Problem A
Good Coalition

The Dutch political system is in turmoil. There have been six coalition governments in the past fourteen years, all of which have fallen before completing their term in office. Recently there have been elections (again), the outcome of which has been described as "impossible" by several political commentators. The only bright spot in this bleak situation is that they have appointed *you* as the "informateur". As the informateur it is your task to find a suitable coalition.

Being the rational person you are, you have decided the first negotiation attempt should be started between the parties forming the *most stable coalition*. A coalition is formed by a set of parties having won a strict majority of seats in the election (i.e. at least 76 seats out of a total of 150). The most stable coalition is one that has the highest chance of completing its term in office. A coalition falls (and new elections must be held) if a single party leaves the coalition. The probability of a coalition completing their term is estimated by the product of the probabilities of each party in the coalition completing their term. This probability is in turn based on historical data.

Find the best coalition and save the Netherlands from becoming a banana republic!

Input

On the first line one positive number: the number of test cases, at most 100. After that per test case:

- one line with an integer n ($1 \leq n \leq 150$), the number of political parties that have won at least one seat in the election.
- n lines, each with two space-separated integers s_i and p_i ($1 \leq s_i \leq 150$ and $1 \leq p_i \leq 100$): the number of seats won by the i-th party and the probability (expressed as a percentage) that the i-th party will complete its term in office, respectively.

Note that there are exactly 150 seats divided among all parties $\left(\sum_{i=1}^{n} s_i = 150\right)$.

Output

Per test case:

- one line with a floating point number: the probability (expressed as a percentage) of the most stable coalition sitting out their term in office.

This number should be accurate up to 10^{-6} relative or absolute precision.

Sample Input 1	Sample Output 1
1 4 35 80 25 70 60 60 30 90	54.0

Problem B
Chasing the Cheetahs

A National Geographic film crew is visiting the ZOO this week. They are creating a documentary about animal speed and they would like to film one or more cheetahs running at full pace. A solitary running cheetah has been filmed successfully many times. Therefore, the crew is also after a much more spectacular feat: As many as possible cheetahs sprinting on parallel tracks filmed together in one shot.

"No, that is impossible," said the director. "We cannot squeeze those animals into some sort of a start box, as you probably imagine, and then open the box and make them run all at once. It is clearly too dangerous and unpredictable. No."

"Then let us make more boxes and open some of them earlier and some of them later," said one of the filmmakers. "Could that work?"

"And if we open the boxes with the slower cheetahs a bit earlier then after a while the faster ones will be overtaking the slower ones and that would be a great shot," pointed out another filmmaker. "We would like to see the whole pack rather short with the last animals close the leading ones. As close as possible and at least for a moment."

It was a long and difficult discussion which ensued, but in the end the circumstances of the experiment were agreed upon.

You are given the start time and the speed of each cheetah. The length of the pack, which is defined as the distance between the first and the last cheetah in the pack, might be different at different moments. Find the minimum length of the pack during the run, where all cheetahs must be running. You may also suppose that the track is so long that the minimum length of the pack happens at least a moment before the first cheetah reaches the finish line.

All start boxes will be so close that you may consider them to be in the same place. The k-th cheetah will be released from its start box at the given time t_k. The k-th cheetah is expected to run the whole distance at constant speed v_k.

Input

The first line contains the number of cheetahs N ($1 \leq N \leq 100\,000$). Next, there are N lines, each line contains two integers t_k, v_k separated by spaces and representing the start time and velocity of the k-th cheetah ($1 \leq k \leq N$). All input values t_k and v_k are positive and less than 10^5.

Output

Print a single line with one floating point number L specifying the minimum length of the running pack. Your answer should not differ from the correct answer by more than 10^{-2}. The length of the pack is the distance between the first and the last animal in the pack. The length can be measured at any time $T \geq \max_{k=1,\ldots,N} t_k$. We suppose that each cheetah is running at a constant speed for all the time from the start and also at its moment of release from the start box.

Sample Input 1

```
2
1 1
1 1
```

Sample Output 1

```
0.000
```

Sample Input 2

```
2
1 99999
99999 99999
```

Sample Output 2

```
9999700002.000
```

Sample Input 3

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

Sample Output 3

```
0.500
```

Problem C
Virus Synthesis

Viruses are usually bad for your health. How about fighting them with... other viruses? In this problem, you need to find out how to synthesize such good viruses.

