# School Personal Contest #3 (Winter Computer School 2010/11)<br>-<br>Codeforces Beta Round #45 (ACM-ICPC Rules)

## A. Rock-paper-scissors

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Uncle Fyodor, Matroskin the Cat and Sharic the Dog live their simple but happy lives in Prostokvashino. Sometimes they receive parcels from Uncle Fyodor's parents and sometimes from anonymous benefactors, in which case it is hard to determine to which one of them the package has been sent. A photographic rifle is obviously for Sharic who loves hunting and fish is for Matroskin, but for whom was a new video game console meant? Every one of the three friends claimed that the present is for him and nearly quarreled. Uncle Fyodor had an idea how to solve the problem justly: they should suppose that the console was sent to all three of them and play it in turns. Everybody got relieved but then yet another burning problem popped up — who will play first? This time Matroskin came up with a brilliant solution, suggesting the most fair way to find it out: play rock-paper-scissors together. The rules of the game are very simple. On the count of three every player shows a combination with his hand (or paw). The combination corresponds to one of three things: a rock, scissors or paper. Some of the gestures win over some other ones according to well-known rules: the rock breaks the scissors, the scissors cut the paper, and the paper gets wrapped over the stone. Usually there are two players. Yet there are three friends, that's why they decided to choose the winner like that: If someone shows the gesture that wins over the other two players, then that player wins. Otherwise, another game round is required. Write a program that will determine the winner by the gestures they have shown.

### Input

The first input line contains the name of the gesture that Uncle Fyodor showed, the second line shows which gesture Matroskin showed and the third line shows Sharic's gesture.

### Output

Print "F" (without quotes) if Uncle Fyodor wins. Print "M" if Matroskin wins and "S" if Sharic wins. If it is impossible to find the winner, print "?".

**Sample test(s)**

| input |
| --- |
| rock<br>rock<br>rock |
| output |
| ? |

| input |
| --- |
| paper<br>rock<br>rock |
| output |
| F |

| input |
| --- |
| scissors<br>rock<br>rock |
| output |
| ? |

| input |
| --- |
| scissors<br>paper<br>rock |
| output |
| ? |

# B. Land Lot

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Vasya has a beautiful garden where wonderful fruit trees grow and yield fantastic harvest every year. But lately thieves started to sneak into the garden at nights and steal the fruit too often. Vasya can't spend the nights in the garden and guard the fruit because there's no house in the garden! Vasya had been saving in for some time and finally he decided to build the house. The rest is simple: he should choose in which part of the garden to build the house. In the evening he sat at his table and drew the garden's plan. On the plan the garden is represented as a rectangular checkered field $n \times m$ in size divided into squares whose side length is 1. In some squares Vasya marked the trees growing there (one shouldn't plant the trees too close to each other that's why one square contains no more than one tree). Vasya wants to find a rectangular land lot $a \times b$ squares in size to build a house on, at that the land lot border should go along the lines of the grid that separates the squares. All the trees that grow on the building lot will have to be chopped off. Vasya loves his garden very much, so help him choose the building land lot location so that the number of chopped trees would be as little as possible.

## Input

The first line contains two integers $n$ and $m$ ($1 \le n, m \le 50$) which represent the garden location. The next $n$ lines contain $m$ numbers 0 or 1, which describe the garden on the scheme. The zero means that a tree doesn't grow on this square and the 1 means that there is a growing tree. The last line contains two integers $a$ and $b$ ($1 \le a, b \le 50$). Note that Vasya can choose for building an $a \times b$ rectangle as well a $b \times a$ one, i.e. the side of the lot with the length of $a$ can be located as parallel to the garden side with the length of $n$, as well as parallel to the garden side with the length of $m$.

## Output

Print the minimum number of trees that needs to be chopped off to select a land lot $a \times b$ in size to build a house on. It is guaranteed that at least one lot location can always be found, i. e. either $a \le n$ and $b \le m$, or $a \le m$ и $b \le n$.

**Sample test(s)**

input

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

output

```
0
```

input

```
4 5
0 0 1 0 1
0 1 1 1 0
1 0 1 0 1
1 1 1 1 1
2 3
```

output

```
2
```

## Note

In the second example the upper left square is (1,1) and the lower right is (3,2).

# C. The Race

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Every year a race takes place on the motorway between cities A and B. This year Vanya decided to take part in the race and drive his own car that has been around and bears its own noble name — The Huff-puffer.

So, Vasya leaves city A on the Huff-puffer, besides, at the very beginning he fills the petrol tank with $\alpha$ liters of petrol ($\alpha \geq 10$ is Vanya's favorite number, it is not necessarily integer). Petrol stations are located on the motorway at an interval of $100$ kilometers, i.e. the first station is located $100$ kilometers away from the city A, the second one is $200$ kilometers away from the city A, the third one is $300$ kilometers away from the city A and so on. The Huff-puffer spends $10$ liters of petrol every $100$ kilometers.

