A. Wizards and Demonstration

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

Some country is populated by wizards. They want to organize a demonstration.

There are n people living in the city, x of them are the wizards who will surely go to the demonstration. Other city people (n - x people) do not support the wizards and aren't going to go to the demonstration. We know that the city administration will react only to the demonstration involving at least y percent of the city people. Having considered the matter, the wizards decided to create clone puppets which can substitute the city people on the demonstration.

So all in all, the demonstration will involve only the wizards and their puppets. The city administration cannot tell the difference between a puppet and a person, so, as they calculate the percentage, the administration will consider the city to be consisting of only n people and not containing any clone puppets.

Help the wizards and find the minimum number of clones to create to that the demonstration had no less than y percent of the city people.

Input

The first line contains three space-separated integers, n, x, y ($1 \le n, x, y \le 10^4, x \le n$) — the number of citizens in the city, the number of wizards and the percentage the administration needs, correspondingly.

Please note that y can exceed 100 percent, that is, the administration wants to see on a demonstration more people that actually live in the city ($> n$).

Output

Print a single integer — the answer to the problem, the minimum number of clones to create, so that the demonstration involved no less than y percent of n (the real total city population).

Sample test(s)

input
```
10 1 14
```
output
```
1
```

input
```
20 10 50
```
output
```
0
```

input
```
1000 352 146
```
output
```
1108
```

Note

In the first sample it is necessary that at least 14% of 10 people came to the demonstration. As the number of people should be integer, then at least two people should come. There is only one wizard living in the city and he is going to come. That isn't enough, so he needs to create one clone.

In the second sample 10 people should come to the demonstration. The city has 10 wizards. They will all come to the demonstration, so nobody has to create any clones.

B. Wizards and Minimal Spell

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

Let's dive into one of the most interesting areas of magic — writing spells. Learning this exciting but challenging science is very troublesome, so now you will not learn the magic words, but only get to know the basic rules of writing spells.

Each spell consists of several lines. The line, whose first non-space character is character "#" is an *amplifying line* and it is responsible for spell power. The remaining lines are common, and determine the effect of the spell.

You came across the text of some spell. Spell was too long, so you cannot understand its meaning. So you want to make it as short as possible without changing the meaning.

The only way to shorten a spell that you know is the removal of some spaces and line breaks. We know that when it comes to texts of spells, the spaces carry meaning only in the amplifying lines, so we should remove all spaces in other lines. Newlines also do not matter, unless any of the two separated lines is amplifying. Thus, if two consecutive lines are not amplifying, they need to be joined into one (i.e. we should concatenate the second line to the first one). Removing spaces in amplifying lines and concatenating the amplifying lines to anything is forbidden.

Note that empty lines must be processed just like all the others: they must be joined to the adjacent non-amplifying lines, or preserved in the output, if they are surrounded with amplifying lines on both sides (i.e. the line above it, if there is one, is amplifying, and the line below it, if there is one, is amplifying too).

For now those are the only instructions for removing unnecessary characters that you have to follow (oh yes, a newline is a character, too).

The input contains the text of the spell, which should be reduced. Remove the extra characters and print the result to the output.

Input

The input contains multiple lines. All characters in the lines have codes from 32 to 127 (inclusive). Please note that the lines may begin with or end with one or more spaces. The size of the input does not exceed 1048576 ($= 2^{20}$) bytes. Newlines are included in this size.

In the Windows operating system used on the testing computer, a newline is a sequence of characters with codes #13#10. It is guaranteed that after each line of input there is a newline. In particular, the input ends with a newline. Note that the newline is the end of the line, and not the beginning of the next one.

It is guaranteed that the input contains at least one character other than a newline.

It is recommended to organize the input-output line by line, in this case the newlines will be processed correctly by the language means.

Output

Print the text of the spell where all extra characters are deleted. Please note that each output line should be followed by a newline.

Please be careful: your answers will be validated by comparing them to the jury's answer byte-by-byte. So, all spaces and newlines matter.

Sample test(s)

input

```
   #   include <cstdio>

using namespace std;

int main     (   ){
puts("Hello # World"); #
#
}
```

output

```
   #   include <cstdio>
usingnamespacestd;intmain(){puts("Hello#World");#
#
}
```

input

```
#

#
```

output

```
#

#
```

Note

In the first sample the amplifying lines are lines 1 and 7. So, lines 2 to 6 are concatenated to each other, all spaces are deleted from them.

In the second sample the amplifying lines are lines 1 and 3. So, no lines are concatenated to each other.

C. Wizards and Trolleybuses

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

In some country live wizards. They love to ride trolleybuses.

A city in this country has a trolleybus depot with n trolleybuses. Every day the trolleybuses leave the depot, one by one and go to the final station. The final station is at a distance of d meters from the depot. We know for the i-th trolleybus that it leaves at the moment of time t_i seconds, can go at a speed of no greater than v_i meters per second, and accelerate with an acceleration no greater than a meters per second squared. A trolleybus can decelerate as quickly as you want (magic!). It can change its acceleration as fast as you want, as well. Note that the maximum acceleration is the same for all trolleys.

Despite the magic the trolleys are still powered by an electric circuit and cannot overtake each other (the wires are to blame, of course). If a trolleybus catches up with another one, they go together one right after the other until they arrive at the final station. Also, the drivers are driving so as to arrive at the final station as quickly as possible.

