Problem A. Everyone Loves Playing Games

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	256 mebibytes

One day, Acesrc and Roundgod are playing an interesting game called Important Choice Pairs of Cakes (ICPC).

In this game, there is a variable X. Initially, X is equal to 0. N pairs of numbers (x_i, y_i) are given to Acesrc while M pairs (x'_i, y'_i) are given to Roundgod.

Firstly, for every pair (x_i, y_i), Acesrc will choose either x_i or y_i. Suppose he choose k, X will be changed to $(X \oplus k)$. ($\oplus$ denotes bitwise exclusive or)

After Acesrc's N operations, Roundgod will do the same with his M pairs.

They know about each other's pairs from the beginning. Acesrc wishes the final value of X to be as great as possible while Roundgod wishes it to be as small as possible.

Acesrc and Roundgod are very clever boys and they will choose the best strategy. Can you predict the final value of X?

Input

There are multiple test cases. The first line of the input contains an integer T ($1 \le T \le 20$), indicating the number of test cases. For each test case:

The first line contains two integers N and M ($1 \le N, M \le 10000$).

Then N lines follow. In each line, there are two integers x_i, y_i ($1 \le x_i, y_i \le 10^{18}$), representing Acesrc's pairs.

Then M lines follow. In each line, there are two integers x'_i, y'_i ($1 \le x'_i, y'_i \le 10^{18}$), representing Roundgod's pairs.

Output

For each test case, you should output a single integer in a line as your answer.

Example

standard input	standard output
2 1 1 6 3 4 1 2 2 1 3 4 6 5 4 2 2	2 2

Note

In the first sample, if Acesrc chooses 6, Roundgod will choose 4 and the result will be 2.

If Acesrc chooses 3, Roundgod will choose 1 and the result will also be 2.

Therefore the answer is 2.

Problem B. Gifted Composer

Input file:	*standard input*
Output file:	*standard output*
Time limit:	4 seconds
Memory limit:	256 mebibytes

Acesrc is a gifted composer. He likes writing tuneful and melodic songs. Every song he writes can be viewed as a sequence of musical notes, and each musical note represents the pitch and the duration of the sound. In this problem, we consider only the following seven primary pitches

`do re mi fa sol la si`

and the duration of each note is one unit time. Hence, there are only seven types of notes, and we may use the pitch name to represent a note.

Acesrc composes a song in the following way. Initially, the sequence of notes is empty. Every day, he inserts a new note at the beginning or at the end of the sequence, until the song is done.

Acesrc particularly likes songs with repetitions. For a song with n musical notes, we say the song has a repetition of length k ($1 \le k \le n$), if the song can be partitioned into one or more identical sections with k notes, optionally followed by an incomplete section, which is an initial part of a complete section. For example, `do re do re do` can be partitioned into `do re | do re | do`, so it has a repetition of length 2; similarly, `do re mi do re mi` has a repetition of length 3, and `do re do re mi` has a repetition of length 5.

Acesrc wants to know, after he adds a note each day, the number of different lengths of repetitions the song has. Can you help him?

Input

The first line of input consists of a single line n ($1 \le n \le 10^6$), the number of days Acesrc uses to compose the song. The ith of the remaining n lines contains a character a ($a \in \{\texttt{p}, \texttt{a}\}$) (where `p` denotes prepend, i.e., inserting at the beginning, and `a` denotes append, i.e., inserting at the end) and a string s ($s \in \{\texttt{do}, \texttt{re}, \texttt{mi}, \texttt{fa}, \texttt{sol}, \texttt{la}, \texttt{si}\}$), representing the action Acesrc takes in the ith day.

Output

Output n lines. The ith line should be a single integer, denoting the answer for the ith day.

Examples

standard input	standard output
5 a do p re a re a do p do	1 1 2 2 3
5 a re a do a re p do a mi	1 1 2 2 1

Problem D. String Theory

Input file:	*standard input*
Output file:	*standard output*
Time limit:	4 seconds
Memory limit:	512 mebibytes

Acesrc is a famous string theorist at Nanjing University second to none. He insists that we should always say an important thing k times. He also believes that every string that can be obtained by concatenating k copies of some nonempty string is splendid. So, he always teaches newcomers, "String theory problems are important! String theory problems are important! ... String theory problems are important!"

Today, he wants to examine whether the newcomers remember his instruction. He presents a string consisting of lower case letters and asks them the number of splendid substrings of the presented string. No one can solve this problem, and they will be scolded for hours. Can you help them solve this problem?

Note that equal splendid substrings occurred in different positions should be counted separately.

