6^{th} Virginia Tech High School Programming Contest

Dec 14, 2019

As a reminder, here are the key rules under which this contest is conducted:

- Teams may not communicate with another human during the contest about the problems.
- Teams may not use more than 1 computer.

This problem set contains a large number of problems (13) which target a variety of skill levels. You are not expected to solve all of them, particularly if this is your first programming contest!

Enjoy!

This page is intentionally left blank.

Problem A
Field Trip

Photo by Steve Harvey on Unsplash

You and your classmates are going on an exciting field trip to a downtown museum. Because the museum is an hour away, three buses of identical capacity and a van are dispatched as a means of transportation to the museum and back. The buses will first start loading only students, and once all students have been loaded then teachers will begin to fill in the remaining spots on the buses. Any remaining teachers will ride on the van. To make the bus ride more exciting, all the students are hoping for a "teacher free bus ride"! A teacher free bus ride is when none of the teachers are on a bus.

Students are grouped in n class sections of different sizes. Each class section is assigned a number from 1 to n. All students in one class section must ride on the same bus. The buses will board class sections in increasing order by number. In other words, the first bus will load class numbers 1 to i, the second bus will load class numbers $i+1$ to j, and the third bus will load class numbers $j+1$ to n.

Given the sizes of each of the class sections, determine if it is possible to load students onto 3 identical buses and end up with a "teacher free bus ride!"

Input

The first line of input contains one integer N, the number of class sections ($3 \leq N \leq 1,000,000$). The second line of input contains N integers, the i^{th} integer represents the size of class section number i. Class section sizes can range from $[1, 10\,000]$.

Output

If it is possible to load the students onto three identical buses in the above-described fashion and have a teacher free bus ride, then output two integers i and j where i is the number of the class section which is to be loaded last into the first bus, and j is the class section which is to be loaded last into the second bus. We can assume the third bus will load class sections $j+1$ to n.

If it is not possible, print "`-1`".

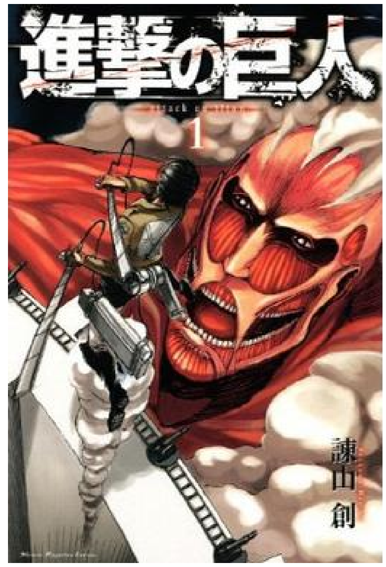
Sample Input 1	Sample Output 1
`3` `3 3 3`	`1 2`

Sample Input 2	Sample Output 2
`3` `9 10 11`	`-1`

Sample Input 3	Sample Output 3
9 1 2 3 1 2 3 1 2 3	3 6

Sample Input 4	Sample Output 4
9 1 2 3 1 2 3 1 2 10	-1

Problem B
Escape Wall Maria



Source: WikiMedia (fair use)

Wall Maria has been broken! Eren must evacuate as soon as possible from his house. He must find the fastest route to escape within Wall Maria before the titans rush in. Wall Maria is represented as a $N \times M$ grid in which Eren can move horizontally or vertically.

There are burning houses and buildings which prevent Eren from passing through them. The burning houses and buildings are represented as '`1`'. Unburned or safe areas are represented as '`0`'. There are some areas which can be entered but only from a specific direction. These areas can be represented by either '`U`', '`D`', '`L`', or '`R`'. For example, if there is an '`R`' that means that area can only be entered from the right neighboring tile within Wall Maria's grid. Similarly, '`U`' tiles can only be entered from above, '`D`' tiles can only be entered from below, and '`L`' tiles can only be entered from the left.

Eren knows the time t at which the titans will rush in. It takes 1 unit of time to traverse 1 zone (which corresponds to 1 tile in the grid). Once he reaches any border of Wall Maria he is safe.

