Maratón de Programación 2019

SILUX

Puedes participar usando cualquiera de estos Lenguajes de Programación

JAVA

Python

C

C++

The key is not the will to win...everybody has that. It is the will to prepareto win that is important. *Bobby Knight*

Universidad Francisco de Paula Santander
Vigilada Mineducación

Programa de Ingeniería de Sistemas
Acreditado de Alta Calidad
"Educación y Tecnología con Compromiso Social"

RED DE PROGRAMACIÓN COMPETITIVA RPC

Contents

1 Maratón de Programación UFPS 2019

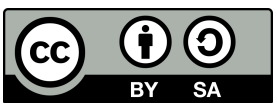
Desde el año 2015, el programa Ingeniería de Sistemas de la Universidad Francisco de Paula Santander (UFPS) inició un interesante proceso para promover la Programación Competitiva, como parte de las actividades del Semillero SILUX (Linux, Software Libre y Licencias Abiertas). El propósito fundamental fue el desarrollo de competencias de resolución de problemas, programación de computadores, trabajo colaborativo y liderazgo.

Los pioneros de dicho proyecto fueron dos estudiantes, quienes ya se graduaron y dejaron como herencia una gran motivación. Ahora siguen colaborando con el Semillero que es liderado directamente por los estudiantes, quienes ingresan desde primer semestre con el entusiasmo que trae el sueño de llegar algún día a la Competencia Mundial ICPC (The International Collegiate Programming Contest https://icpc.baylor.edu/).

Desde entonces, cada año, el evento más importante es la Maratón Interna de Programación de la UFPS, que propone un conjunto de retos para poner a prueba las habilidades en algoritmia, programación y trabajo en grupo de los estudiantes. En dicho evento un actor fundamental siempre ha sido la Red de Programación Competitiva (RPC), quien apoya toda la logística de preparación de la competencia y además ofrece su plataforma tecnológica y su equipo de trabajo. El lema de RPC es "Aquí crecemos juntos" y la UFPS ya lleva cinco (5) años creciendo en Programación Competitiva junto a muchas universidades en varios países de toda Latinoamerica.

Para éste año 2019 nos complace presentar un conjunto de trece (13) problemas inéditos, los cuales fueron escritos por varias personas que se unieron voluntariamente. Tenemos presencia internacional y nacional de estudiantes que ya tuvieron la oportunidad de llegar a la Competencia Mundial ICPC. Además, logramos articular el trabajo de manera remota entre Cúcuta, Pereira, Bogotá, Florencia, Cali y la Habana Cuba:

- Milton Jesús Vera Contreras - UFPS
- Gerson Yesid Lázaro - UFPS
- Angie Melissa Delgado - UFPS
- Hugo Humberto Morales - UTP Pereira
- Eloy Pérez Torres - Universidad de la Habana Cuba
- Manuel José Hernández Jiménez - UniAmazonía Colombia
- Wilmer Emiro Castrillón Calderón - UniAmazonía Colombia
- Juan Camilo Bages - Uniandes Bogotá



2 Grupo de Estudio en Programación Competitiva

El grupo de estudio en Programación Competitiva hace parte del Semillero de Investigación SILUX (Linux, Software Libre y Licencias Abiertas) y tiene como fin preparar y fortalecer a los estudiantes de Ingeniería de Sistemas de la Universidad Francisco de Paula Santander en competencias de resolución de problemas, algoritmia, programación de computadores, trabajo colaborativo y liderazgo. Se aprovecha el entorno competitivo o gamificación porque favorece el aprendizaje tanto de habiidades hard como habilidades soft.

Los integrantes del grupo de estudio han participado en la Maratón Nacional de Programación desde el año 2014, logrando por 5 años consecutivos la clasificación a fase Regional Latinoamericana.

