Codeforces Round #115

A. Robot Bicorn Attack

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

Vasya plays Robot Bicorn Attack.

The game consists of three rounds. For each one a non-negative integer amount of points is given. The result of the game is the sum of obtained points. Vasya has already played three rounds and wrote obtained points one by one (without leading zeros) into the string S. Vasya decided to brag about his achievement to the friends. However, he has forgotten how many points he got for each round. The only thing he remembers is the string S.

Help Vasya to find out what is the maximum amount of points he could get. Take into account that Vasya played Robot Bicorn Attack for the first time, so he could not get more than 1000000 (10^6) points for one round.

Input
The only line of input contains non-empty string S obtained by Vasya. The string consists of digits only. The string length does not exceed 30 characters.

Output
Print the only number — the maximum amount of points Vasya could get. If Vasya is wrong and the string could not be obtained according to the rules then output number -1.

Examples

input
1234
output
37

input
9000
output
90

input
0009
output
-1

Note
In the first example the string must be split into numbers 1, 2 and 34.

In the second example the string must be split into numbers 90, 0 and 0.

In the third example the string is incorrect, because after splitting the string into 3 numbers number 00 or 09 will be obtained, but numbers cannot have leading zeroes.

B. Plane of Tanks: Pro

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

Vasya has been playing Plane of Tanks with his friends the whole year. Now it is time to divide the participants into several categories depending on their results.

A player is given a non-negative integer number of points in each round of the Plane of Tanks. Vasya wrote results for each round of the last year. He has n records in total.

In order to determine a player's category consider the best result obtained by the player and the best results of other players. The player belongs to category:

- "noob" — if **more than 50% of players have better results**;
- "random" — if his result is not worse than the result that 50% of players have, but more than 20% of players have better results;
- "average" — if his result is not worse than the result that 80% of players have, but more than 10% of players have better results;
- "hardcore" — if his result is not worse than the result that 90% of players have, but more than 1% of players have better results;
- "pro" — if his result is not worse than the result that 99% of players have.

When the percentage is calculated the player himself is taken into account. That means that if two players played the game and the first one gained 100 points and the second one 1000 points, then the first player's result is not worse than the result that 50% of players have, and the second one is not worse than the result that 100% of players have.

Vasya gave you the last year Plane of Tanks results. Help Vasya determine each player's category.

Input

The first line contains the only integer number n ($1 \le n \le 1000$) — a number of records with the players' results.

Each of the next n lines contains a player's name and the amount of points, obtained by the player for the round, separated with a space. The name contains not less than 1 and no more than 10 characters. The name consists of lowercase Latin letters only. It is guaranteed that any two different players have different names. The amount of points, obtained by the player for the round, is a non-negative integer number and does not exceed 1000.

Output

Print on the first line the number m — the number of players, who participated in one round at least.

Each one of the next m lines should contain a player name and a category he belongs to, separated with space. Category can be one of the following: "noob", "random", "average", "hardcore" or "pro" (without quotes). The name of each player should be printed only once. Player names with respective categories can be printed in an arbitrary order.

Examples

input
5 vasya 100 vasya 200 artem 100 kolya 200 igor 250
output
4 artem noob igor pro kolya random vasya random

input
3 vasya 200 kolya 1000 vasya 1000
output
2 kolya pro vasya pro

Note

In the first example the best result, obtained by `artem` is not worse than the result that 25% of players have (his own result), so he belongs to category "`noob`". `vasya` and `kolya` have best results not worse than the results that 75% players have (both of them and `artem`), so they belong to category "`random`". `igor` has best result not worse than the result that 100% of players have (all other players and himself), so he belongs to category "`pro`".

In the second example both players have the same amount of points, so they have results not worse than 100% players have, so they belong to category "`pro`".

C. Geometry Horse

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

Vasya plays the Geometry Horse.

The game goal is to destroy geometric figures of the game world. A certain number of points is given for destroying each figure depending on the figure type and the current factor value.

