Problem A. RSA Factorization

Input file: a.in
Output file: a.out
Time limit: 2 seconds

The positive integer n is given. It is known that $n = pq$, where p and q are primes, $p \le q$ and $|q - kp| \le 10^5$ for some given positive integer k. You must find p and q.

Input

Each line contains integers n ($1 < n < 10^{120}$) and k ($0 < k < 10^8$).

Output

For each pair of numbers n and k print in separate line the product `p * q` such that $p \le q$

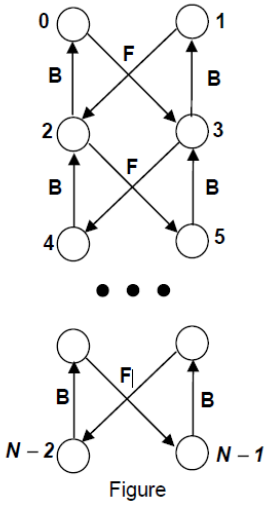
Example

a.in	a.out
35 1 121 1 1000730021 9	5 * 7 11 * 11 10007 * 100003

Problem B. Hard-working Student

Input file: b.in
Output file: b.out
Time limit: 2 seconds

Billy is a hard-working student. He is fond of computers and intends to learn as much as possible. Now he studies graph theory, and must write a program to build the graph which is shown on the Figure.



Figure

The vertices of the graph are labeled sequentially with integer keys starting from 0 to $N - 1$ ($N \le 10000$). There are two types of edges: backward edges — labeled with B in the Figure (for example from node 4 to node 2, or from node 3 to node 1), and forward edges, labeled with F in the Figure (for example from node 1 to node 2 or from node 0 to node 3). Billys program starts with an initial graph that contains the vertices 0, 1, 2, 3, and must continue to build the graph based on a sequence of commands written in a text file. A command has the following specification:

index0 string-of-characters index1

where index0 and index1 are the keys of vertices, and string-of-characters is a sequence of actions executed from right to left. An action is represented by one of the following characters:

Characte	Action
f	Follow the Forward edge if it does exist or creates it and the corresponding vertex from an argument node
b	Follow the backward edge if it does exist or creates it and corresponding vertex, starting from an argument node
k	Prints the key of the argument node
<	v[index0] = argument node
=	Prints '=' if v[index0] == node or '#' otherwise

where v is the array of the nodes of the graph. The argument of the first operation is the node v[index1]. The result of the operations f and b is a node that represents the argument for all the other operations. The operations < and = are the leftmost specified. For example, for the command `4 <kff 0` the actions are:

index0 = 4, index1 = 0

x = f(v[0]) // forward to node 3, x = 3

y = f(x) // forward creates node (4), y = 4

k(y) // prints the key (4)

v[4] = y // put node (4) in array v

A node is put in the array v only by the command ¡. Initially the array contains the nodes with keys 0, 1, 2, 3, v[0]=0, v[1]=1, v[2]=2 and v[3]=3.

Input

The program input is from a text file. The file contains the sequence of commands. White spaces can occur freely in the input. The input data terminate with an end of file.

Output

Each print must be to the output file from the beginning of a line. There are no empty lines in between.

Example

b.in	b.out
4 <kf 3 0 =bb 4 7 <ff 3	4 =

Problem C. System Engineer

Input file: c.in
Output file: c.out
Time limit: 2 seconds

Bob is a skilled system engineer. He is always facing challenging problems, and now he must solve a new one. He has to handle a set of servers with differing capabilities available to process job requests from persistent sources — jobs that need to be processed over a long or indefinite period of time. A sequence of persistent job requests arrives revealing

a subset of servers capable of servicing their request. A job is processed on a single server and a server processes only one job. Bob has to schedule the maximum number of jobs on the servers. For example, if there are 2 jobs j1, j2 and 2 servers s1, s2, job j1 requiring the server s1, and job j2 requiring also the server s1. In this case Bob can schedule only one job. Can you help him?

In the general case there are n jobs numbered from 0 to n-1, n servers numbered from n to $2n-1$, and a sequence of job requests. The problem asks to find the maximum number of jobs that can be processed.

Input

The program input is from a text file (at most 1 MB). Each data set in the file stands for a particular set of jobs. A data set starts with the number n ($n \leq 10000$) of jobs, followed by the list of required servers for each job, in the format: jobnumber: (nr-servers) $s_1 \ldots s_{nr-servers}$

Output

The program prints the maximum number of jobs that can be processed.