We have prepared for you a set of strings of the letters A, G, T and C. They correspond to the DNA nucleotide sequences of viruses that we want to synthesize, using the following operations:

- Adding a nucleotide either to the beginning or the end of the existing sequence.
- Replicating the sequence, reversing the copied piece, and gluing it either to the beginning or to the end of the original (so that e.g., AGTC can become AGTCCTGA or CTGAAGTC).

We're concerned about efficiency, since we have very many such sequences, some of them very long. Find a way to synthesize them in a minimum number of operations.

Input

The first line of input contains the number of test cases T ($1 \leq T \leq 200\,000$). The descriptions of the test cases follow:

Each test case consists of a single line containing a non-empty string. The string uses only the capital letters A, C, G and T and is not longer than $100\,000$ characters.

The total number of characters in all test cases is at most $2\,600\,000$.

Output

For each test case, output a single line containing the minimum total number of operations necessary to construct the given sequence.

Sample Input 1	Sample Output 1
4 AAAA AGCTTGCA AAGGGGAAGGGGAA AAACAGTCCTGACAAAAAAAAAAAAC	3 8 6 18

This page is intentionally left blank.

Problem D
Hissing Microphone

A known problem with some microphones is the "hissing s". That is, sometimes the sound of the letter s is particularly pronounced; it stands out from the rest of the word in an unpleasant way.

Of particular annoyance are words that contain the letter s twice in a row. Words like `amiss`, `kiss`, `mississippi` and even `hiss` itself.

Input

The input contains a single string on a single line. This string consists of only lowercase letters (no spaces) and has between 1 and 30 characters.

Output

Output a single line. If the input string contains two consecutive occurrences of the letter `s`, then output `hiss`. Otherwise, output `no hiss`.

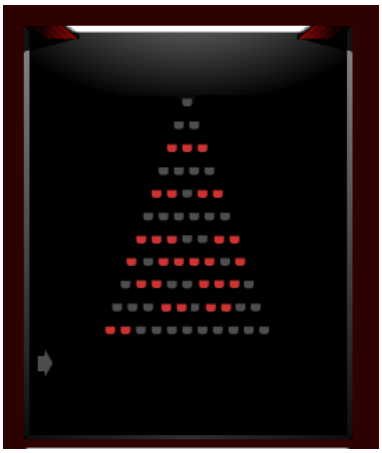
Sample Input 1	Sample Output 1
`amiss`	`hiss`

Sample Input 2	Sample Output 2
`octopuses`	`no hiss`

Sample Input 3	Sample Output 3
`hiss`	`hiss`

This page is intentionally left blank.

Problem E
Private Space



People are going to the movies in groups (or alone), but normally only care to socialize within that group. Being Scandinavian, each group of people would like to sit at least one space apart from any other group of people to ensure their privacy, unless of course they sit at the end of a row. The number of seats per row in the cinema starts at X and decreases with one seat per row (down to a number of 1 seat per row). The number of groups of varying sizes is given as a vector $(N_1, \ldots, N_n)$, where N_1 is the number of people going alone, N_2 is the number of people going as a pair etc. Calculate the seat-width, X, of the widest row, which will create a solution that seats all (groups of) visitors using as few rows of seats as possible. The cinema also has a limited capacity, so the widest row may not exceed 12 seats.

Input

The first line of input contains a single integer n ($1 \leq n \leq 12$), giving the size of the largest group in the test case.

Then follows a line with n non-negative integers, the i-th integer (1-indexed) denoting the number of groups of i persons who need to be seated. Each such number is at most 30, and not all of them are 0.

Output

A single number; the size of the smallest widest row that will accommodate all the guests. If this number is greater than 12, output `impossible` instead.

Sample Input 1	Sample Output 1
3 0 1 1	3

Sample Input 2	Sample Output 2
3 2 1 1	4

This page is intentionally left blank.

Problem F
Indoorienteering

Lukáš really likes orienteering, a sport that requires locating control points in rough terrain. To entertain the NWERC participants Lukáš wants to organize an orienteering race. However, it would be too harsh for the participants to be outdoors in this cold Swedish November weather, so he decided to jump on the new trend of indoor races, and set the race inside the B building of Linköping University.

Lukáš has already decided on the locations of the control points. He has also decided on the exact length of the race, so the only thing remaining is to decide in which order the control points should be visited such that the length of the total race is as he wishes. Because this is not always possible, he asks you to write a program to help him.

Note from the organizer: the NWERC indoorienteering race for this year has been cancelled since we neglected to apply for an orienteering permit in time from the university administration. (We still need you to solve the problem so that we can organize it for next year.)

