A. Games

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

Manao works on a sports TV. He's spent much time watching the football games of some country. After a while he began to notice different patterns. For example, each team has two sets of uniforms: home uniform and guest uniform. When a team plays a game at home, the players put on the home uniform. When a team plays as a guest on somebody else's stadium, the players put on the guest uniform. The only exception to that rule is: when the home uniform color of the host team matches the guests' uniform, the host team puts on its guest uniform as well. For each team the color of the home and guest uniform is different.

There are n teams taking part in the national championship. The championship consists of $n \cdot (n - 1)$ games: each team invites each other team to its stadium. At this point Manao wondered: how many times during the championship is a host team going to put on the guest uniform? Note that the order of the games does not affect this number.

You know the colors of the home and guest uniform for each team. For simplicity, the colors are numbered by integers in such a way that no two distinct colors have the same number. Help Manao find the answer to his question.

Input

The first line contains an integer n $(2 \le n \le 30)$. Each of the following n lines contains a pair of distinct space-separated integers h_i, a_i $(1 \le h_i, a_i \le 100)$ — the colors of the i-th team's home and guest uniforms, respectively.

Output

In a single line print the number of games where the host team is going to play in the guest uniform.

Sample test(s)

input

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

output

```
1
```

input

```
4
100 42
42 100
5 42
100 5
```

output

```
5
```

input

```
2
1 2
1 2
```

output

```
0
```

Note

In the first test case the championship consists of 6 games. The only game with the event in question is the game between teams 2 and 1 on the stadium of team 2.

In the second test sample the host team will have to wear guest uniform in the games between teams: 1 and 2, 2 and 1, 2 and 3, 3 and 4, 4 and 2 (the host team is written first).

B. Buttons

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

Manao is trying to open a rather challenging lock. The lock has n buttons on it and to open it, you should press the buttons in a certain order to open the lock. When you push some button, it either stays pressed into the lock (that means that you've guessed correctly and pushed the button that goes next in the sequence), or all pressed buttons return to the initial position. When all buttons are pressed into the lock at once, the lock opens.

Consider an example with three buttons. Let's say that the opening sequence is: {2, 3, 1}. If you first press buttons 1 or 3, the buttons unpress immediately. If you first press button 2, it stays pressed. If you press 1 after 2, all buttons unpress. If you press 3 after 2, buttons 3 and 2 stay pressed. As soon as you've got two pressed buttons, you only need to press button 1 to open the lock.

Manao doesn't know the opening sequence. But he is really smart and he is going to act in the optimal way. Calculate the number of times he's got to push a button in order to open the lock in the worst-case scenario.

Input

A single line contains integer n $(1 \leq n \leq 2000)$ — the number of buttons the lock has.

Output

In a single line print the number of times Manao has to push a button in the worst-case scenario.

Sample test(s)

input

```
2
```

output

```
3
```

input

```
3
```

output

```
7
```

Note

Consider the first test sample. Manao can fail his first push and push the wrong button. In this case he will already be able to guess the right one with his second push. And his third push will push the second right button. Thus, in the worst-case scenario he will only need 3 pushes.

C. Beautiful Sets of Points

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

Manao has invented a new mathematical term — a beautiful set of points. He calls a set of points on a plane *beautiful* if it meets the following conditions:

1. The coordinates of each point in the set are integers.
2. For any two points from the set, the distance between them is a non-integer.

Consider all points (x, y) which satisfy the inequations: $0 \le x \le n$; $0 \le y \le m$; $x + y > 0$. Choose their subset of maximum size such that it is also a beautiful set of points.

Input

The single line contains two space-separated integers n and m ($1 \le n, m \le 100$).

Output

In the first line print a single integer — the size k of the found beautiful set. In each of the next k lines print a pair of space-separated integers — the x- and y- coordinates, respectively, of a point from the set.

If there are several optimal solutions, you may print any of them.

Sample test(s)

input
```
2 2
```
output
```
3
0 1
1 2
2 0
```

input
```
4 3
```
output
```
4
0 3
2 1
3 0
4 2
```

Note

Consider the first sample. The distance between points (0, 1) and (1, 2) equals $\sqrt{2}$, between (0, 1) and (2, 0) — $\sqrt{5}$, between (1, 2) and (2, 0) — $\sqrt{5}$. Thus, these points form a beautiful set. You cannot form a beautiful set with more than three points out of the given points. Note that this is not the only solution.

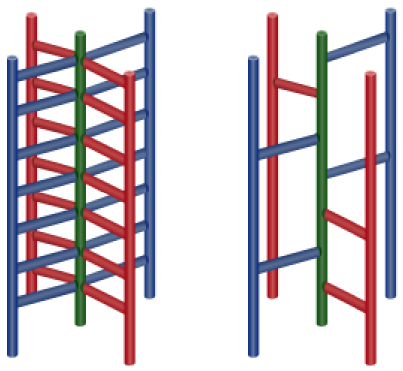
D. Wall Bars

time limit per test: 4 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

Manao is working for a construction company. Recently, an order came to build wall bars in a children's park. Manao was commissioned to develop a plan of construction, which will enable the company to save the most money.

