Problem A. Arbitrage

Input file:	`arbitrage.in`
Output file:	`arbitrage.out`
Time limit:	2 seconds
Memory limit:	256 megabytes

Serge is working in a large Antarctic financial company. Recently he noticed that he has access to a private currency exchange that sometimes offers better exchange rates than usual interbank exchange. He decided that he would make some exchanges and keep extra money from operations for himself.

After some investigations, Serge developed the following model. He makes all calculations in Antarctic dollars AD, converting each currency to AD using Antarctic central bank rate. For some pairs (i, j) of currencies he can make an exchange operation by changing some amount of the i-th currency to some amount of the j-th currency. As a result he gets profit equivalent to $a_{i,j}$ antarctic dollars (AD) for each 1000 AD equivalent he exchanges. (this value can be negative, that means that Serge loses money on this operation). In order to stay out of suspicions about his tricks, the amount of currency changed must not exceed equivalent of $c_{i,j}$ thousands AD. After all operations are completed, the amount of each currency must stay the same as before the operations, so for each i the sum of equivalents in AD that were exchanged from the i-th currency must be the same as the sum of equivalents in AD that were exchanged to the i-th currency. Interface of exchange system only allows exchange operations with integer number of 1000 AD equivalents.

Help Serge find out what is the maximal profit he can get.

Input

The first line of the input file contains n — the number of foreign currencies available for trade ($2 \le n \le 30$) and m — the number of possible exchange operations ($1 \le m \le n \cdot (n-1)$).

Each of the following m lines describes possible exchange operation. Each operation is specified by number of source currency i, number of target currency j, profit for each thousand AD equivalent exchange $a_{i,j}$ ($-100 \le a_{i,j} \le 100$, $a_{i,j}$ is integer) and maximal possible number of thousands AD that can be exchanged $c_{i,j}$ ($1 \le c_{i,j} \le 100$, $c_{i,j}$ is integer). Using this operation Serge can exchange the i-th currency to the j-th one.

Output

Output one integer — the maximal profit Serge can get.

Example

arbitrage.in	arbitrage.out
4 6 1 2 3 2 2 1 -1 1 2 3 0 1 3 4 0 2 4 1 2 2 1 3 -3 1	7

Problem B. Border

Input file:	`border.in`
Output file:	`border.out`
Time limit:	2 seconds
Memory limit:	256 megabytes

Colonists from Earth are preparing to build a new power field border on Pandora. They would like to surround their base together with some surrounding area.

The colonists' base has the form of a triangle with sides equal to a, b and c, respectively. The power generated by the base allows to create the border field of total length l. Colonists would like to create such border that it completely contains the base and contains maximal possible area outside the base.

Help them to create such border.

Input

The input file contains four integer numbers: a, b, c and l ($1 \le a, b, c \le 100$, $a + b + c \le l \le 2000$, a, b and c are sides of a non-degenerate triangle).

Output

Output one real number: the maximal possible area that can be surrounded by the border. Your output must have absolute or relative error of at most 10^{-6}.

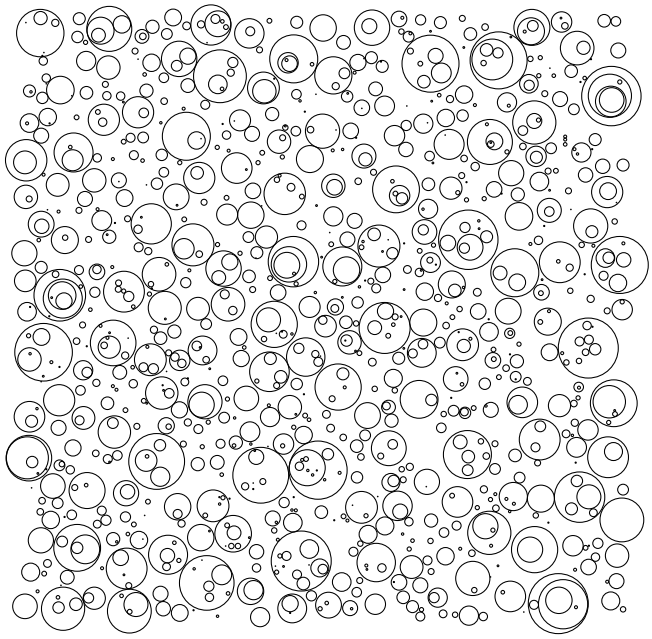
Example

border.in	border.out
3 4 5 100	795.774715459476679
3 4 5 12	6.0
3 4 5 13	12.261818799960896