Vanya checks the petrol tank every time he passes by a petrol station. If the petrol left in the tank is not enough to get to the next station, Vanya fills the tank with $\alpha$ liters of petrol. Otherwise, he doesn't stop at the station and drives on.

For example, if $\alpha = 43.21$, then the car will be fuelled up for the first time at the station number $4$, when there'll be $3.21$ petrol liters left. After the fuelling up the car will have $46.42$ liters. Then Vanya stops at the station number $8$ and ends up with $6.42 + 43.21 = 49.63$ liters. The next stop is at the station number $12$, $9.63 + 43.21 = 52.84$. The next stop is at the station number $17$ and so on.

You won't believe this but the Huff-puffer has been leading in the race! Perhaps it is due to unexpected snow. Perhaps it is due to video cameras that have been installed along the motorway which register speed limit breaking. Perhaps it is due to the fact that Vanya threatened to junk the Huff-puffer unless the car wins. Whatever the reason is, the Huff-puffer is leading, and jealous people together with other contestants wrack their brains trying to think of a way to stop that outrage.

One way to do this is to mine the next petrol station where Vanya will stop. Your task is to calculate at which station this will happen and warn Vanya. You don't know the $\alpha$ number, however, you are given the succession of the numbers of the stations where Vanya has stopped. Find the number of the station where the next stop will be.

## Input

The first line contains an integer $n$ ($1 \leq n \leq 1000$) which represents the number of petrol stations where Vanya has stopped. The next line has $n$ space-separated integers which represent the numbers of the stations. The numbers are positive and do not exceed $10^6$, they are given in the increasing order. No two numbers in the succession match. It is guaranteed that there exists at least one number $\alpha \geq 10$, to which such a succession of stops corresponds.

## Output

Print in the first line "unique" (without quotes) if the answer can be determined uniquely. In the second line print the number of the station where the next stop will take place. If the answer is not unique, print in the first line "not unique".

**Sample test(s)**

| input |
| --- |
| 3<br>1 2 4 |
| output |
| unique<br>5 |

| input |
| --- |
| 2<br>1 2 |
| output |
| not unique |

## Note

In the second example the answer is not unique. For example, if $\alpha = 10$, we'll have such a sequence as $1, 2, 3$, and if $\alpha = 14$, the sequence will be $1, 2, 4$.

# D. Permutations

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

A permutation is a sequence of integers from $1$ to $n$ of length $n$ containing each number exactly once. For example, $(1)$, $(4, 3, 5, 1, 2)$, $(3, 2, 1)$ are permutations, and $(1, 1)$, $(4, 3, 1)$, $(2, 3, 4)$ are not.

There are many tasks on permutations. Today you are going to solve one of them. Let's imagine that somebody took several permutations (perhaps, with a different number of elements), wrote them down consecutively as one array and then shuffled the resulting array. The task is to restore the initial permutations if it is possible.

## Input

The first line contains an integer $n$ ($1 \le n \le 10^5$). The next line contains the mixed array of $n$ integers, divided with a single space. The numbers in the array are from $1$ to $10^5$.

## Output

If this array can be split into several permutations so that every element of the array belongs to exactly one permutation, print in the first line the number of permutations. The second line should contain $n$ numbers, corresponding to the elements of the given array. If the $i$-th element belongs to the first permutation, the $i$-th number should be $1$, if it belongs to the second one, then its number should be $2$ and so on. The order of the permutations' numbering is free.

If several solutions are possible, print any one of them. If there's no solution, print in the first line $-1$.

**Sample test(s)**

| input |
| --- |
| 9<br>1 2 3 1 2 1 4 2 5 |
| output |
| 3<br>3 1 2 1 2 2 2 3 2 |

| input |
| --- |
| 4<br>4 3 2 1 |
| output |
| 1<br>1 1 1 1 |

| input |
| --- |
| 4<br>1 2 2 3 |
| output |
| -1 |

## Note

In the first sample test the array is split into three permutations: $(2, 1)$, $(3, 2, 1, 4, 5)$, $(1, 2)$. The first permutation is formed by the second and the fourth elements of the array, the second one — by the third, the fifth, the sixth, the seventh and the ninth elements, the third one — by the first and the eigth elements. Clearly, there are other splitting variants possible.