There are n types of geometric figures. The number of figures of type k_i and figure cost c_i is known for each figure type. A player gets $c_i \cdot f$ points for destroying one figure of type i, where f is the current factor. The factor value can be an integer number from 1 to $t + 1$, inclusive. At the beginning of the game the factor value is equal to 1. The factor is set to $i + 1$ after destruction of p_i $(1 \le i \le t)$ figures, so the $(p_i + 1)$-th figure to be destroyed is considered with factor equal to $i + 1$.

Your task is to determine the maximum number of points Vasya can get after he destroys all figures. Take into account that Vasya is so tough that he can destroy figures in any order chosen by him.

Input

The first line contains the only integer number n $(1 \le n \le 100)$ — the number of figure types.

Each of the following n lines contains two integer numbers k_i and c_i $(1 \le k_i \le 10^9, 0 \le c_i \le 1000)$, separated with space — the number of figures of the i-th type and the cost of one i-type figure, correspondingly.

The next line contains the only integer number t $(1 \le t \le 100)$ — the number that describe the factor's changes.

The next line contains t integer numbers p_i $(1 \le p_1 < p_2 < ... < p_t \le 10^{12})$, separated with spaces.

Please, do not use the `%lld` specificator to read or write 64-bit integers in C++. It is preferred to use `cin`, `cout` streams or the `%I64d` specificator.

Output

Print the only number — the maximum number of points Vasya can get.

Examples

input
1 5 10 2 3 6
output
70

input
2 3 8 5 10 1 20
output
74

Note

In the first example Vasya destroys three figures first and gets $3 \cdot 1 \cdot 10 = 30$ points. Then the factor will become equal to 2 and after destroying the last two figures Vasya will get $2 \cdot 2 \cdot 10 = 40$ points. As a result Vasya will get 70 points.

In the second example all 8 figures will be destroyed with factor 1, so Vasya will get $(3 \cdot 8 + 5 \cdot 10) \cdot 1 = 74$ points.

D. Plane of Tanks: Duel

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

Vasya plays the Plane of Tanks.

Tanks are described with the following attributes:

- the number of hit points;
- the interval between two gun shots (the time required to recharge the gun);
- the probability that the gun shot will not pierce armor of the enemy tank;
- the damage to the enemy's tank.

The gun damage is described with a segment $[l, r]$, where l and r are integer numbers. The potential gun damage x is chosen with equal probability among all integer numbers of the segment $[l, r]$. If the shot pierces the armor of an enemy's tank then the enemy loses x hit points. If the number of hit points becomes non-positive then the enemy tank is considered destroyed.

It is possible that the shot does not pierce the armor of a tank. In this case the number of hit points doesn't change. The probability that the armor will not be pierced is considered as the shooting tank attribute and does not depend on players' behavior.

The victory is near and there is only one enemy tank left. Vasya is ready for the battle — one more battle between the Good and the Evil is inevitable! Two enemies saw each other and each of them fired a shot at the same moment... The last battle has begun! Help Vasya to determine what is the probability that he will win the battle by destroying the enemy tank?

If both tanks are destroyed (after simultaneous shots), then Vasya is considered a winner. You can assume that each player fires a shot just after the gun recharge and each tank has infinite number of ammo.

Input

The first line contains five integer numbers separated with spaces describing Vasya's tank: the number of hit points hp $(10 \le hp \le 200)$, the interval between two shots dt $(1 \le dt \le 30)$, gun damage segment l and r $(10 \le l \le r \le 100)$, the probability that the enemy's tank armor will not be pierced p $(0 \le p \le 100)$ (percents).

The second line describes the tank of Vasya's enemy in the same format.

Output

Print the only number with absolute or relative error no more than 10^{-4} — probability of Vasya's victory.

Examples

input
100 3 50 50 0 100 3 50 50 0
output
1.000000

input
100 3 50 50 0 100 2 48 50 0
output
0.888889

input
100 3 50 50 0 100 1 50 50 50
output
0.500000

Note

In the first example both tanks are destroyed at once after the second shot. The probability of destroying the enemy tank is 1.