Example

c.in	c.out
2 0: (1) 2 1: (1) 2 1 0: (1) 1	1 1

Problem D. Cycles of Lanes

Input file:	d.in
Output file:	d.out
Time limit:	2 seconds

Each of the M lanes of the Park of Polytechnic University of Bucharest connects two of the N crossroads of the park (labeled from 1 to N). There is no pair of crossroads connected by more than one lane and it is possible to pass from each crossroad to each other crossroad by a path composed of one or more lanes. A cycle of lanes is simple when passes through each of its crossroads exactly once.

The administration of the University would like to put on the lanes pictures of the winners of Regional Collegiate Programming Contest in such way that the pictures of winners from the same university to be on the lanes of a same simple cycle. That is why the administration would like to assign the longest simple cycles of lanes to most successful universities. The problem is to find the longest cycles? Fortunately, it happens that each lane of the park is participating in no more than one simple cycle.

Input

On the first line of the input file the number T of the test cases will be given. Each test case starts with a line with the positive integers N and M, separated by interval ($4 \leq N \leq 4444$). Each of the next M lines of the test case contains the labels of one of the pairs of crossroads connected by a lane.

Output

For each of the test cases, on a single line of the output, print the length of a maximal simple cycle.

Example

d.in	d.out
1 7 8 3 4 1 4 1 3 7 1 2 7 7 5 5 6 6 2	4

Problem E. Multiprocessor Scheduling

Input file:	e.in
Output file:	e.out
Time limit:	2 seconds

There are two applications running on a multiprocessor machine. Each application i ($i = 1, 2$) consists of N procedures which are numbered from 1 to N and must be executed sequentially (in the order $1, \ldots, N$). A procedure will be identified by a pair (i, j), where $i = 1, 2$ identifies the application and $1 \leq j \leq N$ represents the index of the procedure in the sequence of procedures of the application i. A procedure (i, j) can only be executed on the processor $P(i, j)$ of the machine and its execution lasts for $D(i, j)$ seconds. We want to schedule the execution of the procedures of the two applications over the processors of the machine in such a way that the time moment when the last procedure finishes its execution (from any of the two applications) is minimum; this time moment is called makespan. We consider that the two applications are available for scheduling starting from the time moment 0. The schedule needs to obey the following rules:

- once the execution of a procedure (i, j) starts on the processor $P(i, j)$, we cannot interrupt it until the execution of the procedure ends
- we cannot execute multiple procedures on the same processor at the same time, but we can execute multiple procedures in parallel on different processors
- the execution of the procedure (i, j) ($2 \leq j \leq N$) starts either at the exact time moment tm when the procedure $(i, j-1)$ finishes its execution or at any time moment after tm
- if a procedure begins its execution at time moment tm, then it will finish its execution at the time moment $tm + D(i, j)$.

Write a program that, given the information regarding the procedures of the two applications, computes the minimum makespan.

Input

The first line of the input file contains the number T of test cases which are described next. The first line of a test case contains the number N ($1 \le N \le 300$) of procedures composing each of the two applications. Then, N lines follow, describing the first application. The j-th of these N lines contains two integers, separated by a blank: $P(1, j)$ and $D(1, j)$. After this, other N lines follow, describing the second application. The j-th of these N lines contains two integers, separated by a blank: $P(2, j)$ and $D(2, j)$. We have $1 \le P(i, j) \le 10$ and $1 \le D(i, j) \le 15000$ ($i = 1, 2$; $1 \le j \le N$). Notice that we may have $P(i, j) = P(k, l)$ – this implies that the procedures (i, j) and (k, l) cannot be executed during overlapping time intervals (notice also that if $i = k$ this would not matter, as the procedures of the same application must be executed sequentially).

Output

The output file must contain exactly T lines with a single number each – the minimum makespan for the corresponding test from the input file. These answers must be printed in the order in which the test cases are given in the input file (i.e. the i-th line of the output file contains the answer to the i-th test case from the input file).