Eren's starting position is represented by the letter '`S`'. If Eren escapes at or before time t, he is safe. Given his position within Wall Maria determine if it is possible to escape. If it is possible determine the number of zones that must be traversed to lead to the quickest escape.

Input

The input consists of a single test case. The first line contains three integers t $(1 \leq t \leq 200)$, N $(1 \leq N \leq 100)$ and M $(1 \leq M \leq 100)$. The rest of N lines will be Wall Maria's grid containing characters '`1`', '`0`', '`S`', '`U`', '`D`', '`L`', or '`R`'. There is exactly one '`S`' in the input.

Output

If it is possible to escape Wall Maria, output the minimum number of zones that must be traversed to escape. If it is not possible to escape, print "`NOT POSSIBLE`"!

Sample Input 1	Sample Output 1
2 4 4 1111 1S01 1011 0U11	2

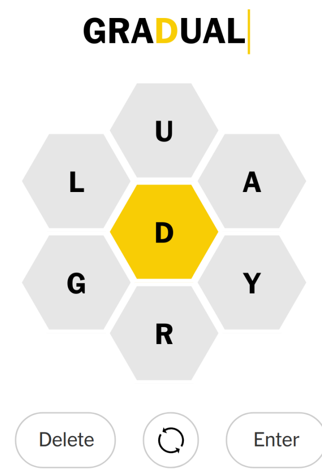
Sample Input 2	Sample Output 2
2 4 4 1111 1S01 1011 0L11	NOT POSSIBLE

Sample Input 3	Sample Output 3
1 4 4 1S01 1001 1011 0U11	0

Problem C
Spelling Bee

The New York Times publishes a daily puzzle called the "Spelling Bee." In this puzzle, 7 letters are shown in a hexagonal arrangement of 6 letters around a center letter. The task is to come up with as many words as possible that

- contain only letters that are displayed in the hexagon,
- are at least of length 4, and
- contain the center letter.



An example of the Daily NY Times Spelling Bee Puzzle

A letter may be used more than once, and not all letters need to be used.

After playing for a while, you get stuck, but then you remind yourself that the Linux distribution on your computer comes with a machine-readable file of $102\,305$ dictionary words in `/usr/share/dict/words`!

You decide that even if you can't excel at the Spelling Bee you can still excel at programming, so you decide to write a program that finds all solutions to a Spelling Bee puzzle from your dictionary.

Input

The input consists of a single test case, which starts with a line with 7 distinct lowercase English letters. The first of these letters is the center letter. The next line contains an integer n ($1 \leq n \leq 102\,305$), the size of the dictionary. This line is followed by n lines, each containing a dictionary word of l lowercase English letters ($1 \leq l \leq 24$).

Output

Output the word list matching the Spelling Bee puzzle in the order in which they appear in the dictionary. You are guaranteed that at least one dictionary entry will match.

Sample Input 1	Sample Output 1
drulyag 27 dryad duly spelling multiplexed janna lard dryly the instances gradual gradually dual inimically off dullard grad equipage gladly mauritania drug a drag pickering yard daddy on lallygag	dryad duly lard dryly gradual gradually dual dullard grad gladly drug drag yard daddy

Problem D
I.O.U.


Photo by Sharon McCutcheon on Unsplash

You are developing a new app intended to simplify expense-sharing among groups of friends. This app will allow them to keep track of who encountered an expense and how it should be shared with others through the form of I.O.U.s. For instance, if Alice pays for a meal shared with Bob and Carol, and Bob's and Carol's share were $\$5$ and $\$10$, respectively, then Bob would issue an I.O.U. over $\$5$ to Alice and Carol would issue an I.O.U. over $\$10$ to Alice.

Your app will maintain a ledger of who owes whom. Note that cycles can occur: For instance, if Bob initially owes Alice $\$10$ and later pays a $\$5$ expense on behalf of Alice, Alice would issue an I.O.U. over $\$5$ to Bob. This I.O.U. would then cancel out, or reduce, the I.O.U. Alice holds from Bob from $\$10$ to $\$5$. It's also possible for cycles to involve more than 2 people.