# E. Ivan the Fool VS Gorynych the Dragon

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Once upon a time in a kingdom far, far away… Okay, let's start at the point where Ivan the Fool met Gorynych the Dragon. Ivan took out his magic sword and the battle began. First Gorynych had $h$ heads and $t$ tails. With each strike of the sword Ivan can either cut off several heads (from $1$ to $n$, but not more than Gorynych has at the moment), or several tails (from $1$ to $m$, but not more than Gorynych has at the moment). At the same time, horrible though it seems, Gorynych the Dragon can also grow new heads and tails. And the number of growing heads and tails is determined uniquely by the number of heads or tails cut by the current strike. When the total number of heads and tails exceeds $R$, Gorynych the Dragon strikes its final blow and destroys Ivan the Fool. That's why Ivan aims to cut off all the dragon's heads and tails as quickly as possible and win. The events can also develop in a third way: neither of the opponents can win over the other one and they will continue fighting forever.

The tale goes like this; easy to say, hard to do. Your task is to write a program that will determine the battle's outcome. Consider that Ivan strikes consecutively. After each blow Gorynych grows a number of new heads and tails depending on the number of cut ones. Gorynych the Dragon is defeated if after the blow he loses all his heads and tails and can't grow new ones. Ivan fights in the optimal way (fools are lucky), i.e.

- if Ivan can win, he wins having struck the least number of blows;
- if it is impossible to defeat Gorynych, but is possible to resist him for an infinitely long period of time, then that's the strategy Ivan chooses;
- if Gorynych wins in any case, Ivan aims to resist him for as long as possible.

## Input

The first line contains three integers $h$, $t$ and $R$ ($0 \le h, t, R \le 200$, $0 < h + t \le R$) which represent the initial numbers of Gorynych's heads and tails and the largest total number of heads and tails with which Gorynych the Dragon does not yet attack. The next line contains integer $n$ ($1 \le n \le 200$). The next $n$ contain pairs of non-negative numbers "$h_i$ $t_i$" which represent the number of heads and the number of tails correspondingly, that will grow if Gorynych has $i$ heads ($1 \le i \le n$) cut. The next line contains an integer $m$ ($1 \le m \le 200$) and then — the description of Gorynych's behavior when his tails are cut off in the format identical to the one described above. All the numbers in the input file do not exceed $200$.

## Output

Print "Ivan" (without quotes) in the first line if Ivan wins, or "Zmey" (that means a dragon in Russian) if Gorynych the Dragon wins. In the second line print a single integer which represents the number of blows Ivan makes. If the battle will continue forever, print in the first line "Draw".

**Sample test(s)**

| input |
| --- |
| 2 2 4<br>2<br>1 0<br>0 1<br>3<br>0 1<br>0 1<br>0 0 |

| output |
| --- |
| Ivan<br>2 |

| input |
| --- |
| 2 2 4<br>1<br>0 1<br>1<br>1 0 |

| output |
| --- |
| Draw |

| input |
| --- |
| 2 2 5<br>1<br>1 1<br>1<br>3 0 |

| output |
| --- |
| Zmey<br>2 |

# F. Snow sellers

time limit per test: 10 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

The New Year celebrations in Berland last $n$ days. Only this year the winter is snowless, that's why the winter celebrations' organizers should buy artificial snow. There are $m$ snow selling companies in Berland. Every day the $i$-th company produces $w_i$ cubic meters of snow. Next day the snow thaws and the company has to produce $w_i$ cubic meters of snow again. During the celebration new year discounts are on, that's why the snow cost decreases every day. It is known that on the first day the total cost of all the snow produced by the $i$-th company is equal to $c_i$ bourles. Every day this total cost decreases by $a_i$ bourles, i.e. on the second day it is equal to $c_i$ - $a_i$,and on the third day — to $c_i$ - $2a_i$, and so on. It is known that for one company the cost of the snow produced by it does not get negative or equal to zero. You have to organize the snow purchase so as to buy every day exactly $W$ snow cubic meters. At that it is not necessary to buy from any company all the snow produced by it. If you buy $n_i$ cubic meters of snow ($0 \le n_i \le w_i$, the number $n_i$ is not necessarily integer!) from the $i$-th company at one of the days when the cost of its snow is equal to $s_i$, then its price will total to $\frac{n_i s_i}{w_i}$ bourles. During one day one can buy the snow from several companies. In different days one can buy the snow from different companies. It is required to make the purchases so as to spend as little money as possible. It is guaranteed that the snow produced by the companies will be enough.

## Input

The first line contains integers $n$, $m$ and $W$ ($1 \le n \le 100$, $1 \le m \le 500000$, $1 \le W \le 10^9$) which represent the number of days, the number of companies and the amount of snow that needs to be purchased on every one of the $n$ days. The second line contains $m$ integers $w_i$. The third line contains $m$ integers $c_i$. The fourth line contains $m$ integers $a_i$. All the numbers are strictly positive and do not exceed $10^9$. For all the $i$ the inequation $c_i$ - $(n$ - $1)a_i > 0$ holds true.

