A. STOCK MANAGEMENT SOFTWARE

Store XYZ wants to build a software to manage its stock in and stock out data. The software can receive multiple commands in the format of ***"[+/-] ItemCode Quantity ItemPrice Time".*** The ***[+/-]*** part indicates that the input is stock in or stock out data, respectively. The software will reject any stock-out command that has ***Quantity*** greater than the current number of items in the stock. There are two types ***ItemPrices***: If ***ItemPrice*** is combined with the + sign, it is the buying price. Otherwise, If ***ItemPrice*** is combined with the - sign, it is the selling price.

INPUT

The first line contains an integer N, which is the number of commands ($N \leq 10^6$)

The i^{th} line of the next N lines describes the i^{th} command. This line starts with a sign either + or –, then followed by 4 integers: C_i is the ***ItemCode***, Q_i is the ***Quantity***, P_i is the ***ItemPrice***, and T_i is the ***Time*** (***Time*** has been simplified by using an integer). All values are separated by a space. $0 \leq C_i, Q_i, P_i, T_i \leq 10^9, T_{i-1} \leq T_i$.

OUTPUT

For each ***ItemCode*** which has some items in stock, output in one line: ***ItemCode***, the current number of items in the stock, the average value of the buying prices of items in the stock, and the earliest buying time of the items in the stock.

In addition, the ***ItemCode*** values must be in the ascending order, and the average value is rounded down to units.

EXAMPLE

<table>
<tr><th>Input</th><th>Output</th></tr>
<tr><td><pre>6
+ 1 25 20000 1
- 1 20 30000 2
+ 2 15 30000 2
- 2 25 25000 4
- 2 5 20000 5
+ 2 15 20000 5</pre></td><td><pre>1 5 20000 1
2 25 24000 2</pre></td></tr>
</table>

B. INCOME TAX

TAX! It's a word not associate with much pleasant, however, tax is vital for society to function. Tax is a way for individual to contribute toward the public benefits for society. Normally, the government would impose and collect tax as well as the spending of it, all on behalf of the betterment for the nation. There are many variety of tax, corporate tax, value added tax, import tax, but one kind of tax has the most direct impact on individual life: the INCOME TAX.

Like most other kind of tax, Income tax are percentage based, that means people with higher income will have to pay higher amount of money in tax. However, most civil rights activist argue that, to ensure fairness among citizen, rich people have to pay not only a higher amount of tax, but a higher percentage of their income as tax as well. And thus come the progressive tax strategy. In progressive tax, taxable amount is divided into many tax brackets, each bracket will be charge a different tax rate, the higher the bracket, the higher tax rate.

For example, an income tax bracket and associated tax rate for Vietnamese citizen as per Circular No.111/2013/TT-BTC is give as follow:

Taxable income bracket	Tax rate
0 – 60.000.000	5%
60.000.001 – 120.000.000	10%
120.000.001 – 216.000.000	15%
216.000.001 – 384.000.000	20%
384.000.001 – 624.000.000	25%
624.000.001 – 960.000.000	30%
960.000.001+	35%

As you can see, the last bracket in the example above have no upper bound, the lower bound in the last bracket is called the cut-off value. Any amount going past that value will be taxed at a fixed rate of 35%.

Most country also define an amount of tax exemption income (mostly based on personal circumstance). The taxable amount is the subtraction of tax exemption amount from total annual income.

Your government is proposing a big changes in tax policy, many proposal are given for discussion, so many that your parents find it difficult to follow. As a IT student, you set out to write a program for your parent that can automatically calculate their anual income tax given a specific policy.

INPUT

The first line in input is an integer N $(0 < N < 100)$, this is the number of tax bracket.

The second line contain N integer, each integer is no larger than 10e+18 and in strictly increasing order, those are the lower bound of each tax-bracket. You can safely assume the upper bound of a bracket is exactly 1 currency unit less that the lower-bound of the next bracket.