Fredrik and Tommy lost in the B building. Photo by Tommy Olsson. cc-by-sa

Input

The input consists of:

- one line with two integers n ($2 \leq n \leq 14$) and L ($1 \leq L \leq 10^{15}$), the number of control points and the desired length of the race, respectively;
- n lines with n integers each. The jth integer on the ith line, d_{ij}, denotes the distance between control point i and j ($1 \leq d_{ij} \leq L$ for $i \neq j$, and $d_{ii} = 0$). For all $1 \leq i, j, k \leq N$ it is the case that $d_{ij} = d_{ji}$ and $d_{ij} \leq d_{ik} + d_{kj}$.

Output

Output one line with "`possible`" if it is possible to visit all control points once in some order and directly return to the first one such that the total distance is exactly L, and "`impossible`" otherwise.

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

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

Problem G
Peg Solitaire

The game of peg solitaire, popular at the court of the French king Louis XIV, has the following rules. Given a two-dimensional board with a mesh of holes, each hole can contain one peg (pin). The only legal move of a peg is a vertical or horizontal jump over an adjacent peg into the empty hole next to the jumped peg in line with it, which is then removed. The original goal of the game was to leave a single peg in the predefined position on the board by performing only legal moves. Obviously, such a solution is possible only for certain board forms and starting configurations. To drop this restriction, we slightly redefine the problem: Given the starting configuration of the board, determine the minimum number of pegs that can be achieved by means of legal moves as well as the minimum number of moves required to reach that number of pegs.

Input

The first line of the input contains one number, $1 \leq N \leq 100$ which represents the number of test cases. Each test case is described by the following lines of input that represent the initial state of the solitaire board.

In this representation '`.`' denotes an empty hole and '○' a hole with a peg in it. '`#`' indicates a part of the board without a hole. All boards have the same shape, see sample input (that includes position of holes). In its initial state, the board can contain at most 8 pegs. There is an empty line between two consecutive test cases.

Output

For each test case your program should output a line with two numbers separated by a single whitespace, with the first one denoting the minimum number of pegs achievable by legal moves starting with the given initial state, and the second one providing the minimum required number of moves.

Sample Input 1

```
3
###...###
..○○.....
.....○○..
.........
###...###

###...###
..○○.○...
...○.○○..
...○○....
###...###

###○..###
.○.○○....
○.○......
.○.○.....
###...###
```

Sample Output 1

```
1 3
1 7
1 7
```

Sample Input 2	Sample Output 2
3 ###...### ..○○..... .....○○.. .... ###...### ###...### ..○○.○... ...○.○○.. ...○○.... ###...### ###○..### .○.○○.... ○.○...... .○.○..... ###...###	1 3 1 7 1 7

Problem H
Particle Collision

Particle colliders are difficult to build and experiments are costly to run. Before running any real experiments it is better to do a simulation to test out the ideas first. You are required to write a very simple simulator for this problem.

There are only three particles in this system, and all particles are confined to an infinite plane so that they can be modelled as circles. Their locations are specified only by the x_i and y_i coordinates of their centers ($1 \leq i \leq 3$). All three particles have the same radius r, and are initially stationary.

We are given a vector (x_v, y_v) specifying the direction particle 1 will move when the experiment starts. When particle i hits particle j, particle j will start moving in the direction perpendicular to the tangent at the point of the contact, away from particle i. Particle i will cease to exist and be converted to radiation. A moving particle that does not hit another will continue moving indefinitely.

There are a number of possible scenarios your simulator should identify:

1. particle 1 hits particle 2, which in turns hits particle 3;
2. particle 1 hits particle 3, which in turns hits particle 2;
3. particle 1 hits particle 2, which moves indefinitely;
4. particle 1 hits particle 3, which moves indefinitely;
5. particle 1 moves indefinitely.

Input

The input contains four lines. The first three lines each contains two integers x_i and y_i ($|x_i|, |y_i| \leq 1000$), describing particles 1, 2, and 3 in this order. The fourth line contains three integers x_v, y_v, and r ($|x_v|, |y_v| \leq 1000$, $0 < r \leq 50$).

You may assume that no two particles touch or overlap initially, and that the distance between the centers of particles 2 and 3 is greater than $4r$.

Output

Output a single integer giving the number (1–5) identifying the scenarios described above.

Although you should take care of your calculations, it is guaranteed that the outcome would not change if the initial vector (x_v, y_v) is rotated by one degree either way.