Problem C. Circles

Input file:	`circles.in`
Output file:	`circles.out`
Time limit:	5 seconds
Memory limit:	256 megabytes

There are n circles located on the plane. Circles may have common points, but for any two circles their intersection is either a point, or one of the two circles.



Find the total area covered by at least one circle.

Input

The first line of the input file contains integer number n ($1 \le n \le 100\,000$). The following n lines contain three integers each and describe circles. The i-th circle is described by coordinates of its center x_i and y_i and its radius r_i ($-10^6 \le x_i, y_i \le 10^6$, $1 \le r_i \le 10^6$).

Output

Output one real number: the total area covered by at least one circle. Your answer must have absolute or relative error of at most 10^{-9}.

Example

circles.in	circles.out
4 2 2 2 2 2 1 5 2 1 5 5 2	28.2743338823081391

Problem D. Diamonds and Golden Strings

Input file:	`diamonds.in`
Output file:	`diamonds.out`
Time limit:	2 seconds
Memory limit:	256 megabytes

Ancient Aztec civilization is famous for its exotic rituals. Recently scientists have discovered ancient ruins that give evidence that the following game was popular among Aztec nobles.

Several golden necklaces with diamonds are put to the table. Two players make moves in turn.

Each necklace consists of several diamonds connected to each other by golden strings to form a cycle. A player that makes a move cuts one string. If after cutting the string some diamond has no more diamonds connected to it, the player who cut the string takes the diamond and continues his move by cutting another string. If after making a cut each diamond still has some diamond attached to it, the move passes to another player.

When no more diamonds are left on the table the game ends. During the game each player tries to maximize the number of diamonds he takes.

The archaeologists that discovered the ruins found the description of the game that started with n necklaces containing $a_1, a_2, \ldots, a_n$ diamonds, respectively. Now they wonder how many diamonds would each of the players get should both of them play optimally.

Input

The first line of the input file contains n — the number of necklaces ($1 \le n \le 100$). The second line contains n integers: the sizes of the necklaces $a_1, a_2, \ldots, a_n$ ($3 \le a_i \le 10^9$).

Output

Output two integers: the number of diamonds the first player gets and the number of diamonds the second player gets.

Example

diamonds.in	diamonds.out
2 3 3	3 3
2 5 5	4 6

In the second example the optimal play proceeds as follows. The first player cuts any cycle. The second player has the chain of length 5 and a cycle of length 5. He cuts away and takes one diamond from the chain of length 5 and cuts the remaining chain of length 4 to two chains of length 2. The first player now must cut the cycle any way, so he takes two chains of length 2 by cutting each, and then cuts the cycle. The second player now gets a chain of length 5 and cuts all diamonds from it one after another.

Problem E. Expedition to Mars

Input file:	`expedition.in`
Output file:	`expedition.out`
Time limit:	2 seconds
Memory limit:	256 megabytes

Earth colonists are planning to build the new expedition base on Mars. The base would consist of n equal cubic containers set up on the planet surface.

The arrangement of containers must follow several rules. It is not allowed to put containers on top of one another, so they must all be put on the planet surface side by side. The containers must be set up in such way that their sides are parallel to North-South and West-East directions. If two containers touch each other, they must have either a common side, or a common corner. Finally, since the outer sides of containers need protection from hazardous Mars environment, the containers must be set up in such way that the total length of outer sides is minimal possible.

Now the leader of the expedition wonders what is the number of ways to set up containers to create the expedition base. This number can be quite large, so you must find it modulo $10^9 + 7$.

Input

The input file contains one integer number: the number of containers ($1 \le n \le 500$).