3 Instrucciones

Puedes utilizar Java, C, C++ o Python, teniendo en cuenta:

1. Resuelve cada problema en un único archivo. Debes enviar a la plataforma únicamente el archivo .java, .c, .cpp, o .py que contiene la solución.
2. Todas las entradas y salidas deben ser leídas y escritas mediante la entrada estándar (Java: Scanner o BufferedReader) y salida estándar (Java: System.out o PrintWriter).
3. En java, el archivo con la solución debe llamarse tal como se indica en la linea "Source file name" que aparece debajo del título de cada ejercicio. En los otros lenguajes es opcional (puede tener cualquier nombre).
4. En java, la clase principal debe llamarse igual al archivo (Si el Source File Name indica que el archivo debe llamarse example.java, la clase principal debe llamarse example).
5. En java, asegúrate de borrar la linea "package" antes de enviar.
6. Tu código debe leer la entrada tal cual se indica en el problema, e imprimir las respuestas de la misma forma. No imprimas lineas adicionales del tipo "La respuesta es..." si el problema no lo solicita explicitamente.
7. Si su solución es correcta, en unos momentos la plataforma se lo indicará con el texto "YES". En caso contrario, obtendrá un mensaje de error.

4 Reglas

1. Gana el equipo que resuelve mas problemas. Entre dos equipos que resuelvan el mismo número de problemas, gana el que los haya resuelto en menos tiempo (ver numeral 2).
2. El tiempo obtenido por un equipo es la suma de los tiempos transcurrido en minutos desde el inicio de la competencia hasta el momento en que se envió cada solución correcta. Habrá una penalización de 20 minutos que se suman al tiempo por cada envío erróneo (esta penalización solo se cuenta si al final el problema fue resuelto).
3. NO se permite el uso de internet durante la competencia. Únicamente se puede acceder a la plataforma en la cual se lleva a cabo la competencia.
4. NO se permite el uso de dispositivos electrónicos durante la competencia.

5. NO se permite la comunicación entre miembros de equipos diferentes. Cada integrante solo puede comunicarse con sus dos compañeros de equipo.

6. Se permite todo tipo de material impreso (libros, fotocopias, apuntes, cuadernos, guías) que el equipo desee utilizar.

5 Lista de Problemas

A continuación la lista de los trece (13) problemas a resolver. Un primer reto y logro es resolver alguno de los problemas. Un segundo reto es lograr resolver cualquier de los problemas más rápido que todos los demás. Y un tercer reto es lograr resolver todos los problemas.

Problem A. A problem with the longest title and the shortest description of the whole contest

Source file name:	alongestitle.c, alongestitle.cpp, alongestitle.java, alongestitle.py
Input:	`standard`
Output:	`standard`
Author(s):	`Gerson Yesid Lázaro - UFPS Colombia`

We all hate very long problems. Therefore, this is a short problem: Given the Fibonacci sequence:

1. $f(0) = 0$
2. $f(1) = 1$
3. $f(n) = f(n-1) + f(n-2)$ for all $n \geq 2$, $n \in \mathbb{N}$

Input

The input consists of multiple test cases. Each case contains two integers K y M ($1 \leq K \leq 10^{10}$; $1 \leq M \leq 60$).

Output

For each case, print a line with an integer, the index of the $K - th$ Fibonacci number that is a multiple of M.

Example

Input	Output
1 4 1 6 2 8	6 12 12

Explanation

In the Fibonacci sequence:

1. The first multiple number of 4 is $f(6) = 8$.
2. The first multiple number of 6 is $f(12) = 144$.
3. The second multiple number of 8 es $f(12) = 144$, the first was $f(6) = 8$.

Problem B. Billar Pool

Source file name:	billarpool.c, billarpool.cpp, billarpool.java, billarpool.py
Input:	`standard`
Output:	`standard`
Author(s):	`Manuel José Hernández Jiménez - UniAmazonía Colombia`

Tomas is a lover of mathematics. One day a co-worker extended an invitation to the famous game of precision pool billiards, this game consists of introducing balls in holes located on the edges of the board, are 15 balls and are listed from 1 to 15, the winner of the game is the one that reaches the highest total sum of the balls he has introduce. Tomas likes mathematics a lot and decided to have fun counting as many possible ways as he can get a P score, but with 15 balls he got bored quickly, so he decided to play with N billiard balls labeled from 1 to N, and he wants to calculate how many Different ways you can achieve a given score, for example:

The score 5 with 15 balls can be obtained in 3 different ways:

1. 5
2. $1, 4$
3. $3, 2$

Tomas is very busy counting possible moves and therefore requires your help to automate the process of verification of the result.