In the second example Vasya's enemy tank fires the second shot before Vasya's tank, but has no time for the third shot. In order to destroy Vasya's tank it is necessary to fire two shots with damage 50. The probability of that event is $(\frac{1}{3})^2 = \frac{1}{9}$. Otherwise, Vasya wins.

In the third example Vasya's enemy tank fires three shots with probability of armor piercing 0.5. In order to destroy Vasya's tank it is necessary that at least 2 of 3 shots pierce the armor of Vasya's tank. The probability of this event is 0.5.

E. Power Defence

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

Vasya plays the Power Defence.

He must pass the last level of the game. In order to do this he must kill the Main Villain, who moves in a straight line at speed 1 meter per second from the point $(-\infty, 0)$ to the point $(+\infty, 0)$ of the game world. In the points $(x, 1)$ and $(x, -1)$, where x is an integer number, Vasya can build towers of three types: fire-tower, electric-tower or freezing-tower. However, it is not allowed to build two towers at the same point. Towers of each type have a certain action radius and the value of damage per second (except freezing-tower). If at some point the Main Villain is in the range of action of k freezing towers then his speed is decreased by $k+1$ times.

The allowed number of towers of each type is known. It is necessary to determine the maximum possible damage we can inflict on the Main Villain.

All distances in the problem are given in meters. The size of the Main Villain and the towers are so small, that they can be considered as points on the plane. The Main Villain is in the action radius of a tower if the distance between him and tower is less than or equal to the action radius of the tower.

Input

The first line contains three integer numbers nf, ne and ns — the maximum number of fire-towers, electric-towers and freezing-towers that can be built $(0 \le nf, ne, ns \le 20, 1 \le nf + ne + ns \le 20)$. The numbers are separated with single spaces.

The second line contains three integer numbers rf, re and rs $(1 \le rf, re, rs \le 1000)$ — the action radii of fire-towers, electric-towers and freezing-towers. The numbers are separated with single spaces.

The third line contains two integer numbers df and de $(1 \le df, de \le 1000)$ — the damage a fire-tower and an electronic-tower can inflict on the Main Villain per second (in the case when the Main Villain is in the action radius of the tower). The numbers are separated with single space.

Output

Print the only real number — the maximum possible damage to the Main Villain with absolute or relative error not more than 10^{-6}.

Examples

input
1 0 0 10 10 10 100 100
output
1989.97487421

input
1 0 1 10 10 10 100 100
output
3979.94974843

Note

In the first sample we've got one fire-tower that always inflicts the same damage, independently of its position.

In the second sample we've got another freezing-tower of the same action radius. If we build the two towers opposite each other, then the Main Villain's speed will be two times lower, whenever he enters the fire-tower's action radius. That means that the enemy will be inflicted with twice more damage.

F. Gnomes of Might and Magic

time limit per test: 8 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Vasya plays a popular game the Gnomes of Might and Magic.

In this game Vasya manages the kingdom of gnomes, consisting of several castles, connected by bidirectional roads. The kingdom road network has a special form. The kingdom has m main castles $a_1, a_2, \ldots, a_m$, which form the Good Path. This path consists of roads between the castles a_i, a_{i+1} $(1 \le i < m)$ as well as the road between a_m and a_1. There are no other roads between the castles of the Good Path.

In addition, for each pair of neighboring Good Path castles u and v there is exactly one Evil Shortcut — a path that goes along the roads leading from the first castle (u) to the second one (v) and not having any common vertexes with the Good Path except for the vertexes u and v. It is known that there are no other roads and castles in the kingdom there, that is, every road and every castle lies either on the Good Path or the Evil Shortcut (castles can lie in both of them). In addition, no two Evil Shortcuts have any common castles, different than the castles of the Good Path.