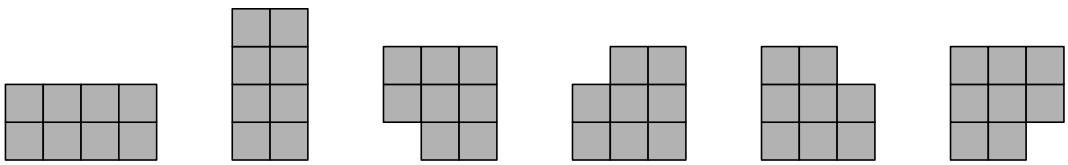
Output

Output one integer number: the number of ways to set up the expedition base modulo $10^9 + 7$.

Example

expedition.in	expedition.out
8	6

Possible base configurations are shown on the following picture.



Problem F. Formula Verification

Input file:	`formula.in`
Output file:	`formula.out`
Time limit:	2 seconds
Memory limit:	256 megabytes

It is well known that first order logic over integers is undecidable. No algorithm can take formula with $\forall$ and $\exists$ quantifiers, addition, multiplication and relations and output whether the formula is true over integers.

Unlike integers, logic of first order for real numbers is decidable. In this problem you have to verify first order logic formula with two variables and two quantifiers over real numbers.

Let us define formulae by the following BNF.

$$
\begin{aligned}
\langle \text{formula} \rangle &\longrightarrow \langle \text{quantifier} \rangle \texttt{ x } \langle \text{quantifier} \rangle \texttt{ y } \langle \text{polynomial} \rangle \langle \text{relation} \rangle \langle \text{polynomial} \rangle \\
\langle \text{quantifier} \rangle &\longrightarrow \forall \mid \exists \\
\langle \text{relation} \rangle &\longrightarrow \texttt{=} \mid \texttt{<} \mid \texttt{>} \mid \texttt{<=} \mid \texttt{>=} \\
\langle \text{polynomial} \rangle &\longrightarrow \langle \text{term} \rangle \mid \langle \text{polynomial} \rangle + \langle \text{term} \rangle \mid \langle \text{polynomial} \rangle - \langle \text{term} \rangle \\
\langle \text{term} \rangle &\longrightarrow \langle \text{factor} \rangle \mid \langle \text{term} \rangle * \langle \text{factor} \rangle \\
\langle \text{factor} \rangle &\longrightarrow x \mid y \mid \langle \text{integer} \rangle \\
\langle \text{integer} \rangle &\longrightarrow \textit{integer from 0 to 100}
\end{aligned}
$$

Additionally, in definition above "⟨polynomial⟩" is a polynomial of variables x and y of degree at most 2.

The formula $\forall a F(a)$ is true if for all real numbers a the predicate $F(a)$ is true. The formula $\exists a F(a)$ is true if for at least one real number a the predicate $F(a)$ is true.

Examples of true formulae: "$\forall x \exists y\, x < y$", "$\forall x \exists y\, x * x = y$", "$\forall x \forall y\, x * x + y * y \geq 0$". Examples of false formulae: "$\exists x \forall y\, x < y$", "$\forall x \exists y\, x = y * y$".

Input

Input file contains at most 20 test cases, one on a line.

Each line of the input file contains the first order formula satisfying constraints described. The formula uses variables "`x`" and "`y`", operations "`*`", "`+`", "`-`", relations "`>`", "`<`", "`=`", "`>=`", "`<=`", quantifiers "`A`" for $\forall$, and "`E`" for $\exists$, and non-negative integer numbers not exceeding 100. Spaces can occur at arbitrary locations in the input except inside integers and relations. When converting given polynomials to a form where no distinct terms have the same powers of both x and y no coefficient would exceed 1000, including those appearing in reasonable internal calculations. The length of each line doesn't exceed 200.

Output

For each formula output one line. Output "`true`" if the formula is true over real numbers, or "`false`" if the formula is false.

Example

formula.in	formula.out
AxEy y=x*x AxEy x=y*y	true false

Problem G. Great Minds

Input file:	great.in
Output file:	great.out
Time limit:	5 seconds
Memory limit:	256 megabytes

The judges of the TV show "Great Minds" have prepared n questions for the show. The questions prepared are ranked from 1 to n according to their difficulty, so that no two questions got the same difficulty rank.