Input

The first line contains an integer Q $(0 < Q < 450)$, which indicates the number of test cases. The next Q lines contains two integers N $(0 < N \leq 200)$ y P $(0 < P \leq N * (N + 1)/2)$, which indicate the amount of balls and the score on which you want to know the game.

Output

Given N balls numbered from 1 to N indicate how many forms possible a score P can be reached, as the answer can be very large so print with a module of $10^9 + 7$.

Example

Input	Output
6 15 1 5 5 20 10 15 20 15 60 15 120	1 3 10 57 722 1

Problem C. Card Game

Source file name:	cardgame.c, cardgame.cpp, cardgame.java, cardgame.py
Input:	`standard`
Output:	`standard`
Author(s):	`Wilmer Emiro Castrillón Calderón - UniAmazonía Colombia`

Pepito and his friends are university students, and every time they have a group work they decide to divide the work into parts and finally join each part, but always everyone wants to select the easiest part of the work (clearly only one can take it), So they decided to do a simulation of a small card game, which whoever wins it may be the first to select the part of the job he wants to do (clearly he will take the easiest one).

The game consists of the following, take N listed cards from 1 to N, put them in an ordered stack with card 1 above and card N below and perform the following process: first the card that is in the middle (in the position *size* of the *stack*/2 rounding up) is removed, and second take the card below and put above of the stack. This process is repeated until there is only one card left, this is the winner card. At the beginning all the members of the group select a card, and the winner is the one who chooses the winning card, if none chose it then the winner is the one who has the closest card to the winner and also has a larger number. Then the winning student can do the easiest part of the job.

Pepito knows your skills and has asked for your help, he is a very busy person (the busiest of his team) and would like to know the winning card for a game of size N, can you help him?

Input

The entry consists of multiple test cases, each case contains an integer N ($1 < N \leq 2^{58}$) which represents the size of the game, the entry ends with EOF.

Output

For each test case, print on a line the number of the winning card for a game of size N.

Example

Input	Output
3	1
5	4
7	3
4543	2019
123456789	5683375

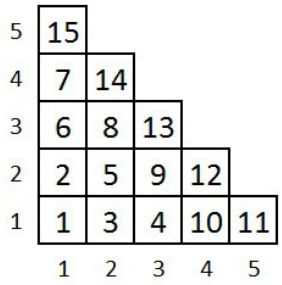
Problem D. Dora the Explorer II

Source file name:	dora.c, dora.cpp, dora.java, dora.py
Input:	`standard`
Output:	`standard`
Author(s):	`Hugo Humberto Morales Peña - UTP Colombia`

One day, there was this little ant called "Dora the Explorer", the ant arrived to a triangular board of d diagonals. the ant wanted to explore every boxes of the board, so she began to walking on this board starting on the diagonal containing only one box.

Dora the explorer began on the box $(1,\ 1)$. In the first place she took one step forward to the box $(1,\ 2)$, then she descended by the diagonal to reach the box $(2,\ 1)$, after that, the ant took one step to the right to the box $(3,\ 1)$, then she ascended by the respective diagonal passing trough the boxes $(2,\ 2)$ and $(1,\ 3)$. Each time the ant added a new diagonal to her trajectory either ascending or descending by the diagonal depending of the first box she reached in the respective diagonal.

For example, the ant in her first 15 steps made the following trajectory in the triangular board, where the number of each box or cell indicates the order in which it was visited:



The 10th step was in the box $(4,\ 1)$, while the 15th step was in the box $(1,\ 5)$.

The objective is to determine the number of steps that Dora the Explorer must make in her trajectory to reach the box $(x,\ y)$ of the triangular board. You will suppose that to reach the box $(1,\ 1)$ is necessary one step, this box is the origin on the triangular board. Additionally, you will be assumed that the triangular board is big enough to admit the coordinates of any box.

Input

The input contains several test cases. Each line of the input contains two space-separate positive integer numbers $x\ y$ $(1 \leq x,\ y \leq 2 \times 10^9)$, denoting the coordinates of the box with the column and row values, respectively. The input ends with a line containing 0 0, this case must not be processed.

Output

For each test case, print a line with a positive integer number n indicating the number of steps that Dora the Explorer must make in her trajectory to reach the box.