The third line contain N integer with value no larger than 100, those are the tax rate for each bracket

The last line in input contain two integer no larger than 10e+18, which are the annual income and the tax exemption amount respectively.

OUTPUT

The amount of income tax have to be paid.

EXAMPLE

Input	Output

`7`
`0 60000001 120000001 216000001 384000001 624000001 960000001`
`5 10 15 20 25 30 35`
`284040000`
`0`
	`37008000`

`5`
`5000000 60000001 100000001 200000001 500000001`
`0 10 15 25 40`
`832434000`
`50000000`
	`206973599`

`5`
`5000000 60000001 100000001 200000001 500000001`
`0 10 15 25 40`
`99832434000`
`50000000`
	`39806973600`

C. FREQUENT ITEMSET

BINH, a bright IT student want a glorious career so he decide to study the hottest field nowadays: Machine learning and Data mining. A big chapter in data mining is *frequent itemset*. Even if you haven't got time to look into those terms yourself, a frequent itemsent among huge amount of transaction of item can be defined as follow: Each transaction $t \in T$ can be represented as a set of items. We calculate the *support* of any arbitrary set-of-items (itemset) by the following formula:

$$supp(X) = \frac{|t \in T; X \subseteq t|}{|T|}$$

Given a predefined value *minsup*, an arbitrary itemset X is called frequent itemset iff $supp(X) \geq minsup$. Now, if you have large amount of transaction data, finding all frequent itemset can be a daunting task. And so BINH decide to focus on those itemset that contain exactly ONE item. As a good friends of BINH you gotta help him design an algorithm for that problem

INPUT

First line is the number of transaction T ($1 \leq T \leq 10^6$), and a real number minsupp ($0.0 \leq$ minsup ≤ 1.0).

T line that follow, each line represent one transaction and is given in the following format: The first number is N_i ($1 \leq N_i \leq 10^6$), the number of item in that transaction. The next N_i integer on that line is the id of items in transaction N_i

OUTPUT

Print a single integer: the number of frequent itemset contain exactly 1 item.

EXAMPLE

Input	Output	Note
4 0.6 3 1 3 4 3 2 3 5 4 1 2 3 5 2 2 5	3	3 frequent itemsets are: {2}, {3}, {5}
5 0.6 8 0 2 3 5 6 8 12 15 7 0 1 2 5 11 12 14 6 1 5 7 9 14 22 5 1 2 10 15 17 8 0 2 4 5 11 12 13 15	6	6 frequent itemsets are: {0}, {1}, {2}, {5}, {12}, {15}.

D. GET THE MOST OUT OF DIVISION

We all know adding bracket can greatly alter the result of an math expression. Given an expression contain only one operator - division and all of the operands are integer. Eg: Given the expression 1000/100/10/2 we can alter its result with bracket:

1. 1000/100/10/2 = 0.5
2. 1000/(100/10)/2 = 50
3. 1000/(100/(10/2)) = 50

Let's find a way to insert brackets so that the result is maximum.

INPUT

One single string of digits and "/" sign. It is assured that the expression is valid, all operands are 32-bit positive integer and there is no more than 20 operators.

OUTPUT:

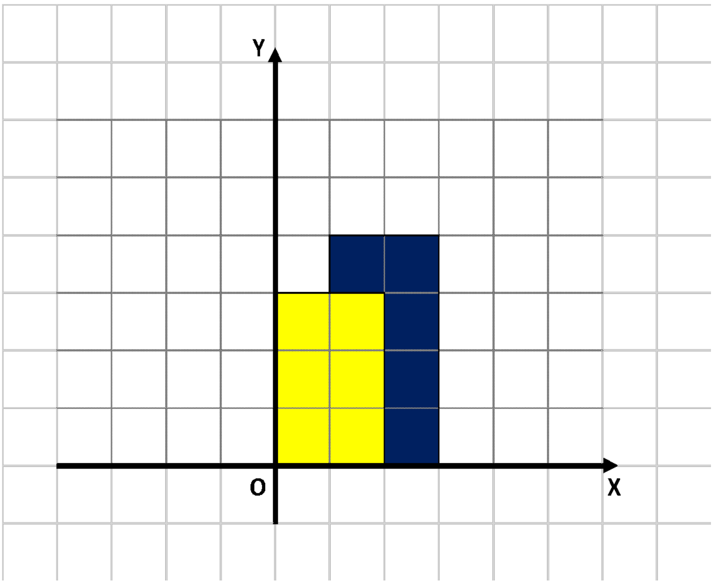
The expression after adding brackets. If there are serveral ways of adding bracket. Output the one that add minimal number of bracket and have opening bracket appears the closest to the start of the expression.

EXAMPLE

Input	Output
`1000/100/10/2`	`1000/(100/10/2)`

E. OVERLAPPING RECTANGLE

On Cartesian coordiante system, a rectangle with one side lay flat on the x-axis can be describe by 3 number: X_L, X_R, H ($X_L < X_R$) which are the left side abscissa, right side abscissa, and height, respectively. Given a bunch of overlapping rectangles, calculate the area cover by all of those rectangles



INPUT

- The first line is an integer N ($0 < N \leq 10^6$)
- N lines that follow, each contains 3 real numbers X_L , X_R và H and represent one rectangle

OUTPUT

The area cover by those rectangles rounded to 3 digit after decimals point.

EXAMPLE

Input	Output
2 0 2 3.0 1 3 4	11.000

F. NUMBER GAME

Sudoku is kid stuff. True hard-core IT student play a much more challenging game: Complete a 6x6 table with 1 digit in each cell so that when you concatenate the cells along a row, a column or the 2 longest diagonal lines, you get a 6 digits prime number every time. Such table is called prime-table Eg:

6	1	1	1	1	1
1	8	1	1	2	3
1	1	7	1	1	9
1	1	1	5	2	1
1	2	2	3	9	3
3	3	7	3	1	3

Given a 6x6 table, some cells have already assigned a number. Your tasks is to fill the empty cells with digit to get a prime-table. If exists several ways to fill the table, output the smallest table (two tables are compare by comparing each cells top to bottom and right to left).

INPUT

6 lines each contain a string 6 character in length. Each character represent a cell in the table. Blank cell is represent by a hash-tag character ('#')

OUTPUT

The smallest prime table

EXAMPLE

Input	Output
6##### #8#### ##7### ###5## ###### ######	611111 181123 117119 111521 122393 337313

G. CARD PRINTER

The F1 Office of University of Information technology just bought a new card printer that can automatically print text in 3 styles: bold, italic and underline.

Text sent to the printer between ‘<’ and ‘>’ will be print in italic, ‘{‘ and ‘}’ for bold, ‘[‘ and ‘]’ for underline.

We can combine those styles by nesting those bracket. We can even print in **bolder** style or higher shear angle italic by double the bracket

But since the printer lack any error reporting mechanism , we need a program to check whether or not the text sent to it is in correct format.

INPUT

A string consists of: English character, digit, white space and 6 formatting digit: ‘{‘,’}’,’]’,’[‘,’<’,’>’.

OUTPUT

Print ‘0’ if the string is in the wrong format. Print ‘1’ if it’s correct.

