# Problem B. Brackets and Dots

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

Andrey likes strings, composed from the brackets, and he likes to replace brackets with dots...

Andrey got the string of length $N$, and applies $M$ operations to it. Each operation is given by two integers $l_i$ and $r_i$, and in the result of thos operation in substring $s_{l_i}s)l_i + 1, \ldots, s_{r_i}$ longest regular bracket sequence is replaced by dots such as each bracket is replaced by an one dot.

If the substring have several longest regular bracket sequences, then the minimal one is selected. Regular bracket sequences are compared in next way: lets $a_1$, $a_2$, ..., $a_l$ — indices of the opening brackets for the first sequence, and $c_1$, $c_2$, ..., $c_n$ — indices of the opening brackets for the second sequence. If for some $k$ $a_i = c_i$ $(1 \le i < k)$ and $a_k$¿$c_k$, then first sequence is considered lesser, than second one. If the opening brackets are at the same places in both sequences, then consider indices of the closing brackets $b_1$, $b_2$, ..., $b_l$ for the first sequence and $d_1$, $d_2$, ..., $d_n$ for the second sequence. If exists $k$ $(1 \le k \le l)$ such as $b_i = d_i$ $(1 \le i < k)$ and $b_k < d_k$, then first sequence is considered lesser.

For each operation print number of brackets replaced by dots.

## Input

First line of the input contains one non-empty string $S$ ($|S| \le 5 \cdot 10^5$), consisting of opening and closing brackets ('(' and ')').

Second line contains one integer $M$ $(1 \le M \le 5 \cdot 10^5)$ — number of operations.

Each of the next $M$ lines describe one operation and consists of two integers $l_i$ and $r_i$ $(1 \le l_i \le r_i \le N)$ — starting and ending indices of the substring, respectively.

## Output

For each operation print one integer — number of brackets replaced by dots.

## Example

| standard input | standard output |
|---|---|
| (((())(())) | 4 |
| 3 | 2 |
| 4 9 | 4 |
| 2 8 | |
| 1 10 | |

# Problem C. Crossword

| | |
|---|---|
| Input file: | *standard input* |
| Output file: | *standard output* |
| Time limit: | 2 seconds |
| Memory limit: | 256 mebibytes |

While creating the crosswords often appear next subtask:

Given four words, create the crossword-rectangle from them, i.e:

- Two words must be placed horizontally, and two — vertically.
- Horizontally placed words must be read from left to right, vertically placed words — upside down.
- Each word must have intersection exactly with two other words; each horizontally placed word intersects with both vertically placed words and each vertically placed word intersects with both horizontally placed words.
- Area of the empty rectangle inside the words must be greater than zero (i.e. no intersection can use neighbour letters).

You have four different words. Find the number of different ways to create the crossword-rectangle from them.

## Input

Input consists of 4 lines, each containing non-empty word consisting of at least 3 and at most 100 lowercase English letters. Words are pairwise distinct.

## Output

Print one integer — number of different ways to create the crossword-rectangle from the given words.

## Example

| standard input | standard output |
|---|---|
| aaa<br>axa<br>aya<br>aza | 24 |
| aaaa<br>abba<br>baab<br>bbbb | 40 |

different

# Problem D. Digit

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

Lets write on the paper decimal notation for all integers between 1 and $N$. Find the most frequent decimal digit on the paper. If there are more than one such digit, choose maximal one.

## Input

First line of the input contains one integer $N$ ($1 \le N \le 10^{100\,000}$).

## Output

Print one digit — answer to the problem.

## Example

| standard input | standard output |
|---|---|
| 100 | 1 |
| 99 | 9 |

# Problem E. Enormous Table

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

Given the infinite table:

```
1  2  6  7  15 ...
3  5  8  14 ...
4  9  13 ...
10 12 ...
11 ...
```

Print the value of the element in $a$-th row and $b$-th column.

## Input

Input contains two integers $a$ and $b$ — row and column ($1 \le a, b \le 10^9$).

## Output

Print one integer — answer to the problem.

## Example

| standard input | standard output |
|---|---|
| 2 3 | 8 |

# Problem G. Game of Tic-Tac-Toe

| | |
|---|---|
| Input file: | *standard input* |
| Output file: | *standard output* |
| Time limit: | 2 seconds |
| Memory limit: | 256 mebibytes |

**This is interactive problem**

Your are to play against the jury program in classic tic-tac-toe.

Tic-tat-toe is a game for two players, `X` and `O`, who take turns marking the spaces in a $3 \times 3$ grid. The player who succeeds in placing three of their marks in a horizontal, vertical, or diagonal row wins the game. If the grid is filled and noone won, the game considered to be a draw.

Jury program plays with `O`, you use `X`. If you lose the game, you fail.

## Interaction Protocol

First jury program tells who plays first: '`X`' if you begin the game or '`O`' (uppercase English O) otherwise.