Sample Input 1	Sample Output 1
0 0 50 45 91 50 42 50 10	1

Sample Input 2

```
0 0
50 50
141 50
41 50 10
```

Sample Output 2

```
3
```

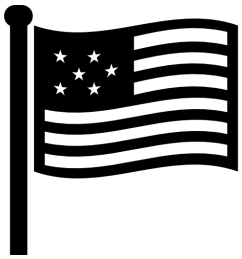
Sample Input 3

```
0 0
50 50
131 50
100 50 10
```

Sample Output 3

```
4
```

Problem I
Star Arrangements



The recent vote in Puerto Rico favoring United States statehood has made flag makers very excited. An updated flag with 51 stars rather than the current one with 50 would cause a huge jump in U.S. flag sales. The current pattern for 50 stars is five rows of 6 stars, interlaced with four offset rows of 5 stars. The rows alternate until all stars are represented.

```
* * * * * *
 * * * * *
* * * * * *
 * * * * *
* * * * * *
 * * * * *
* * * * * *
 * * * * *
* * * * * *
```

This pattern has the property that adjacent rows differ by no more than one star. We represent this star arrangement compactly by the number of stars in the first two rows: `6,5`.

A 51-star flag that has the same property can have three rows of 9 stars, interlaced with three rows of 8 stars (with a compact representation of `9,8`). Conversely, if a state were to leave the union, one appealing representation would be seven rows of seven stars (`7,7`).

A flag pattern is *visually appealing* if it satisfies the following conditions:

- Every other row has the same number of stars.
- Adjacent rows differ by no more than one star.
- The first row cannot have fewer stars than the second row.

Your team sees beyond the short-term change to 51 for the US flag. You want to corner the market on flags for any union of three or more states. Given the number S of stars to draw on a flag, find all possible *visually appealing* flag patterns.

Input

The input consists of a single line containing the integer S ($3 \leq S \leq 32\,767$).

Output

On the first line, print S, followed by a colon. Then, for each visually appealing flag of S stars, print its compact representation, one per line.

This list of compact representations should be printed in increasing order of the number of stars in the first row; if there are ties, print them in order of the number of stars in the second row. The cases 1-by-S and S-by-1 are trivial, so do not print those arrangements.

The compact representations must be printed in the form "`x,y`", with exactly one comma between `x` and `y` and no other characters.

Sample Input 1	Sample Output 1
3	3: 2,1

Sample Input 2	Sample Output 2
50	50: 2,1 2,2 3,2 5,4 5,5 6,5 10,10 13,12 17,16 25,25

Sample Input 3	Sample Output 3
51	51: 2,1 3,3 9,8 17,17 26,25

Problem J
Outer Space Invaders

The aliens from outer space have (finally!) invaded Earth. Defend yourself, or be disintegrated! Or assimilated. Or eaten. We are not yet sure.

The aliens follow a known attack pattern. There are n attackers, the i-th one appears at time a_i, at distance d_i from you. He must be destroyed no later than at time b_i, or else he will fire his weapon, which will definitely end the fight.

Your weapon is an area-blaster, which can be set to any given power. If fired with power R, it momentarily destroys all aliens at distance R or smaller. It also consumes R fuel cells.

Determine the minimal cost (measured in fuel cells) of destroying all the aliens, without being killed.

Input

The first line of input contains the number of test cases T ($1 \leq T \leq 5$). The descriptions of the test cases follow:

Each test case starts with a line containing the number of aliens n ($1 \leq n \leq 300$). Of the next n lines, the i-th one contains three integers a_i, b_i, d_i, ($1 \leq a_i < b_i \leq 10\,000$; $1 \leq d_i \leq 10\,000$). The i-th alien appears at time a_i, is idle until b_i, and his distance from you is d_i.

Output

For each test case, output one line containing the minimum number of cells needed to destroy all the aliens.

Sample Input 1	Sample Output 1
1 3 1 4 4 4 7 5 3 4 7	7

This page is intentionally left blank.

Problem K
Plankton Food

Only few ZOOs can afford to cultivate all types of plankton food they need to feed to various sea vertebrates and invertebrates which live in their voluminous aquariums. Some ZOOs might be from time to time in a short supply of a particular kind of plankton food which is momentarily difficult to obtain. On the other hand, they also might store some reserves of other plankton food types which they do not need to spend immediately and those reserves might be huge.