Your app will be given a list of I.O.U.s issued and settle them as much as possible by considering all cycles and reducing each debt in a cycle by the minimum amount of debt occurring in the cycle. After all cycles are considered and canceled, your app should output who owes whom how much. If there are multiple ways in which cancelation can occur, you may choose any of them as long as there are no cycles left at the end. However, you may not introduce I.O.U.s between friends that never gave an I.O.U. to each other, e.g., if Alice owes Bob money, and Bob owes the same amount to Carol, you cannot remove Bob from the picture and declare that Alice now owes Carol.

Input

The input consists of a single test case. The first line contains two integers n and m ($1 \leq n \leq 100, 0 \leq m \leq 10\,000$), where n denotes the number of friends and m denotes the number of I.O.U.s issued. Friends are numbered 0 to $n-1$. This is followed by m lines containing three integers a, b, c ($0 \leq a < n, 0 \leq b < n, a \neq b, 0 < c \leq 1\,000$) denoting an I.O.U. given by friend a to friend b over c dollars. Any friend i holds at most one I.O.U. from any friend j ($i \neq j$), but friend i may hold an I.O.U. from friend j at the same time that friend j holds an I.O.U from i.

Output

First, output a single number p, denoting the number of I.O.U.s left after canceling all cycles. Then, on the following p lines, output the I.O.U.s that are left in the same form in which they appear in the input (e.g. using 3 integers a, b, c denoting that friend a owes friend b c dollars). Do not include any I.O.U.s fully canceled, i.e., all the I.O.U.s you output must have $c > 0$.

Sample Input 1	Sample Output 1
4 5 0 1 10 1 2 10 0 3 10 3 2 10 2 0 20	0

Sample Input 2	Sample Output 2
2 2 0 1 20 1 0 5	1 0 1 15

Sample Input 3	Sample Output 3
4 5 0 1 10 1 2 10 0 3 10 3 2 10 2 0 10	2 3 2 10 0 3 10

Problem E
Musical Trees


Source: Pixabay

It's Christmas time and JW's 1-dimensional shop is selling Christmas trees. However, the demand for trees is much higher than the number of trees available. Hence, JW has come up with a special strategy to help decide who gets what tree: a game of Musical Trees!

Musical Trees is much like the game Musical Chairs. There's a set of trees lined up in a straight (1-dimensional) line. At first, everyone starts by wandering around the store while the music is playing. When the music stops, everyone runs to the nearest tree (the tree the smallest distance away) and whoever reaches a tree first gets to the claim that tree. Since people are lazy, they will only ever try to run to the closest tree to them, and hence multiple people may try to get the same tree. Note this means some trees may be unclaimed if they are closest to no one. Also, like in Musical Chairs, no tree can be claimed by more than one person.

The music has just stopped in Musical Trees and as everyone is running to the closest tree, you want to figure out the number of people who won't get any tree.

Input

The first line consists the number of people n ($1 \leq n \leq 100$) and the number of trees m ($1 \leq m \leq 100$). The next line contains n integers $p_1, p_2, \ldots, p_n$, the position of all the people when the music stops ($1 \leq p_i \leq 1\,000$). The last line contains m integers $t_1, t_2, \ldots, t_m$, the position of all the trees ($1 \leq t_i \leq 1\,000$). No two people or two trees will have the same position. Some people may try to cheat though, and will already be at the same position as a tree when the music stops. Note that if a person has more than one closest tree to them, they will always go for the one with the smallest p_i.

Output

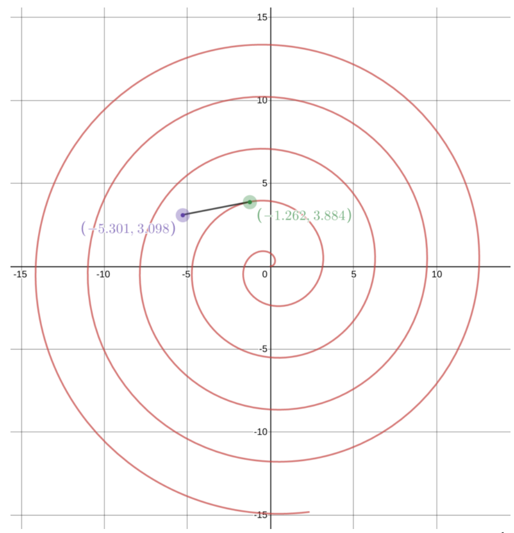
Output the number of people who won't get a tree.