Example

e.in	e.out
2 1 2 6 1 10 3 2 31 2 18 4 15 2 26 3 40 5 16	10 90

Problem F. Maze Stretching

Input file:	f.in
Output file:	f.out
Time limit:	2 seconds

Usually the path in a maze is calculated as the sum of steps taken from the starting point until the ending point, assuming that the distance of one step is exactly 1. Lets assume that we could stretch (shorten or extend) the maze in vertical dimension (north-south). By stretching, we are just changing the passed distance between two cells. (it becomes X instead of one). We have a two dimensional maze which has '#' for walls, 'S' in the starting cell and 'E' at the ending cell.

Due to outside conditions, we need to make the shortest path to be exactly L in size. We are not allowed to change the maze configuration, nor to make any changes in the horizontal dimension. We are only allowed to stretch the vertical dimension, and that can be done by any percentage.

Find the percentage of the stretch P, for which the shortest path of the maze will be exactly L.

Input

First line of the input contains the number of test cases. For each test case, firstly two numbers L and N are given, where L is the required length of the shortest path and N is the number of lines that are given describing the maze. The following N lines describes the maze such as that each line describes a row of the maze. (Each row length is the horizontal dimension of the maze).

The height and width of the maze are maximum 100 cells.

All the lines describing one maze are the same size.

There will always be a solution.

The result will be between 0.000 and 1000.000 inclusive

There will be no direct horizontal only path connecting 'S' and 'E'. (the result is always unique).

Output

For each test case output the percentage of the stretch in the following format:

```
Case #K: P%
```

- P should have leading zero if the number is between 0 and 1.
- P should be rounded up on 3 decimals, and always formatted on 3 decimals (with trailing zeros if needed).

Example

f.in	f.out
2 2.5 4 ##### #S # # E# ##### 21 13 ############ #S## #E# # ## # # # # # # # # ### # # # # # # # # # # ## # # # ## # # # # ### # # # # ## # # # # # ## # # # # # ############	Case #1: 50.000% Case #2: 21.053%

Problem G. Software Industry Revolution

Input file:	g.in
Output file:	g.out
Time limit:	2 seconds

Making revolutions in the software industry is not an easy task. That's why this problem is about something else.

Stanescu has just invented a new super-cool way to develop software. It is similar to writing program code, but instead of writing it, you ask some else to do it. In such way, one could create great software, without even knowing what a Turing Machine is. As you can see, this is not just software industry revolution. Really, Stanescu does not care about the software industry at all. He just wants to make money.

In order to protect the money he is going to make, he needs to pick a special password for his bank account, satisfying the following requirements:

- The password should not be too complex, so that Stanescu can remember it. The complexity of a password is the sum of the complexity of its characters and the complexity of a character is its position in the alphabet (for 'a' it is 1, for 'b' 2, and so on). For example, the complexity of the string "ala" is $1 + 12 + 1 = 14$;
- It should match a given pattern string (composed of lowercase Latin letters, '?' and '*', no longer than 1000 characters). '?' is matched by one arbitrary lowercase Latin letter, and '*' – by zero or more arbitrary lowercase Latin letters;
- It should be a sub-string of given super-password string (composed of lowercase Latin letters, no longer than 10000).

You have to write a program that computes the complexity of simplest possible password.

Input

Several test cases are given at the input. Each of them consists of a single line containing the pattern and the super-password strings separated by a white space.

Output

For each test case, your program should print a single line with one integer – the complexity of the simplest possible password. If no password satisfies the given requirements, the program should print -1.

Example

g.in	g.out
a?a alabala a*c?a axcbaabcbax	4 9

Problem H. The Lucky Numbers

Input file:	h.in
Output file:	h.out
Time limit:	2 seconds

John and Brus are freshmen at the primary school. Their first home task is to learn some integer numbers. It is not so hard for the guys and they decide to impress their teacher by learning all lucky numbers between A and B, inclusive. As you already know from the previous year contest 4 and 7 are lucky digits, and all the other digits are not lucky. A lucky number is a number that contains only lucky digits in decimal notation.

After learning all lucky numbers in [A, B] range John and Brus still have some free time and now they decide to learn additionally each lucky number N that is out of [A, B] range, but the reversed number of N is in this range. Here reversed number of N is the number N written in decimal notation, but the order of digits is reversed. For example, the reversed number of 447 is 744 and reversed number of 774474444 is 444474477.

You are given integers A and B and your task is to find the total number of lucky numbers learned by John and Brus.

