slide onto any rod.

In the original game, the puzzle starts with the disks stacked on rod A in descending order of their sizes, with the largest one at the bottom. The player has to move all disks to rod B in the least number of steps so that they are in the same order as initially on rod A.

During the game the player has to follow the following rules:

- the player can take any topmost disk and put it onto another rod (this is considered one step);
- no larger disk can ever be placed on top of a smaller one.

The original game (initial configuration pictured below) with three disks can be solved in 7 steps (the sequence shows from which to which rod a disk is moved: $A \to B$, $A \to C$, $B \to C$, $A \to B$, $C \to A$, $C \to B$ and $A \to B$).



Now you are asked to solve a different task. Given the initial configuration of n disks where no larger disk is placed on top of a smaller one, find the least number of steps required to move all disks to rod B following the aforementioned rules.

Input

The first line contains a single integer n ($1 \le n \le 10^5$) – the number of disks.

The second line contains a string of length n consisting of characters "`A`", "`B`" and "`C`" where the i-th ($1 \le i \le n$) character denotes the rod on which the i-th disk is at.

Output

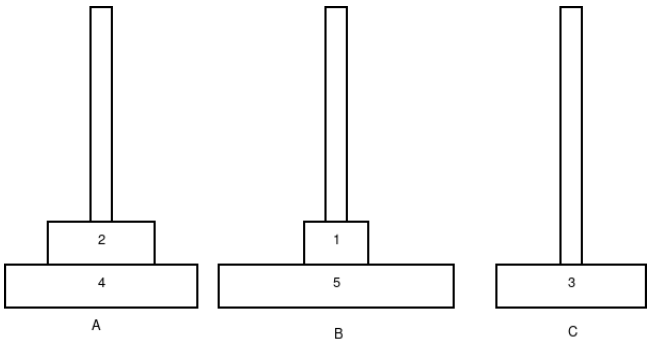
Output a single integer — the answer to the problem modulo $10^9 + 7$.

Examples

standard input	standard output
5 BACAB	10
33 AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA	589934535

Note

The initial configuration of the first example is illustrated below:



One of the possible sequence of steps is $A \to C$, $B \to C$, $A \to B$, $C \to B$, $C \to A$, $B \to A$, $C \to B$, $A \to C$, $A \to B$, $C \to B$.

The second example is the original Towers of Hanoi puzzle with 33 disks. The required number of moves is 8 589 934 591 which is 589 934 535 modulo 1 000 000 007.

Problem I. Interesting sequence

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

The first element a_1 of a sequence of natural numbers $\{a_i\}$ is 1. For each i ($i > 1$) the value of the i-th element a_i is determined by expressing value i in the following form: $m \times 2^s$, where m is an odd integer, s — a non-negative integer.

If the remainder of m divided by 4 is 1, then $a_i = a_{i-1} + 1$, and if the remainder is 3, then $a_i = a_{i-1} - 1$.

i	1	2	3	4	5	6	7	8	9	10	11	12	13
$m \pmod 4$	1	1	3	1	1	3	3	1	1	1	3	3	1
a_i	1	2	1	2	3	2	1	2	3	4	3	2	3

Given natural numbers n and p, find such index k for which $a_k = n$ and it is the p-th element in the sequence with this value.

Input

The first and only line of the input contains two space-separated natural numbers n ($n \le 100$) and p ($p \le 10^{12}$).

Output

Output a single integer, the index k for which $a_k = n$ and in the sequence a_k is the p-th element with value n. It is guaranteed that for all tests, $k \le 2^{62}$.

Examples

standard input	standard output
2 5	12
3 4	13

Problem J. Transformation of numbers

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

There is a following natural number transformation game, where a player starts with a natural number, and in each turn can do one of the following operations:

- increase the natural number by 10^y, where y is a non-negative integer,
- replace a digit in its decimal notation with any other decimal digit (the first digit cannot be replaced by zero).

Given natural numbers a and b, find the minimum possible number of moves required to obtain b from a.

For example, to obtain 107 from 31, three moves are needed: $31 \to 131 \to 101 \to 107$, but to obtain 3 000 from 2 909 two moves are needed: $2\,909 \to 2\,999 \to 3\,000$.

Input

The first line contains the natural number a and the second line contains the natural number b ($a < b < 10^{10^5}$).

Output

Output a single integer — the minimum possible number of moves required to obtain b from a.

Examples

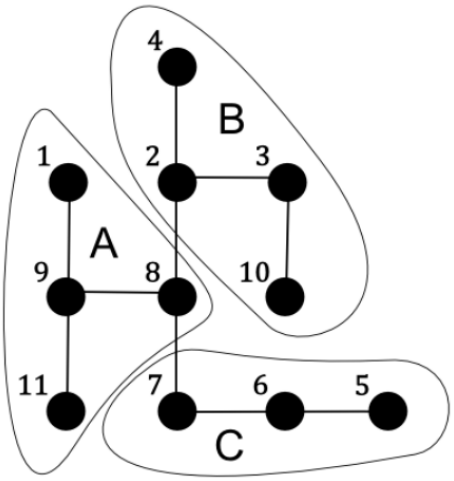
standard input	standard output
3909 7001	3
398 2015	5

Problem K. Kingdom division

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

There is a kingdom with n cities (n is an odd number) numbered from 1 to n. Cities are connected by roads in such a way that it is possible to get (maybe passing through other cities) from any city to any other. The King decided to partition his kingdom into 3 parts (A, B, C) in such a way that the following conditions hold:

- each part should contain at least one city;
- each city should be assigned to one part;
- all cities in one part form a connected component;
- number of cities in part A is equal to number of cities in part B, and it is the maximum possible.



For example, if the kingdom initially had 11 cities connected as shown in the picture then they can be partitioned into 3 parts so that parts A and B have 4 cities each, but part C has 3. One of the possible partitions is also shown in the picture.

Write a program that finds such a partition.

Input

The first line contains a single integer n ($3 \le n \le 10^5$) — the number of cities.

Then $n-1$ lines follow. The i-th line contains two integers u_i and v_i ($1 \le u_i, v_i \le n, u_i \ne v_i$) — the cities connected by the i-th road.

Output

Output a string of size n. For each i ($1 \le i \le n$), the i-th character should be "`A`" if the city i is in part A, "`B`" if the city is in part B, or "`C`" if the city is in part C.

If there exist multiple solutions, output any of them.

Example

standard input	standard output
11 1 9 4 2 9 11 2 8 10 3 8 7 2 3 9 8 7 6 5 6	CBBBAAAACBC