A. Anxious About Airdrops

1 second, 1024 mebibytes

There is a new game called "Well-known problemgrounds" out there. In this game n people are standing on a 2-dimensional grid, hoping to solve some problems.

The game proceeds as follows. At some point the judges summon an airdrop somewhere on the board. The airdrop contains a statement of a well-known problem. The players rush to the place of the airdrop immediately with constant velocity of 1 cell per second, longing to solve the task (and vainly hoping for some fresh ideas). However, they only move horizontally and vertically.

More specifically, all of them follow one and the same algorithm: at any second, if their column is not the same as the column of the airdrop, they move horizontally in the direction of the airdrop's column. If the airdrop is in the same column, they move vertically in the direction of the airdrop. The first player (or players) to reach the airdrop stops and solves the task in 1 second (the other players will continue moving towards the airdrop during this second) and announces this to everyone else. At this point everyone stops and starts waiting for the next airdrop to be deployed.

As a jury, you came up with $n - 1$ problems and want to deploy some of them in such a way that in the end all players will gather in one and the same place. You know the starting position of each player. Decide where to deploy the airdrops to achieve your goal.

Input

The first line contains a single integer $1 \le n \le 5000$ — the number of players. Each of the next n lines contains two integers x_i, y_i ($-10^8 \le x_i, y_i \le 10^8$) — the initial coordinates of the i-th player.

Output

On the first line of the output print a single integer $0 \le k \le n - 1$ — the number of airdrops deployed. Then output k lines, each containing two integers x and y ($-10^8 \le x, y \le 10^8$): the i-th of those lines should contain the coordinates of the point where the i-th airdrop is deployed. If there are several possible answers, output any one of them. In particular, you don't have to minimize k.

input
2 0 0 4 5

output
1 4 0

input
8 3 4 4 3 -3 4 4 -3 -3 -4 -4 -3 3 -4 -4 3

output
1 0 0

In the first example the distances between the players and the location of the airdrop are 4 and 5 respectively. The first player will be the first to reach the location, but since he needs 1 second to solve the task, the second player will also reach the location.

In the second example all players are at the same distance $3 + 4 = 7$ from the airdrop, so everyone will get there simultaneously.

B. Base By Base

2 seconds, 1024 mebibytes

This is an interactive problem.

There is a secret positive integer n between 1 and 10^{15} (both inclusive).

You may ask questions of the following type: "Given base $2 \le b \le 10^9$, what is the sum of digits in base b representation of the integer n?" You have to guess n in no more than 2 queries.

Interaction

Interaction is started by the jury program sending a line with the number of scenarios t ($1 \le t \le 1000$).

In each scenario, the interaction is started by your program sending the request in the format "? b", where b ($2 \le b \le 10^9$) is the base of the numeral system to represent n. The jury program prints a line with the answer s — the sum of the digits of representation of n in the b-based numeral system. When you are ready to tell the value of n, print "! n". This action is not counted as a query.

If you use more than two queries or your answer is incorrect, your solution is rejected and interaction stops immediately. Otherwise the next scenario is started (or the interaction stops if this was the last scenario).

You may assume that for each scenario the value of n is fixed before the start of interaction (i.e. that the interactor is not adaptive). All integers in the process of interaction (b, s and n) are sent and received in the usual decimal system.

input
1 2023 1

output
? 1000000 ? 2023 ! 2023

C. Cheater Changes Costs

2 seconds, 1024 mebibytes

There are n students who are going to play "Dictator Game" in a class of professor Shilish.

This game will consist of $n \cdot (n - 1)$ rounds, one round for each ordered pair (i, j) of people ($i \ne j$).

You are given a positive integer s. In each round s coins will be distributed between people i and j if they agree. In the round for pair (i, j), player i is proposer and player j is responder. Player i should propose to player j some integer number of coins c ($0 \le c \le s$) to take. If j agrees, then in this round player i gets $s - c$ coins and player j gets c coins. Otherwise both players get 0 coins. At the end of the game the score of each player is the total number of coins that this player has received during the game.