Example

Input	Output
1 1 1 3 4 1 5 1 1 5 1 6 2 5 0 0	1 6 10 11 15 16 17

Use fast I/O methods

Problem E. Brute Force Hacking

Source file name:	ebrute.c, ebrute.cpp, ebrute.java, ebrute.py
Input:	`standard`
Output:	`standard`
Author(s):	`Juan Camilo Bages - Universidad de Los Andes Colombia`

Disclaimer: The following is just a fictional scenario and should not be replicated. Also, breaking into systems is not that easy ;).

Training for programming competitions can be tough and very time consuming. Because of this, some people usually have troubles keeping up with their university classes. You are one of these cases and you have decided to make some sketchy move. You are planning to get into the university grade system and give yourself a small boost in each one of your courses. In order to do this, you need to get into the university's server by breaking their complex security system. The security method they employ consists of the following steps:

1. First, the system generates a random string S with characters ranging from "0" to "4".
2. Then, the user trying to access the system will remove some characters from S at very specific positions that only university's professors know. This will result in a (possibly empty) string subset of S.
3. Finally, the user will insert the resulting string subset and be granted access.

After making a deep analysis of the system, you have realized that there is no obvious weakness in it. Now, the only choice remaining is to try every possible string subset of S as one of these must be the correct one. This is a tedious job and before starting you want to know how many unique string subsets of S there can be. Two string subsets C $(c_1, c_2...c_{|C|})$ and D $(d_1, d_2...d_{|D|})$ are considered the same if they have the same length, and for every $0 \leq i \leq |C|, c_i = d_i$.

Input

The input consists of multiple test cases. Each test case contains a single string S $(1 \leq |S| \leq 24)$ with characters ranging from "0" to "4".

Output

For each case, print a line with an integer n indicating the number of unique string subsets that can be made from string S.

Example

Input	Output
000 0123	4 16

Explanation

In the first sample, the unique string subsets are "" (empty), "0", "00", and "000". In the second sample, there are $2^4 = 16$ string subsets and all of them are unique.

Problem F. Fake Problem

Source file name:	fakeproblem.c, fakeproblem.cpp, fakeproblem.java, fakeproblem.py
Input:	`standard`
Output:	`standard`
Author(s):	`Milton Jesús Vera Contreras - UFPS Colombia`

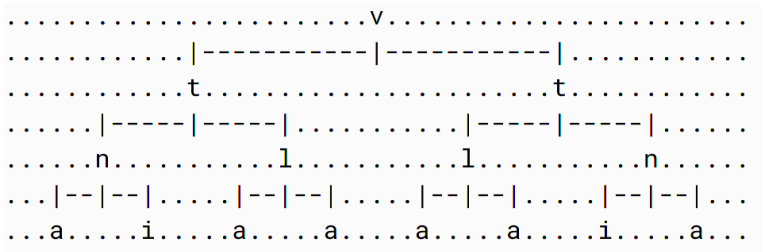
The fake news consist of deliberate disinformation or hoaxes spread via social media. Nowadays the fake news are a worldwide trend used by famous people with tag #FakeNews. Before the #FakeNews tag the teachers already had created #FakeProblem tag but neither the teachers nor tag are famous.

The Competitive Programming Network (RPC) is famous in Latin America, so its teachers want to become famous with the trend #FakeProblem, but they need your help with this fake problem. Could you resolve this #FakeProblem?

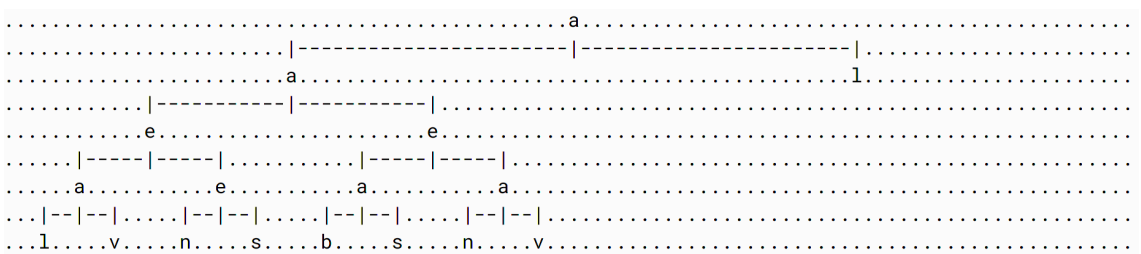
A binary tree can be traversed in six different ways, but only three ways are used (and are taught):