Sample Input 1	Sample Output 1
2 3 1 4 2 4 5	0

Sample Input 2	Sample Output 2
3 2 1 5 10 4 6	1

Sample Input 3	Sample Output 3
2 3 3 1 2 5 4	1

Problem F
Archimedes



Archimedes's Spiral, $b = \frac{1}{2}$

Archimedes is the name of a new 2D video game, which is very simple: the player hits start, at which point their avatar moves along an Archimedean spiral starting from the origin. When they hit a button, the avatar detaches from the spiral in the direction it is currently moving. The goal is to hit a target. The avatar's path after it detaches must not intersect with any part of the spiral.

To help study this game, you're asked to write a program that given the spiral's angular velocity computes the point at which the player would need to press the button to release the avatar from the spiral in order to hit the target.

An Archimedean spiral is created when a point moves with constant velocity along a line that rotates with constant angular velocity around the origin. When expressed in polar coordinates (r, ϕ), an Archimedean spiral has a simple formulation: $r = b\,\phi$ where r is the distance from the origin and ϕ is the angle between the point, origin, and the unit vector $(1, 0)$. b is a constant that determines the distance between successive rings of the spiral.

Input

The input consists of a single line with 3 real numbers b, t_x, and t_y, denoting the parameter b ($0.01 \le b \le 10$) of the spiral described by $r = b\,\phi$ and the x, y coordinates of the target $T = (t_x, t_y)$, restricted by $-10\,000 \le t_x, t_y \le 10\,000$. It is guaranteed that $\sqrt{t_x^2 + t_y^2} > 2\pi b$, i.e., the avatar will stay on the spiral for at least one full 360 degree turn. It is also guaranteed that the distance from point T to the closest point on the spiral will be greater than 10^{-3}. There may be up to 12 significant digits in b, and up to 3 digits after the decimal point in t_x and t_y.

Output

Output the x, y coordinates of the point on the spiral where the avatar should leave the spiral, continue in the direction it is moving, and hit the target without intersecting the spiral.

Your answer will be considered correct if the absolute or relative error of both x and y does not exceed 10^{-5}.

Sample Input 1	Sample Output 1
`0.5 -5.301 3.098`	`-1.26167861 3.88425357`

Sample Input 2	Sample Output 2
`0.5 8 8`	`9.21068947 2.56226688`

Sample Input 3	Sample Output 3
`1 8 8`	`6.22375968 -0.31921472`

Sample Input 4	Sample Output 4
`0.5 -8 8`	`-4.36385220 9.46891588`

Sample Input 5	Sample Output 5
`0.5 0 -8`	`-3.60855706 -3.61140618`

Problem G
Perfect Skyline

Source: pixabay

Zara, an aspiring architect and urban planner, has drawn out what she considers to be the perfect skyline. As Zara is still aspiring she must use her young daughter, Pippa, to test out her designs. In order to test out the designs Pippa must build them out of her building blocks! The building blocks Pippa has have a uniform width and depth, but come in different heights h. Zara's description for Pippa will consist of a list of buildings, each with a target height b.

Pippa must then use some (not necessarily all) of her blocks to stack together such that the sum of the heights of the blocks in each stack corresponds to the height of the building in the skyline. Since Pippa prefers building instead of solving puzzles she wants you to determine how she must stack the blocks so that she must only do the stacking!

Input

The input consists of a single test case. The first line of this test case contains two integers N, S ($1 \leq N \leq 15$ and $1 \leq S \leq 15$), where N is the number of blocks Pippa has and S is the number of buildings in the skyline Zara made.

The next line contains N integers ($1 \leq h_i \leq 10^9$) representing the heights of each block. The last line contains S integers ($1 \leq b_i \leq 10^9$) representing the height of each building.

Output

If it is possible for Pippa to build Zara's skyline then output S lines. On each line output a single number s_i representing the number of blocks needed to build building i where i corresponds to the i^{th} building listed in the input. This should be followed (on the same line) by s_i numbers j representing the blocks of the input used in building i, where j represents the j^{th} block appearing the input.