All players decided to stick with the following strategy: each player i selects an integer a_i ($0 \le a_i \le s$). In each round where player i participates (in any role) they want to get at least a_i coins from this deal in case of agreement. So they will propose some number of coins and agree with some number of coins if this condition will be satisfied in case of agreement. The numbers $a_1, a_2, \ldots, a_n$ are known to everyone before the game and will be used by players to make optimal proposals and get the highest score at the end of the game. In each round proposer will try to get an agreement, if it is possible.

Player i got the information about array a before other players. To get a higher score, they can change value a_i to any other integer between 0 and s. Note that only player i will change their value. What is the highest possible score they can get if the value is selected optimally, and what is the number of optimal choices? Find the answer for all players i. Note that you should calculate these answers for all players independently.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \le t \le 10^5$). The descriptions of the test cases follow.

The first line of each test case contains two integers n, s ($1 \le n \le 5 \cdot 10^5$, $1 \le s \le 10^{12}$) — the number of players and the number of coins that will be used in each round.

The second line of each test case contains n integers $a_1, a_2, \ldots, a_n$ ($0 \le a_i \le s$).

It is guaranteed that the sum of n for all test cases does not exceed $5 \cdot 10^5$.

Output

For each test case print n lines. On the i-th line print two integers: the highest possible score that player i can achieve and the number of ways to select a_i to get this highest possible score.

input
5 1 1000000000000 0 3 6 3 5 6 5 100 45 30 70 55 45 4 6 6 2 6 6 4 5 5 0 3 0

output
0 1000000000001 2 1 6 1 6 2 320 1 305 1 405 1 345 1 320 1 8 1 0 7 8 1 8 1 20 1 11 1 20 1 11 1

Statement is not available on English language

E. Eirt Eht Esrever

5 seconds, 1024 mebibytes

Vasya plays with his favourite string s of length n. He has a sequence of his favourite numbers, each of the numbers being a rational number $n_i/d_i \in [0, 1]$. For each i from 1 to n, Vasya takes a string equal to the prefix of s of size i, reverses it with probability $1 - n_i/d_i$ and puts the resulting string into a set. He then builds a *trie* (see Notes for the definition) on strings from the set. He asks you to calculate the expected number of vertices in the resulting trie.

Input

The first line of the input contains an integer n, the length of the string ($1 \le n \le 2 \cdot 10^5$). The second line of the input contains a string s of length n consisting of lowercase English letters. The third line contains exactly $|s|$ integers p_i ($0 \le p_i < 998\,244\,353$), where p_i is equal to $n_i d_i^{-1} \mod 998\,244\,353$.

Output

Output a single integer — the answer to the problem modulo $998\,244\,353$. That is, if the expected number of vertices is equal to the irreducible fraction a/b, you should print $ab^{-1} \mod 998\,244\,353$. It can be proven that the answer is rational, and that if the answer in the irreducible form is a/b, then b is not divisible by $998\,244\,353$ under the constraints of this problem.

input
2 aa 324 435

output
3

input
2 ab 499122177 499122177

output
499122180

input
6 wordle 23 45 67 89 87 65

output
376824090

input
7 abacaba 0 1 598946612 332748118 598946612 1 0

output
266198508

input
6 banana 0 0 0 0 0 0

output
16

input
7 trytrie 1 1 1 1 1 1 1

output
8

In the first example the set will always consist of two strings, `a` and `aa`, and thus the number of vertices in the trie will always be equal to 3.

In the second example the the set will either be $\{\texttt{a}, \texttt{ab}\}$ or $\{\texttt{a}, \texttt{ba}\}$, each with probability 1/2. The expected number of vertices will be equal to 3.5.

A *trie* built on a collection of distinct strings $s_1, \ldots, s_n$ is a special rooted tree with letters written on the edges and some of the vertices being marked. This tree has the following properties:

- All leaves are marked.
- There are exactly n marked vertices.
- No two edges going down from one vertex have the same letter written on them.
- The set of strings obtained by writing down the letters on the paths from the root to the marked vertices coincides with $\{s_1, \ldots, s_n\}$.