1. Preorder (Root, Left, Right)
2. Inorder (Left, Root, Right)
3. Postorder (Left, Right, Root)
4. Preorder Reverse (Right, Left, Root)
5. Inorder Reverse (Right, Root, Left)
6. Postorder Reverse (Root, Right, Left)

If you construct a binary tree with a palindrome string in which the first leaf to left of the tree corresponds to the first character of the palindrome then Inorder and Inorder Reverse are equals. For example, in spanish is very popular the palindrome "Anita lava la tina", if you ignore spaces and convert to lowercase then its binary tree is (Note: The character "." indicate empty space):



The *string* "¡Laven esa base naval!" is not palinidrome but if you assume that it is and if you ignore spaces and punctuation characters and convert to lowercase then its binary tree is (Note: The character "." indicate empty space):



The fake problem is construct and print a binary tree with an arbitrary string assuming that is palindrome, that is why it is #FakeProblem.

Input

The input consists of multiple test cases. Each case contains one integer n ($0 \leq n \leq 2^{10}$) the *length* of *string* and after the *string*. the entry ends with number 0. The string will only contain uppercase letters [A-Z], lower case letters [a-z], digits [0-9], spaces " " and punctuation marks ".", ",", ";", "¡", "!", "¿", "?" and ":". The *string* is not always palindrome but it must be assumed that it is, that is why it is a #FakeProblem.

Output

Yo must print a binary tree using only lowercase letters and digits and the ASCII characters space (ASCII 32), minus '-' (ASCII 45) and pipe '|' (ASCII 124). You must ignore spaces and punctuation characters from input and convert the letters to lowercase. Each node of the tree will contain only one digit or lowercase letter.

Example

Input	Output
5 ¡Oso! 5 ¿Usa? 19 Anita lava la tina. 17 Yo hago yoga hoy. 11 Ojos Rojos. 0	......s..... ...|--|--|... ...o.....o... ......s..... ...|--|--|... ...u.....a... .....v..... .....|-----|-----|..... .....t.....t..... .....|-----|-----|.....|-----|-----|..... .....n.....l.....l.....n..... ...|--|--|.....|--|--|.....|--|--|.....|--|--|... ...a.....i.....a.....a.....a.....a.....i.....a... .....o..... .....|-----|-----|..... .....a.....o..... .....|-----|-----|.....|-----|-----|..... .....o.....o.....a.....y..... ...|--|--|.....|--|--|.....|--|--|..... ...y.....h.....g.....y.....g.....h..... .....o..... .....|-----|-----|..... .....s.....s..... .....|-----|-----|..... .....j.....o..... ...|--|--|.....|--|--|..... ...o.....o.....r.....j..... Notes: The character "." indicate empty space (ASCII 32).

Problem G. Goldbach

Source file name:	goldbach.c, goldbach.cpp, goldbach.java, goldbach.py
Input:	`Standard`
Output:	`Standard`
Author(s):	`Gabriel Gutierrez Tamayo - UTP Colombia`

Goldbach's conjecture is one of the oldest and best-known unsolved problems in number theory and all of mathematics. It states: "Every even integer greater than 2 can be expressed as the sum of two primes". This conjecture has been checked by computers for all even numbers smaller than 4×10^{18}.

The task consists in, given an integer n, calculate the minimum amount of prime numbers whose sum is equal to n. You can add several times the same prime number.

Input

The input has several test cases and ends with EOF. Each test case consists of a single line containing a positive integer n ($2 \leq n \leq 10^{12}$).

Output

For each test case, your program must print a single line with one positive integer denoting the minimum amount of prime numbers, whose sum is equal to n.

Example

Input	Output
2	1
3	1
4	2
5	1
6	2
20	2
25	2
27	3

Problem H. How Many Sub Sets

Source file name:	subsets.c, subsets.cpp, subsets.java, subsets.py
Input:	`standard`
Output:	`standard`
Author(s):	`Hugo Humberto Morales Peña - UTP Colombia`

We have a set A of positive integers with cardinality N ($|A| = N$, remember that the cardinality of a set is the total of different elements it has). You want to find out how many subsets of size two have a sum of their elements less than or equal to S.