Input

The first line of input consists of a single integer T $(1 \le T \le 10)$, denoting the number of test cases. Each test case starts with a single line of an integer k $(1 \le k \le 20)$. The second line of a test case is a string S consisting of lower case letters only, the length of which is between 1 and 3×10^5 inclusive. The sum of the lengths of strings in all test cases never exceeds 10^6.

Output

For each test case, print the answer as a single integer in a single line.

Example

standard input	standard output
3 2 aabb 2 abababab 3 abc	2 6 0

Problem E. Road Construction

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	256 mebibytes

There are $n+m$ towns in Kingdom of Coffee Chicken, which can be seen as $n+m$ integers coordinates (x_i, y_i) on the 2-dimensional plane. n of them belong to Acesrc while the other m towns belong to Roundgod.

Now both Acesrc and Roundgod want to build straight roads among their towns and they all want their towns are connected, which means there is a path between any two of towns. It is obvious that we need only $n+m-2$ roads to make it possible. Moreover, Acesrc and Roundgod hope that among these $n+m-2$ roads, there is no intersection other than the position of towns.

Now we hope you to provide us a construction plan.

Input

The first line contains two integers $n, m(n > 1, m > 1, n+m \leq 3000)$.

The following n lines describe Acesrc's towns and each line contains two integers $x, y(0 \leq x, y \leq 10^9)$ representing coordinates. Their number is $1-n$ respectively.

The following n lines describe Roundgod's towns and each line contains two integers $x, y(0 \leq x, y \leq 10^9)$ representing coordinates. Their number is $1-m$ respectively.

There is no repeated coordinates among those $n+m$ towns. We also guarantee that no three towns are on the same straight line among them.

Output

Please output $n+m-2$ lines in total, the first $n-1$ lines representing the construction plan of Acesrc's towns and the other $m-1$ lines representing the construction plan of Roundgod's towns. For each line of a construction plan, please output two integers x, y, indicating a straight road connected town x and y.

If it is impossible to find any valid construction plan, output `Impossible` instead.

Example

standard input	standard output
2 3 0 0 1 1 1 0 0 1 2 3	2 1 1 3 3 2

Problem G. Blackjack

Input file:	*standard input*
Output file:	*standard output*
Time limit:	4 seconds
Memory limit:	256 mebibytes

As the saying goes, "if you are hesitant, you will lose the game; if you are decisive, you will die in vain."

Calabash doesn't want to lose the game, neither to die in vain. But, this is not important, because he doesn't have enough money to buy the game.

Fortunately, Roundgod, the banker of a casino, gives him a chance to fulfill his dream. Roundgod promises to buy him the game if he could win in one round of Blackjack. The Blackjack involves a deck of playing cards, each has some points on it. It is carried out as follows: Roundgod gives a number a and tells Calabash the number of points on each of the cards. Then, Calabash begins to draw cards, according to the following procedure:

1. the deck of all cards is shuffled uniformly at random;
2. Calabash could repeatedly draw a card from the deck (he immediately knows the number of the points on the card after drawing the card), or declare to stop drawing at any moment;
3. if, after drawing any card, the total number of points on cards Calabash drawn exceeds a limit b, he loses immediately;
4. otherwise, after Calabash stops drawing, he wins if and only if the total number of points on cards he drew is greater than a.

Calabash wants to know the probability to win if he plays optimally so that he can obtain his favorite game. Please help him to calculate the probability.

Input

There are two lines in the input. The first line consists of three integers n, a, b ($1 \le n \le 500, 1 \le a < b \le 500$), denoting the number of cards in the deck, the number given by Roundgod, and the limit to the total number of points, respectively.

The second line contains n integers $x_1, x_2, \cdots, x_n$ ($1 \le x_i \le 500$), denoting the numbers of points of these cards. It is guaranteed that $\sum_{i=1}^{n} x_i > b$.

Output

Output the answer within an absolute error of no more than 10^{-6}.

Examples

standard input	standard output
5 2 4 1 1 1 5 5	0.100000000000
5 2 4 1 1 1 3 5	0.450000000000

Problem I. A Math Problem

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	256 mebibytes

There are n fans $F_i(i = 1, \cdots, n)$ and m teams $T_j(j = 1, \cdots, m)$.

(i) For any fan F_i, F_i is a fan of at least one team but not a fan of all teams.

(ii) For any two teams $T_i, T_j(1 \le i, j \le m)$, there exists exactly one team $T_k(1 \le k \le m)$ exactly having the fans both T_i and T_j have. Note that i, j, k can be the same.

(iii) For any two teams $T_i, T_j(1 \le i, j \le m)$, there exists exactly one team $T_k(1 \le k \le m)$ exactly having the fans either T_i or T_j have. Note that i, j, k can be the same.

Please calculate that How many kinds of correspondences between the fans and the teams.