Initially all questions are arranged in n slots along the line, the question in the i-th slot has difficulty a_i.

Three players take part in the game. In the beginning of the show each of them chooses a question. Different players must choose different questions.

During the game players try to answer questions. If the player answes a question from slot j correctly, he can either leave the game or choose a new question in one of the slots with the number greater than j. The difficulty of this question must be greater than the difficulty of his last question. If the players altogether answer all questions, they win a special jackpot that is divided fairly among them.

Before the game starts the initial arrangement of questions to slots is unknown, but the players have noticed that in some cases they cannot win the jackpot, no matter how hard they try. For example, if the arrangment of questions is $\langle 4, 3, 2, 1\rangle$, the players cannot answer all questions because of game rules (for any question there is no question with higher difficulty in a slot with greater number). Now the players wonder how many questions arrangements allow them to win the jackpot.

Help them to find this number. It can be rather large, so they are satisfied with its value modulo $10^9 + 7$.

Input

Input file contains one integer number n ($3 \le n \le 500$).

Output

Output one integer number — the number of arrangements of questions to slots that allow players to win the jackpot.

Example

great.in	great.out
4	23

Problem H. Highways

Input file:	highways.in
Output file:	highways.out
Time limit:	2 seconds
Memory limit:	256 megabytes

Recently the king of Flatland has reviewed spendings of the kingdom budget and noticed that they are spending too much on highways maintenance. The king called his minister of transport and asked whether the system of highways in the kingdom is redundant and some highways should be abandoned.

There are n cities in Flatland, connected by m bidirectional highways. The king has chosen a number k and decided to call the system of highways redundant if there is a set A containing s cities such that there are more than $k(s-1)$ highways between cities of A. The minister claims that the system is not redundant, but the king doesn't believe him. He hires you to check whether the system is redundant.

Input

The first line of the input file contains three integers: n, m and k — the number of cities, the number of highways and redundancy level ($2 \le n \le 100$, $1 \le m \le n(n-1)/2$, $1 \le k \le 20$). Each of the following lines describes a highway. Each highway is described by the numbers of cities it connects. Cities are numbered from 1 to n. Each pair of cities is connected by at most one highway, no highway connects a city to itself.

Output

If the system of highways is not redundant, print "OK" on the first line of the output file.

If the system of highways is redundant, print "Redundant" on the first line of the output file. In this case description of the set A which certifies redundancy must follow. The description must start with s — the number of cities in A followed by s integer numbers — cities in A. If there are several such sets, output any one.

Example

highways.in	highways.out
5 8 2 1 2 1 3 2 3 3 5 3 4 5 2 2 4 4 5	OK
5 5 1 1 2 2 3 3 4 4 5 5 1	Redundant 5 1 2 3 4 5

Problem I. Intercity Express

Input file:	`intercity.in`
Output file:	`intercity.out`
Time limit:	5 seconds
Memory limit:	256 megabytes

Andrew is developing the a system for train ticket sales. He is going to test it on Intercity Express line that connects two large cities and has $n - 2$ intermediate stations, so there are a total of n stations numbered from 1 to n.

Intercity Express train has s seats numbered from 1 to s. In test mode the system has access to a database that contains already sold tickets in direction from station 1 to station n and needs to answer questions whether it is possible to sell a ticket from station a to station b and if so, what is the minimal number of seat that is vacant on all segments between a and b. Initially the system will have read only access, so even if there is a vacant seat, it should report so, but should not modify the data to report it reserved.

Help Andrew to test his system by writing a program that would answer such questions.

Input

The first line of the input file contains n — the number of stations, s — the number of seats and m — the number of already sold tickets ($2 \le n \le 10^9$, $1 \le s \le 100\,000$, $0 \le m \le 100\,000$). The following m lines describe tickets, each ticket is described by c_i, a_i, and b_i — the seat that the owner of the ticket occupies, the station from which the ticket is sold, and the station to which the ticket is sold ($1 \le c_i \le s$, $1 \le a_i < b_i \le n$).