For example, if you have the following set $A = \{6,\ 5,\ 1,\ 4,\ 2,\ 3\}$ and $S = 7$, then there is a total of 9 subsets of size two whose sum of its elements is less than or equal to 7, said subsets are: $\{6,\ 1\}$, $\{5,\ 1\}$, $\{5,\ 2\}$, $\{1,\ 4\}$, $\{1,\ 2\}$, $\{1,\ 3\}$, $\{4,\ 2\}$, $\{4,\ 3\}$ and $\{2,\ 3\}$.

Your mission, if you decide to accept it, is to count the total of subsets of size two that have a sum of their elements less than or equal to S.

Input

The input of the problem consists of a single test case. This begins with a line that contains two positive integers n y q ($3 \leq n \leq 10^6$, $1 \leq q \leq 10^2$), which represent respectively the cardinality of the set A and the total of queries that will be made on the set A. Then n lines are presented each of them containing a single positive integer x ($1 \leq x \leq 10^8$), it is obviously guaranteed that the n elements of the set A are different. Finally, q lines, each of them containing a single positive integer S ($1 \leq S \leq 2 \times 10^8$).

Output

Your program should print q lines, each of them containing a single value that represents the total result of subsets of size two that the sum of their elements is less or equal to S.

Example

Input	Output
6 3 6 5 1 4 2 3 7 8 12	9 11 15

Use fast I/O methods

Problem I. Writing Ones

Source file name:	iwritingones.c, iwritingones.cpp, iwritingones.java, iwritingones.py
Input:	`standard`
Output:	`standard`
Author(s):	`Eloy Pérez Torres - Universidad de la Habana Cuba`

Manuel is discovering the amazing world of mathematics, then we see him writing numbers on the board all day. These numbers have a curious property: they are only formed by the digit one (1). For example, 123 is an invalid number but 1, 11 and 111 are valid numbers.

There is no reason to fear Manuel, he is only solving a problem. At the beginning of the day the board is blank and Manuel chooses a positive integer M, then begins to write valid numbers from lowest to highest until he finds the first number that is divisible by M. He understands that "errare humanum est" and therefore he needs your help in order to verify his data. You must print the how many digits has the number found by Manuel or -1 if there is no such number.

Input

The input consists of multiple test cases. Each case contains one integer $1 \leq M \leq 10^{10}$.

Output

For each case, print a line with the number of digits of the lowest positive integer formed by only digits one (1) that is divisible by M. If there is no such number then print -1.

Example

Input	Output
3 7	3 6

Explanation

1. The lowest positive integer formed by only digits one (1) that is divisible by 3 is 111 and it has 3 digits.
2. The lowest positive integer formed by only digits one (1) that is divisible by 7 is 111111 and it has 6 digits.

Problem J. The lady of the towers

Source file name:	towers.c, towers.cpp, towers.java, towers.py
Input:	`standard`
Output:	`standard`
Author(s):	`Angie Melissa Delgado León - UFPS Colombia`

Little lady Denna is very excited, her dear uncle brought her a new towers game from his last travel. The game has a board composed for m contiguous cells numbered from 1 to m, and a set of little towers, each tower with a height h associated. In each game, her uncle will choose a set of n towers ($n \leq m$) and will put every tower in some cell x_1, x_2, ..., x_n of the board. Each tower placed in the board occupies a single cell. After that, the fun begins cause little lady Denna must destroy the towers. When she destroys a tower, the tower can fall to the left (occupying the cells $[xi - hi, xi]$) or to the right (occupying the cells $[x_i, x_i + h_i]$). Denna can only destroy a tower if the cells to be occupied by the destroyed tower doesn't contain any other tower, or if the place where the tower will fall is out of the board.

Given any distribution of towers in the board, little lady Denna wants to destroy as many towers as possible. Can you help her?

Input

Input consist of many test cases, the first line of the input contains an integer t, the number of test cases. Each test case begins with a line with two integers, the size of the board m ($1 \leq m \leq 10^9$) and an integer n ($1 \leq n \leq 10^5$ and $n \leq m$) denoting the number of towers that were placed in the board. Next n lines contains pairs of numbers x_i, h_i ($1 \leq x_i, h_i \leq 10^9$ and $x_i \leq m$) denoting the position of the tower in the board and its height.