If no combination of the blocks can build the desired skyline then output `-1`.

Sample Input 1	Sample Output 1
4 3 3 3 2 1 3 3 3	1 1 1 2 2 3 4

Sample Input 2	Sample Output 2
4 2 3 3 2 2 6 3	-1

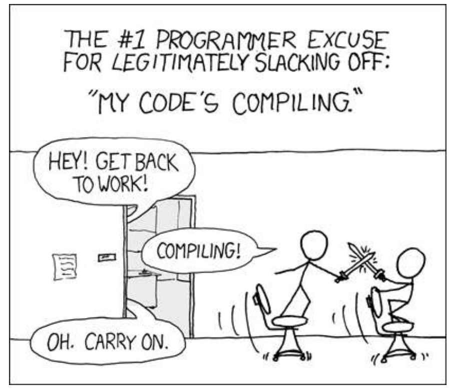
Sample Input 3

```
7 3
5 4 3 6 1 2 2
4 11 4
```

Sample Output 3

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

Problem H
Annoyed Coworkers



A picture of you, not working. Source: XKCD 303

It's another day in the office, and you're a mastermind of not doing any work yourself. Instead, you'll go to your coworkers for "help," but secretly have them do all the work.

You've determined that the more one of your coworkers helps you, the more annoyed they become. You've also been able to determine how much more annoyed a coworker gets everytime you ask them for help. At the beginning of the day, a coworker is initially a annoyed at you. That's their annoyance level. Everytime you ask them for help though, they become d more annoyed at you – their annoyance level a increases by a constant amount d so that $a = a + d$.

You want to complete a project of h tasks solely with "help" from your coworkers, but you need to be careful not to annoy any of them too much.

What's the best you can do?

Input

The first line contains 2 integers h and c, where h ($1 \leq h \leq 100\,000$) is the number of times you have to ask for help to complete the project, and c ($1 \leq c \leq 100\,000$) denotes the number of coworkers you have.

Each of the following c lines contains two positive integers a and d, representing a coworker whose initial annoyance level is a and who is getting more annoyed at you by an increase of d every time you ask them for help ($1 \leq a, d \leq 10^9$).

Output

Output a single number, which is the maximum annoyance level any coworker has at you provided you use an optimal strategy to minimize this level. (In other words, of all possible strategies, choose one that minimizes the annoyance level of the worker or workers who are most annoyed at you at the end.)

Sample Input 1 Explanation

You have 4 coworkers and you need to ask for help 4 times. Initially, their annoyance levels are $a_1 = 1, a_2 = 2, a_3 = 3, a_4 = 4$, the increases are $d_1 = 2, d_2 = 3, d_3 = 4, d_4 = 5$. One optimal solution is to ask for help twice from coworker 1, once from coworker 2, and once from coworker 3, in which case the final annoyance levels are: $a_1 = 1 + 2 \times 2 = 5, a_2 = 2 + 3 = 5, a_3 = 3 + 4 = 7, a_4 = 4$. The coworker that is most annoyed at you is coworker 3, whose annoyance level at you is 7. Or, you could ask coworker 1 for help 3 times and coworker 2 once, leaving you with $a_1 = 1 + 3 \times 2 = 7, a_2 = 2 + 3 = 5, a_3 = 3, a_4 = 4$. Both strategies yield the same minimal maximum amount.

Sample Input 1

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

Sample Output 1

```
7
```

Sample Input 2

```
3 2
1 1000
1000 1
```

Sample Output 2

```
1002
```

Sample Input 3

```
5 2
1 1
2 2
```

Sample Output 3

```
5
```

Problem I
Reconstructing Tape Art

Raelynn is trying to learn the newest craze in modern art: Tape Art! This wonderful new type of art is created by taking a wooden plank and pieces of tape of different colors. Each artwork is constructed by taking multiple pieces of tape and placing them on the plank. For each color that appears in the artwork, only a single piece of tape is used. Tapes can be placed on top of each other in which case the tape placed last obscures already placed pieces of tape with which it overlaps.