Input

There are multiple test cases. The first line of the input contains an integer $T(T \le 100000)$, indicating the number of test cases. For each test case:

The first and only line contains two integers $n, m(1 \le n \le 10^{18}, 2 \le m \le 6)$.

Output

For each test case, output a integer representing the answer modulo $1000000007(10^9 + 7)$ in one line.

Example

standard input	standard output
9 2 2 2 3 3 3 3 4 4 4 4 5 5 5 5 6 6 6	2 12 36 216 1032 7200 46800 453600 3369600

Problem J. Gaokao

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	256 mebibytes

Roundgod is about to attend Gaokao (National Unified Examination for Admissions to General Universities and Colleges) and his dream school is Zhejiang University. He sees an interesting problem while he is studying Math, which is a problem related to Pascal's Triangle.

The definition of Pascal's Triangle is given below:

The first element and the last element of each row in Pascal's Triangle is 1, and the m_{th} element of the n_{th} row equals to the sum of the m_{th} and the $(m-1)_{th}$ element of the $(n-1)_{th}$ row. Here's an example of a 5 levels Pascal's Triangle .

$$
\begin{array}{ccccccccc}
 & & & & 1 & & & & \\
 & & & 1 & & 1 & & & \\
 & & 1 & & 2 & & 1 & & \\
 & 1 & & 3 & & 3 & & 1 & \\
1 & & 4 & & 6 & & 4 & & 1
\end{array}
$$

In the task, Roundgod is required to calculate how many elements in the 126_{th} row of Pascal's Triangle are odd numbers.

After solving it, Roundgod thinks of a harder version of this problem. He gives you many requests about similar questions but the row number will be bigger. Please calculate that how many elements in the k_{th} row of Pascal's Triangle are odd numbers.

Input

There are multiple test cases. The first line of the input contains an integer T $(1 \le T \le 500)$, indicating the number of test cases. For each test case:

The first and only line contains an integer $K(K \le 10^{18})$, indicating the required row number in Pascal's Triangle.

Output

For each test case, output the number of odd numbers in the k_{th} line.

Example

standard input	standard output
3 3 4 5	2 4 2

Problem K. Data Structure

Input file:	*standard input*
Output file:	*standard output*
Time limit:	20 seconds
Memory limit:	512 mebibytes

Andy is a famous data structure expert at Nanjing University second to none. One day he throws a plain dry data structure problem to his friends, but none of them can solve. How about you?

Given a tree rooted at node 1. Each node has a weight which is 0 initially. Define the distance between two nodes as the number of edges in the unique simple path between the two nodes. You need to perform these two types of operations:

- Type 1: given a, x, y, z, add z to the weights of all a's descendants (including a itself) whose distances to a are y modulo x;
- Type 2: given a, return the weight of node a.

Input

The first line of the input is a single integer T ($1 \le T \le 4$), the number of test cases.

Each test cases starts with two integers n, m ($1 \le n, m \le 300000$), denoting that there are n nodes (numbered 1 through n) in the tree and you need to perform m operations. The next line contains $n-1$ integers, $f_1, f_2, \cdots, f_{n-1}$ ($1 \le f_i \le i$), specifying the edges of the trees; the ith integer denotes the parent of node $i+1$. The next m lines describe the operations. Each line is either `1 a x y z` ($1 \le a \le n, 1 \le x \le n, 0 \le y < x, 0 \le z \le 500$), denoting an operation of type 1, or `2 a` ($1 \le a \le n$), denoting an operation of type 2.

Output

For each operation of type 2 in each test case, print the answer in one line.

Example

standard input	standard output
1 5 5 1 1 2 1 1 1 5 4 1 1 1 4 1 5 1 2 1 0 4 2 3 2 1	5 0

Problem L. Landlord

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	256 mebibytes

Calabash is the servant of a landlord. The landlord owns a piece of land, which can be regarded as an infinite 2D plane.

One day the landlord set up two orthogonal rectangular-shaped fences on his land. He asked Calabash a simple problem: how many nonempty connected components is my land divided into by these two fences, both finite and infinite? Calabash couldn't answer this simple question. Please help him!

Recall that a connected component is a maximal set of points not occupied by the fences, and every two points in the set are reachable without crossing the fence.

Input

The first line of input consists of a single integer T $(1 \le T \le 10000)$, the number of test cases.

Each test case contains two lines, specifying the two rectangles. Each line contains four integers x_1, y_1, x_2, y_2 $(0 \le x_1, y_1, x_2, y_2 \le 10^9, x_1 < x_2, y_1 < y_2)$, where $(x_1, y_1), (x_2, y_2)$ are the Cartesian coordinates of two opposite vertices of the rectangular fence. The edges of the rectangles are parallel to the coordinate axes. The edges of the two rectangles may intersect, overlap, or even coincide.