The information of the towers is given in ascending order according with it position x on the board. There are not two towers in the same cell of the board.

Output

For each test case, print the maximum number of towers that little lady Denna could destroy according with the game rules.

Example

Input	Output
3 28 4 10 4 15 1 19 3 20 1 1000000000 1 1000000000 1000000000 30 5 1 2 2 1 5 10 10 9 20 1	4 1 4

Problem K. Kilmes

Source file name:	kilmes.c, kilmes.cpp, kilmes.java, kilmes.py
Input:	`standard`
Output:	`standard`
Author(s):	`Milton Jesús Vera Contreras - UFPS Colombia`

The last winter in Argentina a teacher and his dear students learned so much about Competitive Programming and Argentina's history, culture, people, pizza, steak and beer. For example, their learned about Kilmes's history, although they already knew about Quilmes beer: Kilmes were a native tribe who lived in the surroundings of Tucumán - Argentina. In the 17th century Kilmes were defeated by the Spanish invaders and they were forced to settle in a colony near Buenos Aires. Popular wisdom says that the story is written by the winners and that is why the Spanish invaders changed the word Kilmes for Quilmes. This is the origin the name of the famous beer.

After learn about Quilmes beer, the students searched for a bar in order to drink beer and get drunk. Their teacher warned them not to do it, but they had nostalgia for their saturdays drinking "Bahía" beer in Gamboales (Gamboales is a bar frequented by students and "Bahía" is a cheap beer). While the students had drinking "Bahía" beer, their teacher had walking through Corrientes Avenue, buying books, visiting the Museum Carlos Gardel and the Recoleta.

The students proposed an interesting challenge with beer: David, the most drinker of all students, proposed drink a beer just before to try a Competitive Programming problem. Carlos, the second most drinker of all students, proposed drink a beer just after to resolve a Competitive Programming problem. Manuel, moderate drinker, proposed the next rules:

1. When the contest starts everyone drinks a beer.
2. The person who tries a problem drinks a third (1/3) of everything that person has drunk during the contest.

Crysel usually drinks only soda and Brayan and William only drink milk. They decided not to participate because they do not drink beer, but they established the definitive rules because they were the majority.

1. When the contest starts everyone drinks a beer.
2. The person who solves a problem drinks a third (1/3) of everything that person has drunk during the contest.

Example:

1. David, Carlos and Manuel drink one beer and start the contest.
2. David solves a problem and drink (1/3) of beer.
3. Manuel solves a problem and drink (1/3) of beer.
4. David solves a problem and drink (4/9) of beer.
5. David drank 16/9 of beer, Manuel drank 4/3 beer and Carlos drank 1 beer.

The teacher wants to know the total number of beer that his unruly students drank. He knows that they solve several problems and are very drinkers so he is interested in the number of digits of the number (decimales are ignored). Could you help him?

Input

The input consists of multiple test cases. Each test case contains a single string S ($1 \leq |S| \leq 10^4$) with characters "D" or "C" or "M". A "D" character means that David solved a problem, a "M" character means that Manuel solved a problem and a "C" character means that Carlos solved a problem.

Output

The output are four integer numbers: the first number is the number of digits of the total number of beer that David drank, the second number is the number of digits of the total number of beer that Manuel drank, the third number is the number of digits of the total number of beer that Carlos drank and the fourth number is the number of digits of the total number of beer that the three of them drank (decimals are ignored).

Example

Input	Output
DM	1 1 1 1
DDDDDDDDDMMMMMMMMCCCCCCCCC	2 2 2 2
DDDDDDDDDDDDDDDDDMMMMMMMMCC	3 2 1 3

Remark: This problem express our gratefulness with many wonderful people: Leopoldo Taravilse (qepd), Melissa Delgado, Gerson Lázaro, Agustín Santiago, Matías Hunicken, Pablo Zimmermann, Mariano Crosetti, Ariel Zylber, Melanie Sclar, Nicolas Alvarez, the students of Tucuman and Rosario, all teachers and students of Training Camp Argentina, all teachers and students of Training Camp Cuba, all teachers and students of Training Camp Chile, all teacher and Students of Red de Programación Competitiva (RPC), all teacher and Students of Colombian Collegiate Programming League (CCPL), and our dear UFPS's students. You have to live sowing, always sowing.