F. F For Function

6 seconds, 1024 mebibytes

Let w be a positive integer. For positive integers a and b, let's define a function $f_w(a, b)$ as the largest integer $k \geq 0$ such that for all integers $0 \leq i \leq k$ the number $a + w \cdot i^2$ is divisible by $b + i$. If there is no such k, let $f_w(a, b) = 0$. It can be shown that $a + w \cdot i^2$ cannot be divisible by $b + i$ for all $i \geq 0$, that is, the function $f_w(a, b)$ is well defined.

You are given three positive integers w, m_a, and m_b. Find the sum $\sum\limits_{a=1}^{m_a} \sum\limits_{b=1}^{m_b} f_w(a, b)$.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \leq t \leq 5$). The descriptions of the test cases follow.

The only line of each test case contains three positive integers m_a, m_b, w ($1 \leq m_a, m_b \leq 10^{18}$, $1 \leq w \leq 10^6$).

Output

For each test case print a single integer — the answer to the problem.

input
5 1 1 239 6 3 1 15 10 3 1000 100 11 25 1000 50

output
5 6 19 1519 34

G. Giant Gorilla's Gift

8 seconds, 1024 mebibytes

One day you met a giant gorilla who gave you an array of n integers $a_1, a_2, \ldots, a_n$. You don't know what to do with this gift, so you want to refuse it, but unfortunately the giant gorilla insisted that you have to answer m queries. There are three types of queries:

1. l, r, k — compute the value of the function $\mathrm{gorilla}(l, r, k)$ (defined below);
2. i, x — perform the assignment $a_i := x$;
3. l, r, c, b — for all indices i ($l \leq i \leq r$) for which the inequality $c \cdot a_i \leq b$ holds, perform the assignment $a_i := c \cdot a_i$.

The function $\mathrm{gorilla}(l, r, k)$ is defined as follows. First, all numbers from $a_l, a_{l+1}, \ldots, a_r$ that have **at most** k distinct **prime** divisors are written down **in the same order**. Let the resulting sequence be $y_1, y_2, \ldots, y_s$ (here, s is the number of obtained numbers, $0 \leq s \leq r - l + 1$). Then the value of $\mathrm{gorilla}(l, r, k)$ is the alternating sum of the obtained numbers: $y_1 - y_2 + \ldots + (-1)^{s+1} \cdot y_s$. If $s = 0$, this sum is assumed to be 0.

Input

The first line contains integers n and m ($1 \leq n, m \leq 2 \cdot 10^5$) — the number of elements in the array and the number of queries.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($1 \leq a_i \leq 10^6$) — the elements of the array.

The next m lines describe the queries (all parameters are integers):

- "$\texttt{1}\ \ l\ \ r\ \ k$" ($1 \leq l \leq r \leq n$, $1 \leq k \leq 10^6$) — a query of the first type;
- "$\texttt{2}\ \ i\ \ x$" ($1 \leq i \leq n$, $1 \leq x \leq 10^6$) — a query of the second type;
- "$\texttt{3}\ \ l\ \ r\ \ c\ \ b$" ($1 \leq l \leq r \leq n$, $1 \leq c, b \leq 10^6$) — a query of the third type.

Output

Output the answer to each query of the first type on a separate line.

input
10 5 5 14 62 36 9 1911 10 12 13 35 1 2 6 2 2 5 120 1 3 9 1 3 4 6 7 993 1 1 10 3

output
-21 13 1736

H. Hidden Hierarchy Hints

2 seconds, 1024 mebibytes

This is an interactive problem.

The jury has a hidden tree with n vertices, numbered from 1 to n. The tree has $n - 1$ edges, the i-th edge connects vertices s_i and f_i. You do not know the edges, but you can ask jury questions about the tree.