EXAMPLE

Input	Output
The [{Nguyen Van Ti}]<KHMT>[Khoa 2017]	1
The [{Nguyen Van Ti}]<KHMT[Khoa> 2017]	0

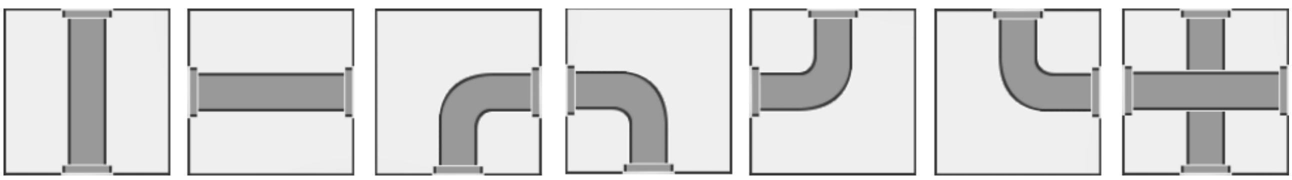
H. OIL PIPE

Crude Oil. A dirty, sticky, flammable liquid substance has been the main factor in accelerating human industrialization during the past century and up until now. Even though producing and consuming oil leave an enormous toll on the environment, oil still remain a vital resources, vital enough for countries to go to war for. Nowadays, super massive oil tanker is one of the most widely use method for transferring oil from the well where it was pumped up from beneath Earth's surface over a large distance where it was needed. Even though those super massive tankers are and wonder in engineering on their own, they are not the most efficient method of transportation for oil however. They are big, expensive to build, hard to handle and if the oil they're transferring leak out, it could cause a disaster for our environment.

The most efficient ways to transfer oil or any liquid in general is a massive network of pump and pipe line spanning across distance, preferably bury beneath the ground. Recently a big oil company has expressed interest in building a big oil pipe line system in your hometown. As per the environmental protection regulation, they have to publish the map of their planned pipe network for everyone to double check. Their map is a rectangular filed divided into multiple squares, each square depict 1 out of 7 types of pipe will be build in that square, those 7 types of pipe are corresponding to numbers:

- 1 : vertical pipe
- 2 : horizontal pipe
- 3 – 6 : curvy pipe
- 7 : crossing pipe (two dis-connected pipe in the same square)

The following picture can give you a better idea of what those pipes are, numbered 1 to 7 respectively:



Oil is pumped from massive pumping stations into each direction that has pipe connected to it (to the maximum of 4 directions). Pumping stations are marked by a lowercase letter on the map. Each pumping stations has a corresponding oil depot, marked by a corresponding uppercase letter on the map.

A good piping system must connect those pumping station to their corresponding oil depot, without leaking oil into the environment.

You have just download the company's map and since it's so huge, you decide to write a computer program to check whether or not this piping system will work.

INPUT

The first line in the input is the positive integer $N < 10000$. This is the number of line in the map.

N lines that follow are *N* string of the same size, no longer than 10000 character. Together this array of string represent the map where each character correspond to 1 square