Then interaction starts.

It its jury program' turn now, it tells you coordinates of his turn as pair of integers $r$ and $c$, denoting number of the row and the column; rows are enumerated by sequential integers from 1 to 3 upside down, columns are enumerated by sequential integers from 1 to 3 from left to right. Each turn is printed in the new line, integers are separated by the one space. You may assume that selected cell is not busy.

At your turn you must print your turn in the same format. If you make an incorrect turn, you immediately fail the test.

When game ends, you receive a message from the jury program: "`WIN`" if you won, "`LOSE`" if you lost and "`DRAW`" if game ends up with a draw.

| standard input | standard output |
|---|---|
| X<br>1 3<br>2 3<br>WIN | 1 1<br>2 1<br>3 1 |
| O<br>1 1<br>3 1<br>2 3<br>1 2<br>2 2<br>DRAW | <br>3 3<br>1 3<br>2 1<br>3 2 |

## Note

Please end each output operation by end-of-line and do not forget to flush output buffer. Here are some examples of flush functions:

- For Pascal: **flush(output)**;
- For /++: **fflush(stdout)** or **cout.flush()**;
- For Java: **System.out.flush()**;
- For Python: **sys.stdout.flush()** from library **sys**;
- For C#: **Console.Out.Flush()**;

# Problem I. It is panic?

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

Your friend works in 911 service. One of his duties is to define if the SMS message is panic. The SMS message is considered as panic if it have form "`AAA...!!!!`", i.e. it can be split onto two non-empty parts such as left part consists only of uppercase '`A`', and right part consists only of '`!`' signs.

Given a message, check if it is panic.

## Input

Input consists of nonempty string composed of upper- and lower case English letters and signs '`!`' and '`?`. Length of the string does not exceeds 100.

## Output

Print "`Panic!`" if the message is panic, or "`No panic`" otherwise.

## Example

| standard input | standard output |
|---|---|
| AAA!!! | Panic! |
| AAA | No panic |
| aaa!!! | No panic |

# Problem J. JokeCoin

| | |
|---|---|
| Input file: | *standard input* |
| Output file: | *standard output* |
| Time limit: | 2 seconds |
| Memory limit: | 256 mebibytes |

The JokeCoin is the new cryptocurrency.

Vasya is planning to start working with this currency. People who are mining this currency receive the schedule of the data blocks for the day.For each data block Vasya knows time interval to process this data block, and the reward in JokeCoins for successful mining of this block (i.e. when block was processed at given interval without interruptions). Vasya can freely select the blocks to process; mining of the next selected block may start at the same second, when mining of current block is finished.

Vasya don't have enough resources to process more than one block simultaneously. Additionally, per each second used for mining, Vasya pays fixed amount of JokeCoins to the power company.

Vasya wants to know, if he will have any profit for work with JokerCoin, so he asks you to write the program which can calculate maximum expected profit for Vasya using given schedule.

## Input

First line contains two integers $N$ and $C$ ($1 \le N < 86\,400, 0 \le C \le 1000$) — number of blocks in the schedule and amount of JokeCoins Vasya pays to the power company per second of mining. Next $N$ lines describe blocks.

Each of those lines contain starting time of mining and ending time of mining in the format `HH:MM:SS` between `00:00:00` and `23:59:59`, inclusively, difference between starting and ending time for one block is atleast 1 second and the bonus $P$ ($0 \le P \le 10^5$) in JokeCoins for successful mining of this data block.

## Output

Print one integer — maximal possible profit, or 0, if profit is zero or negative.

## Example

| standard input | standard output |
|---|---|
| 4 0<br>03:00:00 10:10:00 20<br>01:00:00 02:30:00 50<br>16:10:00 19:00:00 100<br>02:30:00 22:00:00 200 | 250 |
| 3 1<br>16:59:00 17:00:00 100<br>01:01:01 01:01:11 20<br>12:00:00 13:00:00 3601 | 51 |
| 4 10<br>00:00:05 00:01:55 1100<br>00:00:10 00:00:21 100<br>00:01:50 00:02:00 80<br>23:59:00 23:59:05 40 | 0 |

# Problem K. King and ICPC

| | |
|---|---|
| Input file: | *standard input* |
| Output file: | *stadnard output* |
| Time limit: | 2 seconds |
| Memory limit: | 256 mebibytes |

In the kingdom are $N$ ICPC teams, numbered sequentially between 1 and $N$, each containing of 3 contestants. Each contestant have KF (KingForces) rating — an integer.

King plans to welcome one contestant per team from teams with numbers between $l_i$ and $r_i$ inclusively to his palace in such a way that:

1. Sum of ratings of all contestants in the palace is divisible by an integer $D$ (favorite integer of the king).
2. While previous holds, sum of ratings of all contestants shall be maximum possible.