Source: pixabay

Raelynn has decided the best way to learn is by copying Sheila, the world famed tape artist. Unfortunately those pieces of art are under lock and key and Raelynn can see only pictures of these marvels. Since Raelynn is having trouble reverse engineering the artwork from the picture, she has hired you to create a set of instructions with which she can copy the art.

Since Raelynn is spoiled by the ease of IKEA catalogs she requires instructions to be given in the following format: there should only be one instruction per color of tape and instructions should be given in the order they should be executed. Each instruction must consist of three numbers: l r c where $[l, r]$ represents the inclusive range on which the tape should be placed and c represents the color of the tape piece. Planks are divided into n 1-inch sections numbered 1 through n.

Input

The input consists of a single test case. The first line of this test case contains one integer n ($1 \leq n \leq 10^5$), where n is the length of the tape art in inches. The next line contains n integers c_i ($1 \leq c_i \leq n$) representing the color of one inch of the plank.

Output

Output *any* set of instructions that, when executed, will result in the tape art given by the input. Output the string "`IMPOSSIBLE`" if the piece of tape art cannot be reconstructed using only one piece of each color (Sheila must have broken the rules to make it or this piece is a forgery).

Sample Input 1	Sample Output 1
6 1 2 3 3 2 1	3 1 6 1 2 5 2 3 4 3

Sample Input 2	Sample Output 2
4 1 2 1 2	IMPOSSIBLE

Sample Input 3	Sample Output 3
10 3 3 3 5 4 2 4 4 5 1	5 4 9 5 5 8 4 10 10 1 6 6 2 1 3 3

Problem J
Jack The Lumberjack

Source: Conifer Forest by Pexels

Jack the Lumberjack used to love chopping down trees. Jack is getting older and is becoming tired of this activity he used to love. He thinks of an idea, 'The Big One' and fantasizes about going out into the forest one last time to harvest as many trees as possible.

Jack calls the forest administrator for his local evergreen forest. The forest administrator tells him about how the population changes for each species of tree. For each species k, S_k trees are planted in year B_k. For the next Y_k years, the population increases by I_k per year. After Y_k years, it will decrease by the same amount I_k per year, until possibly dying out.

Armed with this information, Jack wants to figure out the maximum amount of trees that could be harvested at once from now until the future. If he is no longer around to do it, his descendants will be!

Assume all populations change instantly and at the same time, once per year. Jack would assess each population's size after the yearly change occurred.

Input

The input contains a single test case. The first line contains an integer N ($1 \leq N \leq 1\,000$) representing the number of tree species in the forest.

Each of the following N lines represents a single tree species population. Each of these population lines contains 4 integer numbers `Y I S B` ($0 \leq Y \leq 1\,000\,000$, $0 \leq I \leq 1\,000$, $0 \leq S \leq 1\,000\,000$, $0 \leq B \leq 1\,000\,000$). where S is the starting population size, B the year in which the population is planted, Y the number of years during which the population increases each year by I before it decreases by I until it (possibly) dies out.

Output

Print the maximum amount of trees that can be harvested in any single year.

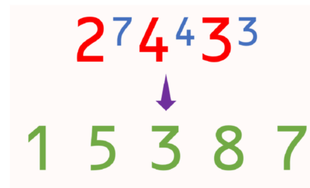
Sample Input 1	Sample Output 1
1 10 10 0 5	100

Sample Input 2	Sample Output 2
3 5 10 0 4 10 10 10 1 5 5 0 0	145

This page is intentionally left blank.

Problem K
Random Digital Exponentiation

In a random digital exponentiation (RDE) cipher, each digit of a number a is raised to a certain power which is determined by a randomly chosen encryption key. The encrypted number is the number that results when each digit is replaced with its power at the corresponding place value.



For instance, for $a = 243$ and an encryption key of `7 4 3`, the encrypted value b would be $b = 2^7 \times 10^2 + 4^4 \times 10 + 3^3 = 15\,387$.

Given a plaintext value a and an encrypted value b, find the encryption key that was used!

Input

The input will contain a single test case containing two numbers a and b ($0 \le a \le b \le 2\,147\,483\,647$).