OUTPUT

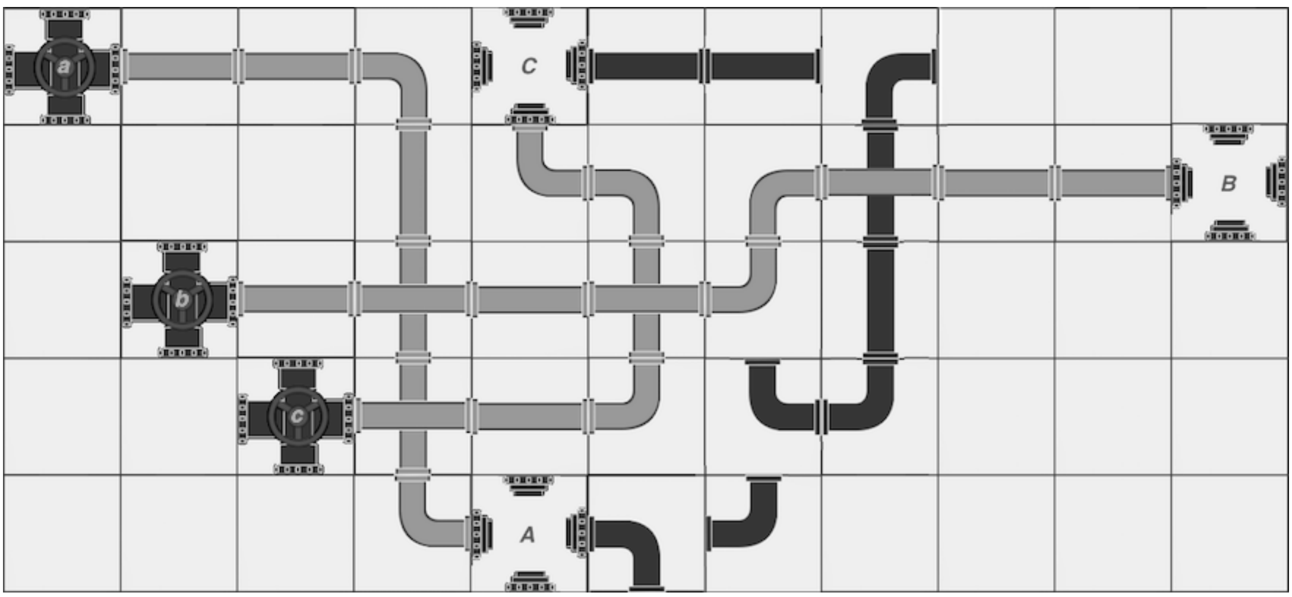
If the pipe system correctly connect each pumping station to its corresponding oil depot, print out the number of pipe square that would be filled with oil when the system work.

If the pipe system would not work correctly, pring out -*X*, where *X* is the number of pipe square would be filled with oil BEFORE the first leakage occurred or the first drop of oil arrive at the depot from the wrong pumping station.

EXAMPLE

Input	Output
5 a224C22300000 0001643722B00 0b27275100000 00c7256500000 0006A45000000	19
7 0002270003777z24 3a40052001000101 1064000001000101 1006774001032501 1000001001010001 1010001001064035 6227206A0622Z250	-48

A picture of the map in the first example is given below for more clarification.



I. THE NEW SQL

You are developing a new, advanced version of SLQ and currently working on LIKE statement. Originally SQL LIKE statement support two wildcard character: Percentage (%) and underscore(_). But you feel that the wildcard supoprt is not enough and devise a new way to combine those two character. The underscore would still represents a single character to match a pattern from a string. The percentage sign would mean that the character precede it could repeat as many times as possible to match a pattern from a string For example, take a look at table CUSTOMERS below :

```
+----+----------+-----+-----------+-------+
| ID | NAME     | AGE | ADDRESS   |SALARY |
+----+----------+-----+-----------+-------+
|  1 | Ha Anh   |  32 | Da Nang   |  2000 |
|  2 | Van Ha   |  25 | Ha Noi    |  1500 |
|  3 | Vu Bang  |  23 | Vinh      |   200 |
|  4 | Thu Minh |  25 | Ha Noi    |  6500 |
|  5 | Hai An   |  27 | Ha Noi    |  8500 |
|  6 | Hoang    |  22 | Ha Noi    |  4500 |
|  7 | Binh     |  24 | Ha Noi    | 10000 |
+----+----------+-----+-----------+-------+
```

The query: `SELECT * FROM CUSTOMERS WHERE SALARY LIKE '200%';` will get the following result:

```
+----+----------+-----+-----------+-------+
| ID | NAME     | AGE | ADDRESS   |SALARY |
+----+----------+-----+-----------+-------+
|  1 | Ha Anh   |  32 | Da Nang   |  2000 |
|  3 | Vu Bang  |  23 | Vinh      |   200 |
+----+----------+-----+-----------+-------+
```

Now you have to write the code that actually do the matching between a pattern and a string

INPUT

First line: the pattern

Second line: the string

OUTPUT:

Print 1 if the string can match the pattern. Print 0 of the string can't match the pattern

EXAMPLE

Input	Output
`_%__a%` `a`	0
`c%a%b` `aab`	1