At the beginning of each week in the kingdom appears one very bad gnome who stands on one of the roads of the kingdom, and begins to rob the corovans going through this road. One road may accumulate multiple very bad gnomes. Vasya cares about his corovans, so sometimes he sends the Mission of Death from one castle to another.

Let's suggest that the Mission of Death should get from castle s to castle t. Then it will move from castle s to castle t, destroying all very bad gnomes, which are on the roads of the Mission's path. Vasya is so tough that his Mission of Death can destroy any number of gnomes on its way. However, Vasya is very kind, so he always chooses such path between castles s and t, following which he will destroy the smallest number of gnomes. If there are multiple such paths, then Vasya chooses the path that contains the smallest number of roads among them. If there are multiple such paths still, Vasya chooses the lexicographically minimal one among them.

Help Vasya to simulate the life of the kingdom in the Gnomes of Might and Magic game.

A path is a sequence of castles, such that each pair of the neighboring castles on the path is connected by a road. Also, path $x_1, x_2, \ldots, x_p$ is lexicographically less than path $y_1, y_2, \ldots, y_q$, if either $p < q$ and $x_1 = y_1, x_2 = y_2, \ldots, x_p = y_p$, or exists such number r $(r < p, r < q)$, that $x_1 = y_1, x_2 = y_2, \ldots, x_r = y_r$ and $x_{r+1} < y_{r+1}$.

Input

The first line contains two integers n and m $(3 \le m \le n \le 100000)$ — the number of castles in the kingdom, and the number of castles on the Good Path, respectively.

The second line contains m integers, which are numbers of Good Path castles (the castles are numbered from 1 to n) in the order of occurrence on the Path, starting with some castle. All Good Path castles are different.

Each of the following m lines describes an Evil Shortcut. First a line contains an integer k_i $(3 \le k_i \le 100000)$ — the number of castles on the corresponding Evil Shortcut (with the two castles which are on the Good Path), followed by a k_i integers — number of castles in the order of occurrence in the given Shortcut. All castles in one Evil Shortcut are different. It is guaranteed that the first and the last castles from the Shortcut are on the Good Path and the first castles in the Evil Shortcuts form the Good Path and are presented in the same order in which the Path was represented on the second line.

The next line contains an integer q $(1 \le q \le 100000)$ — the number of events in the life of the kingdom. Each of the following q lines describes a single event. An event is described by the symbol c_j and two numbers or castles s_j and t_j (the character and numbers of castles are separated by a single space). If the character of c_j is equal to "+" (a plus), it means that a very bad gnome (probably not the first one) has appeared on the road between castles s_j and t_j. If c_j equals "?" (a question), then Vasya sent a Mission of Death from castle s_j to castle t_j. It is guaranteed that for each request "+", the road between castles s_j and t_j exists. The events are given in chronological order, starting with the earliest one. Initially there are no very bad gnomes on the roads.

All numbers in all lines are separated by single spaces. It is guaranteed that all the given Evil Shortcuts and Good Path fit in the limitations given in the problem statement.

Output

For each query "?" print a single number on a single line — the number of very bad gnomes destroyed by the corresponding Mission of Death. Print the answers to queries in the chronological order.

Examples

input
6 3 1 2 3 3 1 4 2 3 2 5 3 3 3 6 1 10 + 1 2 + 4 2

```
+ 1 3
+ 2 3
? 1 2
+ 2 5
? 1 2
? 1 2
+ 1 2
? 1 2
```

output

```
0
1
0
1
```

Note

In the example after the first four requests there is only one path from castle 1 to castle 2, which does not contain roads with very bad gnomes: 1 ⟶ 6 ⟶ 3 ⟶ 5 ⟶ 2.

After a gnome stood on the road (2, 5), the next Mission of Death moves along path 1 ⟶ 2, and destroys the gnome, who was on the road (1, 2). The next Mission of Death follows the same path which is already free of gnomes.

After yet another gnome stood on the road (1, 2), the next Mission of Death goes on the path 1 ⟶ 2, and kills the gnome.