You, as head of the trolleybuses' fans' club, are to determine for each trolley the minimum time by which it can reach the final station. At the time of arrival at the destination station the trolleybus does not necessarily have zero speed. When a trolley is leaving the depot, its speed is considered equal to zero. From the point of view of physics, the trolleybuses can be considered as material points, and also we should ignore the impact on the speed of a trolley bus by everything, except for the acceleration and deceleration provided by the engine.

Input

The first input line contains three space-separated integers n, a, d $(1 \le n \le 10^5, 1 \le a, d \le 10^6)$ — the number of trolleybuses, their maximum acceleration and the distance from the depot to the final station, correspondingly.

Next n lines contain pairs of integers t_i v_i $(0 \le t_1 < t_2 ... < t_{n-1} < t_n \le 10^6, 1 \le v_i \le 10^6)$ — the time when the i-th trolleybus leaves the depot and its maximum speed, correspondingly. The numbers in the lines are separated by spaces.

Output

For each trolleybus print a single line the time it arrives to the final station. Print the times for the trolleybuses in the order in which the trolleybuses are given in the input. The answer will be accepted if the absolute or relative error doesn't exceed 10^{-4}.

Sample test(s)

input

```
3 10 10000
0 10
5 11
1000 1
```

output

```
1000.5000000000
1000.5000000000
11000.0500000000
```

input

```
1 2 26
28 29
```

output

```
33.0990195136
```

Note

In the first sample the second trolleybus will catch up with the first one, that will happen at distance 510.5 meters from the depot. The trolleybuses will go the remaining 9489.5 meters together at speed 10 meters per second. As a result, both trolleybuses will arrive to the final station by the moment of time 1000.5 seconds. The third trolleybus will not catch up with them. It will arrive to the final station by the moment of time 11000.05 seconds.

D. Wizards and Huge Prize

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

One must train much to do well on wizardry contests. So, there are numerous wizardry schools and magic fees.

One of such magic schools consists of n tours. A winner of each tour gets a huge prize. The school is organised quite far away, so one will have to take all the prizes home in one go. And the bags that you've brought with you have space for no more than k huge prizes.

Besides the fact that you want to take all the prizes home, you also want to perform well. You will consider your performance good if you win at least l tours.

In fact, years of organizing contests proved to the organizers that transporting huge prizes is an issue for the participants. Alas, no one has ever invented a spell that would shrink the prizes... So, here's the solution: for some tours the winner gets a bag instead of a huge prize. Each bag is characterized by number a_i — the number of huge prizes that will fit into it.

You already know the subject of all tours, so you can estimate the probability p_i of winning the i-th tour. You cannot skip the tour under any circumstances.

Find the probability that you will perform well on the contest and will be able to take all won prizes home (that is, that you will be able to fit all the huge prizes that you won into the bags that you either won or brought from home).

Input

The first line contains three integers n, l, k $(1 \le n \le 200, 0 \le l, k \le 200)$ — the number of tours, the minimum number of tours to win, and the number of prizes that you can fit in the bags brought from home, correspondingly.

The second line contains n space-separated integers, p_i $(0 \le p_i \le 100)$ — the probability to win the i-th tour, in percents.

The third line contains n space-separated integers, a_i $(1 \le a_i \le 200)$ — the capacity of the bag that will be awarded to you for winning the i-th tour, or else -1, if the prize for the i-th tour is a huge prize and not a bag.

Output

Print a single real number — the answer to the problem. The answer will be accepted if the absolute or relative error does not exceed 10^{-6}.

Sample test(s)

input

```
3 1 0
10 20 30
-1 -1 2
```

output

```
0.300000000000
```

input

```
1 1 1
100
123
```

output

```
1.000000000000
```

Note

In the first sample we need either win no tour or win the third one. If we win nothing we wouldn't perform well. So, we must to win the third tour. Other conditions will be satisfied in this case. Probability of wining the third tour is 0.3.

In the second sample we win the only tour with probability 1.0, and go back home with bag for it.

E. Wizards and Numbers

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

In some country live wizards. They love playing with numbers.

The blackboard has two numbers written on it — a and b. The order of the numbers is not important. Let's consider $a \le b$ for the sake of definiteness. The players can cast one of the two spells in turns:

- Replace b with $b - a^k$. Number k can be chosen by the player, considering the limitations that $k > 0$ and $b - a^k \ge 0$. Number k is chosen independently each time an active player casts a spell.
- Replace b with $b \ mod \ a$.

If $a > b$, similar moves are possible.

If at least one of the numbers equals zero, a player can't make a move, because taking a remainder modulo zero is considered somewhat uncivilized, and it is far too boring to subtract a zero. The player who cannot make a move, loses.

To perform well in the magic totalizator, you need to learn to quickly determine which player wins, if both wizards play optimally: the one that moves first or the one that moves second.

Input

The first line contains a single integer t — the number of input data sets ($1 \le t \le 10^4$). Each of the next t lines contains two integers a, b ($0 \le a, b \le 10^{18}$). The numbers are separated by a space.

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

Output

For any of the t input sets print "`First`" (without the quotes) if the player who moves first wins. Print "`Second`" (without the quotes) if the player who moves second wins. Print the answers to different data sets on different lines in the order in which they are given in the input.

Sample test(s)

input

```
4
10 21
31 10
0 1
10 30
```

output

```
First
Second
Second
First
```

Note

In the first sample, the first player should go to (11,10). Then, after a single move of the second player to (1,10), he will take 10 modulo 1 and win.

In the second sample the first player has two moves to (1,10) and (21,10). After both moves the second player can win.

In the third sample, the first player has no moves.

In the fourth sample, the first player wins in one move, taking 30 modulo 10.