The following line contains q — the number of queries ($1 \le q \le 100\,000$). A special value p must be maintained when reading queries. Initially $p = 0$. The following $2q$ integers describe queries. Each query is described with two numbers: x_i and y_i ($x_i < y_i$). To get cities a and b between which the seat availability is requested use the following formulae: $a = x_i + p$, $b = y_i + p$. The answer to the query is 0 if there is no seat that is vacant on each segment between a and b, or the minimal number of seat that is vacant.

After answering the query, assign the answer for the query to p.

Output

For each query output the answer to it.

Example

intercity.in	intercity.out
5 3 5 1 2 5 2 1 2 2 4 5 3 2 3 3 3 4 10 1 2 1 2 1 2 2 3 -2 0 2 4 1 3 1 4 2 5 1 5	1 2 2 3 0 2 0 0 0 0

Note that actual queries are $(1, 2)$, $(2, 3)$, $(3, 4)$, $(4, 5)$, $(1, 3)$, $(2, 4)$, $(3, 5)$, $(1, 4)$, $(2, 5)$, $(1, 5)$.

Problem J. Jungle Speed

Input file:	`jungle.in`
Output file:	`jungle.out`
Time limit:	2 seconds
Memory limit:	256 megabytes

Jungle Speed is a card game developed by Asmodee Editions. It is played with non-standard playing cards, we will denote cards of different types with different uppercase letters of the English alphabet. This problem describes a simplified version of the game.

There are several types of cards, the playing deck contains two cards of each type. All cards are initially dealt to the players, each player puts her cards in a stack in front of her face down. We will call this stack player's "supply stack". During the game some cards from the supply stack are piled face up in front of the player in another stack that we will call "discard stack".

Players make moves in clockwise order starting from the youngest. Let us number players from 1 to n in this order.

During her move the player takes the topmost card of her stack and puts it face up on top of her discard stack. If after this action there are two cards of the same type on top of two different discard stacks, the two players for which the top cards of discard stacks coincide are said to be matching.

The matching players must grab a special totem. The one who reacts faster, grabs the totem and the losing player must collect the cards from both her and another matching player's discard stacks. She starts from her own discard stack and adds cards to her supply stack one after another, starting from the top card. Then she collects cards from another matching player, starting from the top card again. She adds collected cards under the cards she has, to the bottom of her supply stack.

If the player to move has no cards in her supply stack, the move passes to the next player clockwise. The move is performed only when the card is put to the table.

The game ends when one of the following conditions is met:

- Some player has no cards in neither her supply stack, nor her discard stack. This player is declared a winner. If there are several such players, the one who cleared her supply stack earlier wins.
- All cards are put on the table in discard stacks and no two top cards of discard stacks are matching. In this case the game ends in a draw.
- The game continues for 10^6 moves and still all players have cards in their stacks, and at least one player has cards in her supply stack. In this case game is abandoned, and a new game starts.

A group of n players decided to play Jungle Speed. Given initial configuration of players' supply stacks, find out who wins the game, or whether the game would end in a draw or would be abandoned.

Input

The first line of the input file contains n — the number of players ($2 \le n \le 20$). The second line contains n integer numbers $a_1, a_2, \ldots, a_n$: reactions of the players ($1 \le a_i \le 100$). If two players must grab the totem, the player with smaller reaction grabs it. If two players with the same reaction must grab the totem, the player who opened the last card loses the race and the other player grabs the totem.

The following n lines contain a string of uppercase English letters each and denote players' initial supply stacks. Each player has at least one card in her supply stack initially. Each card type has at most two occurencies in players' stacks. The cards in stacks are given from top to bottom.

Output

If the game ends in a draw, output "`Draw after X moves.`" where X is the number of moves made before all cards are in players' discard stacks. Remember that a situation when a player has no cards in her supply stack and passes the move clockwise is not counted as a move.

If some player wins the game, output "`Player Y wins after X moves.`" where Y is the number of the player that wins the game and X is the number of moves made before the player Y wins, including her winning move.

If the game is abandoned after 10^6 moves, output "`Abandoned after 1000000 moves.`".

Example

jungle.in	jungle.out
2 1 2 ABC CBA	Player 1 wins after 8 moves.
2 2 1 ABC CBA	Player 2 wins after 8 moves.
2 2 1 ABCD CDAB	Draw after 8 moves.