Problem L. Luna and the hopeless progression

Source file name:	luna.c, luna.cpp, luna.java, luna.py
Input:	`standard`
Output:	`standard`
Author(s):	`Angie Melissa Delgado León - UFPS Colombia`

Luna is such a dreamy and curious girl that recently has been fascinated with numbers, especially, with those that she calls fortune numbers. A fortune number f of base l, n is the one that can be represented as $f = (l)^u + (n)^a$, where u and a are integer numbers greater or equal than zero.

Given the base $l = 2$ and $n = 4$, the numbers 9 and 20 are two examples of fortune numbers, 9 can be represented as $(2)^3 + (4)^0$ and 20 can be represented as $(2)^2 + (4)^2$. The number 13 is not a fortunate number for the given base.

Given a sequence of consecutive numbers, Luna named the interval of numbers without fortunate numbers a hopeless progression. Now given the base (l, n) for a fortunate number and a sequence of numbers from a to b, Luna wants to know the length of the maximum hopeless progression in the sequence. If all the numbers in the sequence $[a, b]$ are fortunate numbers, then the answer is 0.

Input

The first line of the input contains an integer t, the number of test cases. Each case consists of 4 integers numbers l, n, a and b ($2 \leq x, y \leq 10^{18}$, $1 \leq l \leq r \leq 10^{18}$).

Output

For each test case, print the length of the greatest hopeless progression in the interval $[a, b]$.

Example

Input	Output
3 3 2 6 7 2 3 3 5 5 3 10 22	1 0 8

Problem M. Alejandro and the meteor shower

Source file name:	meteorshower.c, meteorshower.cpp, meteorshower.java, meteorshower.py
Input:	`standard`
Output:	`standard`
Author(s):	`Gerson Yesid Lázaro - UFPS Colombia`

In addition to programming, Alejandro loves astronomy and waited for this day for a long time: In the afternoon he competes in the UFPS programming contest and at night he will see a meteor shower. Alejandro sees the sky as a rectangle, with meteors that always appear on one of the edges of the rectangle, cross the sky in some direction and disappear on a different edge - the stars always enter and leave the sky through some of its edges, never by a vertex or an interior point of the rectangle -. Alejandro believes that the happiest moment of the night will be when he can see the largest number of meteors in the sky at the same time.

All meteors move in a straight line at constant speed but in different directions, at different speeds, they appear and disappear at different times. Knowing the points and times in which each meteor appears and disappears in the sky, can you tell how many stars there will be in the happiest moment of the night? Keep in mind that if at any given moment there is a meteor appearing in the sky and another disappearing, at this moment both meteors are visible to Alejandro and if that is the happiest moment of the night, both stars will be counted. The sky and the points are indicated in Cartesian coordinates.

Input

The first line of the input contains an integer T ($1 \leq T \leq 300$), the number of test cases. Each test case starts with a line containing 2 integers Fx and Fy ($0 < Fx, Fy \leq 10^6$) representing the upper right corner of the rectangle representing the sky (the lower left corner will always be (0,0)). Next, a line appears with an integer N ($1 \leq N \leq 10^3$), the number of meteors in the meteor shower, then N lines, where each line i represents the path of a meteor through 6 integers: Sxs, Sys, Ts, Sxe, Sye, Te ($0 \leq Sxs$, $Sxe \leq Fx$; $0 \leq Sys$, $Sye \leq Fy$; $0 \leq Te$, $Ts \leq 10^3$; $Te < Ts$), representing respectively the coordinates (X, Y) by which the meteor appeared in the sky, the time in which it appeared in the sky, the coordinates (X, Y) by which the meteor disappeared from the sky and the time in which it disappeared from heaven.

Output

For each test case, print a line indicating the number of meteors that Alejandro will see at the happiest time of the night. If two or more meteors coincide in the same point at that moment, all of them must be counted.

Example

Input	Output
2 10 10 3 0 1 2 1 10 3 0 5 4 10 5 5 5 0 6 5 10 7 10 10 3 0 1 2 1 10 5 0 5 3 10 5 5 5 0 5 5 10 6	1 3