The director is currently in a pressing need for a particular type of plankton food as the construction of a new aquarium with thermal vents is being finished. Fortunately, there are many colleagues in other ZOOs willing to help. They presented the director with various offers. After many lengthy calls, the director has built a comprehensive list of all available offers. Now it is obvious to him that he might need to trade in more steps to obtain the desired food type. In the first trade step, he would exchange the type A, of which there is plenty in his ZOO, for some other type B, which in fact he does not need at all but which can be exchanged with another ZOO for some amount of another type C food, and so on until the final exchange is made which brings in the appropriate food type.

Looking into the list the director is not sure whether a desired chain of exchanges indeed does exist. He calls in his economy vice-director and asks for help. The vice-director studies the list carefully and finally says:

"The amount of food the ZOOs are willing to give in exchange is sometimes bigger and sometimes smaller than the amount of food they would receive and that depends, of course, on the type of the food and generosity of a particular ZOO. I cannot say now if the desired chain of exchanges exists. If it exists, then maybe it would be possible to utilize the offers in such a way that we receive theoretically an unlimited supply of the desired food for only a small investment of the food we have. All these possibilities have to be checked."

You are presented with the list of offers of all other ZOOs. You have to decide if a sequence of exchanges might be organized in such a way that it brings in a theoretically unlimited supply of the needed type of the plankton food in exchange for a limited volume of the unnecessary food (provided that the sources in the other ZOOs are unlimited). We suppose that any exchange based on a particular trade offer might happen arbitrary number of times.

Input

The first line of input contains the number of test cases T ($1 \leq T \leq 5$). The descriptions of the test cases follow:

Each test case starts with a line containing four positive integers T, F, F_u, F_n. T represents the number of trade offers, F represents the number of types of plankton food. The types of food are labeled $1, 2, \ldots, F$. F_u is the label of the unnecessary food which the organizing ZOO is offering, F_n is the label of the necessary food which it wants to obtain by the trade. Next, there are T lines, each line represents a particular trade offer of a particular ZOO. The offer is expressed by three values F_r, F_g, U. F_r and F_g are labels of particular food types. The third number, U, is the natural logarithm (the use of logarithms in the ZOO is very popular because of the large range of sizes of the animals) of the amount of units of plankton food of type F_g which the offering ZOO is willing to give in exchange of receiving 1 unit of plankton food of type F_r. The value of U is a decimal number with at most three digits after the decimal point and with absolute value less than or equal to $1\,000$. Note that the identification of the offering ZOO has been stripped away, as it is not essential for the solution of the problem. The value of T does not exceed $10\,000$, the value of F does not exceed $5\,000$.

Output

For each test case print on a separate line either "TRUE" or "FALSE". Print "TRUE" if and only if there is a way to organize the exchanges in such way that an investment of a limited volume of the unnecessary food type can result in theoretically unlimited supply of the necessary food type.

Sample Input 1	Sample Output 1
3 3 3 1 3 1 2 0.001 2 3 -0.002 3 1 0.001 3 3 1 3 1 2 1.5 2 3 0.5 3 1 -1.999 6 5 1 5 1 2 -0.1 2 3 0.1 3 4 0.1 4 2 0.1 2 5 -0.1 1 5 0	FALSE TRUE TRUE

Problem L
DRM Messages

DRM Encryption is a new kind of encryption. Given an encrypted string (which we'll call a *DRM message*), the decryption process involves three steps: Divide, Rotate and Merge. This process is described in the following example with the DRM message "EWPGAJRB":

Divide – First, divide the message in half to "EWPG" and "AJRB".

Rotate – For each half, calculate its rotation value by summing up the values of each character (A $=$ 0, B $=1, \ldots,$ Z $=25$). The rotation value of "EWPG" is $4+22+15+6=47$. Rotate each character in "EWPG" 47 positions forward (wrapping from Z to A when necessary) to obtain the new string "ZRKB". Following the same process on "AJRB" results in "BKSC".

Merge – The last step is to combine these new strings ("ZRKB" and "BKSC") by rotating each character in the first string by the value of the corresponding character in the second string. For the first position, rotating 'Z' by 'B' means moving it forward 1 character, which wraps it around to 'A'. Continuing this process for every character results in the final decrypted message, "ABCD".

Input

The input contains a single DRM message to be decrypted. All characters in the string are uppercase letters and the string's length is even and $\leq 15\,000$.

Output

Display the decrypted DRM message.

Sample Input 1	Sample Output 1
EWPGAJRB	ABCD

Sample Input 2	Sample Output 2
UEQBJPJCBUDGBNKCAHXCVERXUCVK	ACMECNACONTEST

This page is intentionally left blank.