## Output

Print a single number — the answer to the given problem. Print the answer in the format with the decimal point (even if the answer is integer, it must contain the decimal point), without "e" and without leading zeroes. The answer should differ with the right one by no more than $10^{-9}$.

**Sample test(s)**

input

```
2 3 10
4 4 4
5 5 8
1 2 5
```

output

```
22.000000000000000
```

input

```
100 2 1000000000
999999998 999999999
1000000000 1000000000
1 1
```

output

```
99999995149.999995249999991
```

# G. Galaxy Union

time limit per test: 3 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

In a far away galaxy there are $n$ inhabited planets numbered with numbers from $1$ to $n$. One day the presidents of all the $n$ planets independently from each other came up with an idea of creating the Galaxy Union. Now they need to share this wonderful idea with their galaxymates, that's why each president is busy working out a project of negotiating with the other presidents.

For negotiations between some pairs of the planets there are bidirectional communication channels, each of which is characterized with "dial duration" $t_i$ which, as a rule, takes several hours and exceeds the call duration greatly. Overall the galaxy has $n$ communication channels and they unite all the planets into a uniform network. That means that it is possible to phone to any planet $v$ from any planet $u$, perhaps, using some transitional planets $v_1, v_2, ..., v_m$ via the existing channels between $u$ and $v_1$, $v_1$ and $v_2$, ..., $v_{m-1}$ and $v_m$, $v_m$ and $v$. At that the dial duration from $u$ to $v$ will be equal to the sum of dial durations of the used channels.

So, every president has to talk one by one to the presidents of all the rest $n - 1$ planets. At that the negotiations take place strictly consecutively, and until the negotiations with a planet stop, the dial to another one does not begin. As the matter is urgent, from the different ways to call the needed planet every time the quickest one is chosen. Little time is needed to assure another president on the importance of the Galaxy Union, that's why the duration of the negotiations with each planet can be considered equal to the dial duration time for those planets. As the presidents know nothing about each other's plans, they do not take into consideration the possibility that, for example, the sought president may call himself or already know about the founding of the Galaxy Union from other sources.

The governments of all the $n$ planets asked you to work out the negotiation plans. First you are to find out for every president how much time his supposed negotiations will take.

## Input

The first line contains an integer $n$ ($3 \le n \le 200000$) which represents the number of planets in the Galaxy and the number of communication channels equal to it. The next $n$ lines contain three integers each $a_i$, $b_i$ and $t_i$ ($1 \le a_i, b_i \le n, a_i \ne b_i, 1 \le t_i \le 10^3$) that represent the numbers of planet joined by a communication channel and its "dial duration". There can be no more than one communication channel between a pair of planets.

## Output

In the first line output $n$ integers — the durations of the supposed negotiations for each president. Separate the numbers by spaces.

**Sample test(s)**

input
```
3
1 2 3
2 3 2
1 3 1
```
output
```
4 5 3
```

input
```
3
1 2 3
2 3 2
1 3 5
```
output
```
8 5 7
```

input
```
4
1 2 3
2 3 2
3 4 1
4 1 4
```
output
```
12 8 8 8
```

# H. Black and White

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

According to the legends the king of Berland Berl I was noted for his love of beauty and order. One day he ordered to tile the palace hall's floor where balls and receptions used to take place with black and white tiles according to a regular geometrical pattern invented by him. However, as is after the case, due to low financing there were only $a$ black and $b$ white tiles delivered to the palace. The other $c$ tiles were black and white (see the picture).

The initial plan failed! Having learned of that, the king gave a new command: tile the floor with the available tiles so that no black side of a tile touched a white one. The tiles are squares of one size $1 \times 1$, every black and white tile can be rotated in one of the four ways.

The court programmer was given the task to work out the plan of tiling and he coped with the task and didn't suffer the consequences of disobedience. And can you cope with it?

## Input

The first line contains given integers $n$ and $m$ ($1 \le n, m \le 100$) which represent the sizes of the rectangle that needs to be tiled. The next line contains non-negative numbers $a$, $b$ and $c$, $a + b + c = nm$, $c \ge m$.

## Output

Print $2n$ lines containing $2m$ characters each — the tiling scheme. Every tile is represented by a square $2 \times 2$ in the following manner (the order corresponds to the order of the picture above):

```
##    ..    \#    ./    \.    #/
##    ..    .\    /#    #\    /.
```

If multiple solutions exist, output any.

**Sample test(s)**

| input |
| --- |
| 2 2<br>0 0 4 |
| output |
| \../<br>#\/#<br>\##/<br>.\/. |

| input |
| --- |
| 2 3<br>1 2 3 |
| output |
| ###/\#<br>##/..\<br>#/....<br>/..... |