Input

The first line contains single integer T – the number of test cases. Each test case consists of a single line containing two integers A and B separated by a single space.

$1 \le T \le 74$,

$1 \le A \le B \le 10000000000000000000000000000000000000000000000000$.

Output

For each test case print a single line containing the total number of lucky numbers learned by John and Brus.

Example

h.in	h.out
2 1 100 44 47	6 3

Problem I. The Robbery

Input file:	i.in
Output file:	i.out
Time limit:	2 seconds

In the downtown of Bucharest there is a very big bank with a very big vault. Inside the vault there are N very big boxes numbered from 1 to N. Inside the box with number k there are k very big diamonds, each of weight W_k and cost C_k.

John and Brus are inside the vault at the moment. They would like to steal everything, but unfortunately they are able to carry diamonds with the total weight not exceeding M.

Your task is to help John and Brus to choose diamonds with the total weight less than or equal to M and the maximal possible total cost.

Input

The first line contains single integer T – the number of test cases. Each test case starts with a line containing two integers N and M separated by a single space. The next line contains N integers W_k separated by single spaces. The following line contains N integers C_k separated by single spaces.

$1 \le T \le 74$,

$1 \le N \le 15$,

$1 \le M \le 1000000000$,

$1 \le W_k, C_k \le 1000000000$.

Output

For each test case print a single line containing the maximal possible total cost of diamonds.

Example

i.in	i.out
2 2 4 3 2 5 3 3 100 4 7 1 5 9 2	6 29

Problem J. The Computer Game

Input file: j.in
Output file: j.out
Time limit: 2 seconds

John and Brus are playing a military strategic game on a PC. The game is played on the flat world map. At the beginning of the game Brus places his army. Then John has to choose strategic points for his army according to the following rules:

- each strategic point must be a lattice point (x, y) (a lattice point is a point with integer coordinates) such that $|x| + |y| < N$;
- John can choose any positive number of strategic points;
- all the strategic points must be distinct;
- each of the strategic points must be free (i.e. not occupied by Bruss army);
- each pair of different strategic points must be connected (possibly via some other strategic points).

Here two different lattice points $(x1, y1)$ and $(x2, y2)$ are connected if $|x_1 - x_2| + |y_1 - y_2| = 1$.

If A, B and C are strategic points, A and B are connected, B and C are connected, then A and C are also connected.

Your task is to find the number of ways for John to choose strategic points for his army.

Input

The first line contains single integer T – the number of test cases. Each test case starts with a line containing two integers N and M separated by a single space. N is the number mentioned in the first rule. M is the number of lattice points on the world map already occupied by Bruss army. Each of the following M lines contains two integers X_k and Y_k separated by a single space. Each lattice point (X_k, Y_k) is occupied by Bruss army.

$1 \le T \le 74$, $1 \le N \le 7$, $1 \le M \le 225$, $-7 \le X_k, Y_k \le 7$, all (X_k, Y_k) will be distinct.

Output

For each test case print a single line containing the number of ways for John to choose strategic points for his army.

Example

j.in	j.out
2 2 1 7 7 2 3 0 0 4 -7 7 -4	20 4

Problem K. The Bad Number

Input file: k.in
Output file: k.out
Time limit: 2 seconds

John and Brus believe that number N is a very bad number. Thus they try to avoid it every time and everywhere.

Now the guys would like to represent number M as a sum of positive numbers, each of which not exceeding K. But dont forget about the bad number N! Each summand must not be divisible by N, moreover the number of summands also must not be divisible by N.

Your task is to find the minimal possible number of summands in such representation of M.

For example, if $N = 3$, $M = 11$, $K = 6$ then we can represent M as $5 + 6$, but as far as 6 is divisible by 3 we must have at least 3 summands. But as far as $N = 3$ we cant have 3 summands and thus the answer is 4. One of the possible ways to represent M is $11 = 4 + 4 + 2 + 1$.

Input

The first line contains single integer T – the number of test cases. Each test case consists of a single line containing three integers N, M and K separated by single spaces.

$1 \le T \le 74$, $1 \le N, M, K \le 1000000000$.

Output

For each test case print a single line containing the minimal possible number of summands according to the requirements described above. If it is impossible to do this output "−1" (quotes for clarity) instead.

Example

k.in	k.out
2 3 11 6 2 12 47	4 -1