Output

For each test case, print the answer as an integer in one line.

Example

standard input	standard output
3 0 0 1 1 2 2 3 4 1 0 3 2 0 1 2 3 0 0 1 1 0 0 1 1	3 4 2

Problem N. Number

Input file:	*standard input*
Output file:	*standard output*
Time limit:	7 seconds
Memory limit:	512 mebibytes

— 13.69! What an interesting number! — exclaimed Bob on looking at the cash register receipt.

— Really? — asked Alice. — What's so interesting about 1369?

— 13 squared is 169, and both numbers can be found in the digits of 1369, — explained Bob.

— Big deal! — replied Alice. — 31 squared is 961, 19 squared is 361, 6 squared is 36, 3 squared is 9, 1 squared is 1. All of those numbers are in the digits of 1369 too.

— Wow! That makes 1369 even more interesting!

Write a program to help Bob find all of the hidden pairs of numbers and their squares in a given number.

Input

Input has a list of up to 60 non-negative integers, each less than 10^{13}, one per line. There will not be any leading zeros on the numbers. The end of input is indicated by the value 0. This value should not be processed.

Output

For each number, have a line with "`Hidden squares in `" and the original number. Then have the list of numbers and their squares with digits found in the original number, in ascending order. You should format your answer as shown in the sample output below and have a blank line after the output for each number.

A number is contained in the digits of a containing number if each digit in the number is a digit in the containing number. If a digit appears multiple times in the number, it must appear at least that many times in the containing number. So 100 is contained in 1000, but 100 is not contained in 10.

Example

standard input	standard output
1369 27 10 0	Hidden squares in 1369 1 * 1 = 1 3 * 3 = 9 6 * 6 = 36 13 * 13 = 169 19 * 19 = 361 31 * 31 = 961 Hidden squares in 27 Hidden squares in 10 0 * 0 = 0 1 * 1 = 1

Problem O. Official Visit

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	512 mebibytes

A king wishes to go for a walk on a square chessboard with the following conditions:

1. Any two successive cells of the path must be adjacent, i.e., share an edge or a corner (thus, a cell may have up to eight adjacent cells).
2. Each cell must be visited exactly once; the first and the last cells of the path must coincide (thus, the first cell of the path will be actually visited twice).
3. The path must have no self intersections (if we think of a path as a closed polyline with vertices at cells' centers).

Your task is to find the maximal possible length of a king's path (here we mean the length of the polyline, not the number of king's moves).

Input

The only line of the input file contains an integer N ($1 \le N \le 300$), denoting the size of the chessboard.

Output

The only line of the output file must contain the length of the king's tour with at least three precise digits after the decimal point. The cells have side 1.

Examples

standard input	standard output
1	0.000
2	4.000
3	9.414

Problem P. Grocary

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	256 mebibytes

Your company, Calculation Solutions, which specializes in accounting software has been contacted by the local grocery for help. At the end of each day at the store, the owner, Mr. Smith has a list of transactions.

Each transaction is of the form:

1. p q: which means q items costing p dollars each were sold.
2. p: which means an item costing p dollars was sold.
3. $-p$: which means an item costing p dollars was returned.

Mr. Smith would like for you to help him by generating the revenue his store had on each day.

Input

The first line contains an integer N ($1 \le N \le 100$), the total number of days. Each day starts with an integer T ($0 \le T \le 100$) on a line, T being the total number of transactions on that day. It is followed by T lines containing transactions of type 1, 2 or 3. Since Mr. Smiths store is mid-sized, p and q are relatively small ($1 \le p, q \le 10$). The transactions are in the format: "`1 p q`", "`2 p`" or "`3 -p`" (quotes for clarity).

Output

For each of N days, output the revenue R for that day in the format `‘‘Day D: R dollars.’’`.

Example

standard input	standard output
3 2 1 4 5 2 6 1 3 -7 0	Day 1: 26 dollars. Day 2: -7 dollars. Day 3: 0 dollars.

Problem Q. Numerical Input Verification

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 Mebibytes

Your task is to create simple validator to verify numerical input.

Given a sequence of characters, check whether they describe an unsigned integer. Whitespace is allowed before the number, but the rest of the input must consist of a single, non-negative integer. Only digits will be accepted as the relevant part of this part of the input.

Input

Input file consists of one line: the input to be validated: at least one and at most 50 characters with ASCII codes between 50 and 126, inclusive.

Output

Print a line containing the value of the number (without leading zeros), if the input is valid, or print "`invalid input`" otherwise.

Examples

standard input	standard output
`34 567`	`invalid input`
`-96`	`invalid input`
`00045`	`45`
`45.3`	`invalid input`