Each query you make must contain one vertex v and a binary string b of length $n - 1$. The jury does the following things for the query:

- The tree is considered as a rooted tree with vertex v being the root.
- Each edge of the tree becomes directed. More specifically, if $b_i = 1$, then the i-th edge becomes directed from s_i to f_i. Otherwise, it is directed from f_i to s_i.
- The answer to the query is the number of edges that are directed towards the root. In other words, the answer to the query is the number of tree edges for which the distance between its start to v is larger than the distance between its end and v. Here the distance between two vertices is the number of edges on the simple undirected path between these vertices.

Your task is to guess all s_i and f_i in no more than $10\,000$ queries.

Interaction

The first line contains the only integer n ($2 \leq n \leq 500$) — the number of vertices in the tree.

After reading n, you can make no more than $10\,000$ queries of form "$\texttt{?}\ \ v\ \ b$", where v is the vertex number ($1 \leq v \leq n$), and b is a binary string of length $n - 1$.

For each query, the jury responds with a line containing a single integer, denoting the number of edges that are directed towards v.

When you are ready to tell all s_i and f_i, print "$\texttt{!}\ \ s_1\ \ f_1\ \ s_2\ \ f_2\ \ \ldots\ \ s_{n-1}\ \ f_{n-1}$".

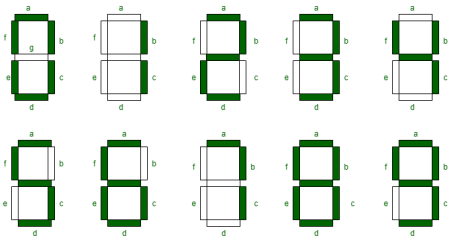
You may assume that the tree itself and all s_i and f_i are defined before the start of interaction (i.e. that the interactor is not adaptive).

input
3 2 2 1
output
? 1 01 ? 2 00 ? 3 11 ! 2 3 2 1

I. Interpreting Indicated Integers

2 seconds, 1024 mebibytes

The display of a pocket device consists of several seven-segment indicators arranged in a row. Each indicator displays a decimal digit:



Unused digits on the left are displayed as zeros, meaning that all indicators are always used.

The device can be taken out of the pocket in the correct orientation or rotated 180 degrees. The probability of each of these events is 50 percent. In some cases, the value is interpreted uniquely even after rotating (for example, rotating 73 results in a display that is not a valid number, so we can restore the orientation, and for 22 the value is the same regardless of the orientation), but in some other cases, there may be two possible interpretations (for example, 92 or 26).

Given the number of the indicators on the display, find the probability that the displayed value **cannot** be interpreted uniquely (considering that all values on the indicator are equiprobable).

Input

The input contains the single integer n ($1 \le n \le 24$).

Output

Output a single real number — the probability that the displayed value cannot be interpreted uniquely. The value shall be **absolutely** precise and shall be printed **without extra leading or trailing zeroes**.

input
2
output
0.3

Statement is not available on English language

K. Kalevich's Key Knowledge

2 seconds, 1024 mebibytes

In his book "The Art of Squares" famous artist Kalevich wrote about the main secret of his method to create square paintings: first Kalevich thinks of two digits $a > 0$ and b, then the size of the new painting is the **smallest** positive integer x with the following properties:

- The first digit of the decimal notation of the number x is the given digit a.
- The last digit of the decimal notation of the number x is the given digit b.
- The number x is a perfect square, meaning there exists an integer y such that $y \cdot y = x$.

Find such x by the digits a and b that Kalevich thought of, or report that such x doesn't exist.

Input

The first line of the input contains a single integer a ($1 \le a \le 9$), specifying the first digit of the number x. The second line of the input contains a single integer b ($0 \le b \le 9$), specifying the last digit of the number x.

Output

Print the smallest value of the perfect square starting with a and ending with b, or -1 if such x does not exist.

input
4 4
output
4

input
2 6
output
256

input
1 7
output
-1

input
1 0
output
100

In the first sample, the answer is the number 4 — for a one-digit number, the first digit is at the same time the last one.

In the second sample, the answer is the number 256 ($16 \cdot 16$), as 26, 206, 216, 226, 236, and 246 are not perfect squares.