After reviewing the formal specifications for the wall bars, Manao discovered a number of controversial requirements and decided to treat them to the company's advantage. His resulting design can be described as follows:

- Let's introduce some unit of length. The construction center is a pole of height n.
- At heights $1, 2, ..., n$ exactly one horizontal bar sticks out from the pole. Each bar sticks in one of four pre-fixed directions.
- A child can move from one bar to another if the distance between them does not exceed h and they stick in the same direction. If a child is on the ground, he can climb onto any of the bars at height between 1 and h. In Manao's construction a child should be able to reach at least one of the bars at heights $n - h + 1, n - h + 2, ..., n$ if he begins at the ground.



The figure to the left shows what a common set of wall bars looks like. The figure to the right shows Manao's construction

Manao is wondering how many distinct construction designs that satisfy his requirements exist. As this number can be rather large, print the remainder after dividing it by 1000000009 $(10^9 + 9)$. Two designs are considered distinct if there is such height i, that the bars on the height i in these designs don't stick out in the same direction.

Input

A single line contains two space-separated integers, n and h $(1 \le n \le 1000,\ 1 \le h \le min(n, 30))$.

Output

In a single line print the remainder after dividing the number of designs by 1000000009 $(10^9 + 9)$.

Sample test(s)

input

```
5 1
```

output

```
4
```

input

```
4 2
```

output

```
148
```

input

```
4 3
```

output

```
256
```

input

```
5 2
```

output

```
376
```

Note

Consider several designs for $h = 2$. A design with the first bar sticked out in direction d_1, the second — in direction d_2 and so on $(1 \le d_i \le 4)$ is denoted as string $d_1d_2...d_n$.

Design "1231" (the first three bars are sticked out in different directions, the last one — in the same as first). A child can reach neither the bar at height 3 nor the bar at height 4.

Design "414141". A child can reach the bar at height 5. To do this, he should first climb at the first bar, then at the third and then at the fifth one. He can also reach bar at height 6 by the route second $\rightarrow$ fourth $\rightarrow$ sixth bars.

Design "123333". The child can't reach the upper two bars.

Design "323323". The bar at height 6 can be reached by the following route: first $\rightarrow$ third $\rightarrow$ fourth $\rightarrow$ sixth bars.

E. Playlist

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

Manao's friends often send him new songs. He never listens to them right away. Instead, he compiles them into a playlist. When he feels that his mind is open to new music, he opens the playlist and starts to listen to the songs.

Of course, there are some songs that Manao doesn't particuarly enjoy. To get more pleasure from the received songs, he invented the following procedure of listening to the playlist:

- If after listening to some song Manao realizes that he liked it, then he remembers it and starts to listen to the next unlistened song.
- If after listening to some song Manao realizes that he did not like it, he listens to all the songs he liked up to this point and then begins to listen to the next unlistened song.

For example, if Manao has four songs in the playlist, A, B, C, D (in the corresponding order) and he is going to like songs A and C in the end, then the order of listening is the following:

1. Manao listens to A, he likes it, he remembers it.
2. Manao listens to B, he does not like it, so he listens to A, again.
3. Manao listens to C, he likes the song and he remembers it, too.
4. Manao listens to D, but does not enjoy it and re-listens to songs A and C.

That is, in the end Manao listens to song A three times, to song C twice and songs B and D once. Note that if Manao once liked a song, he will never dislike it on a subsequent listening.

Manao has received n songs: the i-th of them is l_i seconds long and Manao may like it with a probability of p_i percents. The songs could get on Manao's playlist in any order, so Manao wants to know the maximum expected value of the number of seconds after which the listening process will be over, for all possible permutations of the songs in the playlist.

Input

The first line contains a single integer n $(1 \le n \le 50000)$. The i-th of the following n lines contains two integers, separated by a single space — l_i and p_i $(15 \le l_i \le 1000,\ 0 \le p_i \le 100)$ — the length of the i-th song in seconds and the probability that Manao will like the song, in percents.

Output

In a single line print a single real number — the maximum expected listening time over all permutations of songs. The answer will be considered valid if the absolute or relative error does not exceed 10^{-9}.

Sample test(s)

input

```
3
150 20
150 50
100 50
```

output

```
537.500000000
```

input

```
4
300 0
300 50
240 50
360 80
```

output

```
2121.000000000
```

Note

Consider the first test case. If Manao listens to the songs in the order in which they were originally compiled, the mathematical expectation will be equal to 467.5 seconds. The maximum expected value is obtained by putting the first song at the end of the playlist.

Consider the second test case. The song which is 360 seconds long should be listened to first. The song 300 seconds long which Manao will dislike for sure should be put in the end.