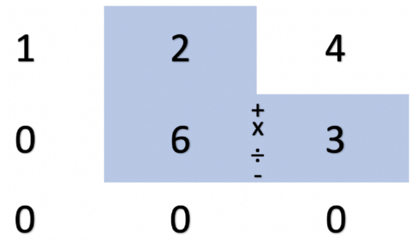
Output

Print the encryption key, which is the list of exponents to which each digit in a was raised, separated by spaces, starting from the leftmost digit's exponent. You are guaranteed that the encryption key exists and is unique.

Sample Input 1	Sample Output 1
`243 15387`	`7 4 3`

This page is intentionally left blank.

Problem L
Counting Codes



Graphic by Henry Wang.

You're one of the king's spies sent on a secret mission to retrieve an item of incredible value, an ancient scroll from the throne room. Legend has it, the scroll contains the answer to the P versus NP problem. When you finally reach the throne room, you realize there is a code the guards enter every day while observing them. As a spy, you've started to notice a few rules for each guard's code:

1. each code is a matrix consisting of nonzero decimal digits (integers from 1 to 9) with m rows and n columns
2. no digit repeats within any row of the code
3. for each digit l in the code, except for those in the topmost row and rightmost column, let u be the digit above it and let r be the digit to its right in the code matrix. Then one of the following must be true:
   - u is the product of l and r
   - u is the sum of l and r
   - u is the difference of l and r or r and l
   - u is the quotient of l and r or r and l

On day 999, you've noticed a guard has seem to walked off while entering his code. Some digits have been omitted, but after careful consideration you think you can crack the code. Digits that have been omitted are represented with a 0. How many complete codes are possible, given the guard's partial code?

Input

A test case starts with a line containing two numbers m ($3 \leq m \leq 6$) and n ($3 \leq n \leq 6$), which is the number of rows and number of columns of the grid. The following m lines contain n integers from 0 to 9, separated by spaces. 0 indicates an unknown value that you can supply, and there will be at most $\lfloor \frac{m*n}{2} \rfloor$ unknown values.

You can assume the guard has followed the rules with the partial code (i.e. no repeated digits appear in any row in the input, and any three pairs of non-zero digits that form an L have the property described above).

Output

For each test case, print the number of complete codes you can find.

Sample Input 1

```
3 3
1 2 4
0 3 6
4 0 3
```

Sample Output 1

```
2
```

Sample Input 2

```
3 4
2 3 0 7
0 0 2 1
0 0 3 0
```

Sample Output 2

```
37
```

Sample Input 3

```
3 4
1 3 0 7
2 0 0 1
0 0 9 0
```

Sample Output 3

```
14
```

Problem M
Broken Calculator

Source: Pixabay/accountant-accounting-adviser

Working on math homework late one night, you realized your calculator is broken. When it performs "addition" it adds the two numbers entered, then subtracts the result from the previous operation. When it performs "subtraction" it subtracts the two numbers entered, then multiplies the result by the previous operation's result. When it performs "multiplication" it squares its answer after multiplying the two numbers entered. When it performs "division" it divides the first number by 2 if it is even, otherwise it adds 1 to the first number and divides it by 2.

You also notice that when the calculator is turned on the previous operation's result is set to 1. You realize that doing math homework this way makes it more fun and want to write a program to perform calculations like your broken calculator.

You spoke to your teacher about your broken calculator and they gave you the guarantee that even with this weird behavior your calculator will never have to compute numbers larger than one quintillion (10^{18}) on your upcoming homework assignments.

Input

The input will begin with an integer n ($1 \leq n \leq 1\,000$), the number of commands you want to run on your calculator. Following the first line, each line will have an integer number a, an operator op, and a number b, separated by a single space, which denote the first operand, the operation, and the second operand, respectively. The operator will be one of $+$, $-$, $\star$, or $/$. The operands will be in the range $0 \leq a, b \leq 100\,000$.

Output

Print the answer the calculator will show after running the command on each line, assuming it is turned on before the first line is entered and stays on for all subsequent lines of input.

Sample Input 1

```
5
4 * 5
2 + 5
3 - 1
20 / 3
13 / 24
```

Sample Output 1

```
400
-393
-786
10
7
```

This page is intentionally left blank.