The King still not decided, which interval to select, so he wants to know answer (maximal summary rating) for several intervals.

## Input

First line of the input contains two integers $N$ ($1 \le N \le 5 \cdot 10^4$) and $D$ ($1 \le D \le 50$).

Each of the next $N$ lines contains description of one team — ratings of the teammates. Rating is an integer between 0 and $10^9$, inclusively.

Next line contains one integer $M$ — number of the intervals.

Each of the next $M$ lines contain description of one interval — two integers $l_i$ and $r_i$ ($1 \le l_i \le r_i \le N$).

## Output

For each interval print the maximum summary rating divisible by $D$. If it is impossible to choose contestants from teams in such way, print $-1$.

## Example

| standard input | stadnard output |
|---|---|
| 2 2<br>0 1 3<br>1 2 3<br>3<br>1 1<br>2 2<br>1 2 | 0<br>2<br>6 |
| 3 3<br>0 3 6<br>1 4 7<br>1 2 3<br>4<br>1 1<br>1 2<br>1 3<br>2 2 | 6<br>-1<br>15<br>-1 |

# Problem M. Math Task

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

Buratino had $N$ apples. Someone gave him $A$ apples. Buratino ate $B$ apples. How much apples have Buratino?

## Input

Input contains two positive integers $A$ and $B$ ($1 \le A, B \le 10^{15}$)

## Output

Print the formula for the number of apples, reducing an insignificant part (like zero coefficients).

## Examples

| standard input | standard output |
|---|---|
| 2 1 | N+1 |
| 2 3 | N-1 |

# Problem N. Naive HTML

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

The Naive HTML language contains zero or more recurrent elements and the text.

```
1.    <html>
2.        <input type="button" id="x" />
3.        <div id="x"class='cls'p="<i/>">
4.            hello <b>world</b> id='x'
5.        </div>
6.    </html>
```

The code in Naive HTML may contains lowercase English letters, characters "`<>=/`" and quotes `'` `"` and so-called whitespaces: spaces, line feeds and tabulations.

In the example above web-page contains elements marked by tags `<html>`, `<input>`, `<div>`, `<b>` and text `"hello"`, `"world"` `"id='x'"` inside of elements. Tags consists of `tag declaration` and, optionally, `tag closing`. For example, consider a tag `tag`:

- Tag declaration consists of start of declaration – "`<tag`" and end of declaration — "`/>`" "`>`". Between start and end of declaration may be placed zero or more whitespaces, and the tag atributes (see below).
- If tag contains inside another tags or text of length zero or more characters (like `<html>`, `<b>` `<div>` from the sample), then end of declaration is "`>`". Such a tag is called *open*. If it does not contains those (like tag "`<input>`" from the sample), then end of declaration is "`/>`", and such a tag is called *closed*.
- Open tag is always closed as "`</tag>`" — like in lines 4, 5, 6. Closed tags are never closed especially.
- Between start and end of declaration the *attributes* may be placed. Each attribute have the value. For example, "`<input>`" have two attributes — "`type`" with value "`button`" and "`id`" with value "`x`". Attributes are encoded as "`attr="val"`" or "`attr='val'`" — as closing quote must be used same quote, as for opening. Attribute name and value are separated only by the '`=`' character without whitespaces. Attribute name is one or more lowercase English characters. Values of attributes may be arbitrary string does not containing the quote used for this attribute encoding. Between attributes, attributes and start of declaration, attributes and end of declaration may be zero or more whitespaces.

Your task is to write a program which for given HTML and three types of selector answers number of elements on this page, which fit to selectors:

- "`html`" — returns all `<html>` elements from the page.
- "`#myid`" — returns all tags where exists attribute `id` with given value `myid`.
- "`.cls`" — returns all tags where exists attribute "`class`" with given value "`cls`".

## Input

First line of the input consists one positive integer $n$ — number of selectors. Then $n$ lines without heading and trailing spaces, denoting selectors $s_1, \ldots, s_n$ to answer. Selectors contain one or more lowercase English letters and characters "`.#`'. Then the html code is placed till the end of the input file. Common length of the input does not exceed 5 000 characters.

## Output

For each request $s_1, \ldots, s_n$ print the answer on the new line, containing the selector and number of the elements fit to selector. Follow the sample output format.

## Example

| standard input | standard output |
|---|---|
| 5<br>html<br>#x<br>.cls<br>#fakeid<br>i<br><html><br><input type="button" id="x" /><br><div id="x"class='cls'p="<i/>"><br>hello world id='x'<br> <br></html> | Selector "html": found 1 elements<br>Selector "#x": found 2 elements<br>Selector ".cls": found 1 elements<br>Selector "#fakeid": found 0 elements<br>Selector "i": found 0 elements |

## Note

Note that the sample coincides with code given in the statement.