L. Latin Language Lesson

2 seconds, 1024 mebibytes

This is an interactive problem.

On a boring Latin Language lesson, Julius and Octavian are playing the following game. A valid non-empty Roman numeral is written on the board.

The following table from Wikipedia displays how Roman numerals are written:

Individual decimal places	Thousands	Hundreds	Tens	Units
1	M	C	X	I

2	MM	CC	XX	II
3	MMM	CCC	XXX	III
4		CD	XL	IV
5		D	L	V
6		DC	LX	VI
7		DCC	LXX	VII
8		DCCC	LXXX	VIII
9		CM	XC	IX

The composition rule:

- The numerals for 4, 9, 40, 90, 400, and 900 are written using "subtractive notation" where the first symbol is subtracted from the larger one: for example, 40 ("`XL`") is written as '`X`' (10) subtracted from '`L`' (50). These are **the only** subtractive forms in standard use.
- A number containing several decimal digits is built by appending the Roman numeral equivalent for each, from highest to lowest.
- Any missing place (represented by a zero in the place-value equivalent) is omitted.
- The largest number that can be represented in the Roman notation is 3999 (MMMCMXCIX).

Julius can choose who makes the first move. Then the players take turns making moves. A move consists of adding one letter to the left or to the right of the number already written on the board in such a way that the resulting number remains a valid Roman numeral. The player who cannot make the next move loses.

Your task is, given the initial number, to win while playing for Julius, and the computer will be playing for Octavian. You may choose your turn (first or second).

Interaction

One test consists of several scenarios. First, the jury program prints a line with an integer t, the number of scenarios ($1 \le t \le 100$).

In each scenario, the interaction starts with the jury program printing a line with a valid non-empty Roman numeral. Then the participant chooses the player to move first: the participant themselves (Julius) or the computer (Octavian). To do this, they print a line with a single integer: `1` if the participant makes the first move, or `2` if the computer makes the first move. Then the players take turns making moves, printing the Roman numeral that is obtained after their move. Each move is printed on a single line.

If the number is not a valid Roman numeral, or if the number cannot be obtained by adding one letter to the left or to the right of the previous number, then the corresponding player loses. The jury program will not attempt to make invalid moves. Instead, if the jury program loses, it outputs the text `GG` on a separate line and proceeds to process the next scenario, if any, or terminates otherwise.

The jury program can use various strategies, and its actions may depend on the participant's moves (i.e., the interactor is adaptive).

input

```
2
MMII

MMMII

GG
MMIV


GG
```

output

```

2

MMMIII


1
MMMIV
```

M. Minor Maze Modifications

2 seconds, 1024 mebibytes

A maze can be represented as a rectangle with r rows and c columns. Each cell in the maze can either contain a wall or be empty. Two cells are guaranteed to be empty: the cell at the intersection of the first row and the first column and the cell at the intersection of the r-th row and the c-th column.

It is possible to move between two empty cells if they share a side (i.e., if one coordinate is the same and the other differs by 1). It is not allowed to move into a cell with a wall.

Maria plans to remove **exactly two** walls, placing empty cells instead of them. In how many ways can she choose two distinct wall cells to be removed (order of removal does not matter) so that, after the removal, there is a path from cell $(1, 1)$ to cell (r, c) consisting only of empty cells?

Two ways to remove two walls are considered distinct if at least one of the walls is removed in one way and not removed in the other.

Input

The first line of the input contains two integers r and c ($1 \le r, c \le 500$). Each of the following r lines contains c characters. These lines describe the maze. The character '`X`' corresponds to a wall, and the character '`.`' corresponds to an empty cell. It is guaranteed that the first cell in the first row and the last cell in the last row are empty.

Output

Output a single integer: the number of ways to remove exactly two walls so that it is possible to move from cell $(1, 1)$ to cell (r, c).

input

```
3 5
..X..
.XX..
..XX.
```

output

```
7
```

input

```
3 2
..
X.
..
```

output

```
0
```