Problem A. Astrology

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	64 megabytes

The famous astrologer Pavel Globus writes a bot for trading stocks in the stock market. Pavel is going to predict the stock price *by the stars*. He analyzed historical data and noticed that, for example, when Mars was in Capricorn, the stock price fell, and when the moon was in Gemini, the quotes went up. Of course, Pavel will not reveal all the details of his algorithm.

Pavel is not so good at programming, and one of the parts of the program that he cannot cope with is determining the zodiac sign, in which the Sun is located, depending on the current date. The zodiac sign for the current date can be determined from the following table:

♈	`Aries`	March 21 — April 19
♉	`Taurus`	April 20 — May 20
♊	`Gemini`	May 21 — June 20
♋	`Cancer`	June 21 — July 22
♌	`Leo`	July 23 — August 22
♍	`Virgo`	August 23 — September 22
♎	`Libra`	September 23 — October 22
♏	`Scorpio`	October 23 — November 22
♐	`Sagittarius`	November 23 — December 21
♑	`Capricorn`	December 22 — January 19
♒	`Aquarius`	January 20 — February 18
♓	`Pisces`	February 19 — March 20

Help Pavel and write a program that determines the zodiac sign by the current date. Pavel, in return, will help you increase your capital.

Input

You are given a string in format ¡¡`YYYY-MM-DD`¿¿, indicating the current date, where `YYYY` — year ($2021 \le$ `YYYY` ≤ 2050), `MM` — month ($01 \le$ `MM` ≤ 12), and `YYYY` — day ($01 \le$ `DD` ≤ 31).

The date is real.

Output

Print one of the words from the list ¡¡`Aries`¿¿, ¡¡`Taurus`¿¿, ¡¡`Gemini`¿¿, ¡¡`Cancer`¿¿, ¡¡`Leo`¿¿, ¡¡`Virgo`¿¿, ¡¡`Libra`¿¿, ¡¡`Scorpio`¿¿, ¡¡`Sagittarius`¿¿, ¡¡`Capricorn`¿¿, ¡¡`Aquarius`¿¿, ¡¡`Pisces`¿¿, corresponding to the zodiac sign.

Example

standard input	standard output
`2021-05-07`	`Taurus`

Problem B. Build the String

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	64 megabytes

Let us have some virtual machine. The machine has a memory stack that can infinitely widen and can contain strings of any finite non-zero length.

The machine supports four operations:

`copy` — copy a string at the stack's end and place it in the stack's end;

`swap` — swap the last string in the stack and the one before the last;

`roll` — cyclically shift three last strings in the stack away from the stack's end;

`fuse` — extract two strings from the stack's end and then place their concatenation at the stack's end.

More formally it looks like that: (`[...]` stands for some sequence of strings at the beginning of the stack, perhaps of zero length):

`copy`: `[...]  x` $\rightarrow$ `[...]  x x`;

`swap`: `[...]  x y` $\rightarrow$ `[...]  y x`;

`roll`: `[...]  x y z` $\rightarrow$ `[...]  y z x`;

`fuse`: `[...]  x y` $\rightarrow$ `[...]  xy`.

Program for a given virtual machine is represented by a command sequence; the machine performs the commands one after another. If the stack doesn't have enough strings to perform the program's current command, then an event `CRASH` occurs and the machine stops functioning. The machine also stops if the program runs out of commands (in this case the `CRASH` event never occurs).

Initially the virtual machine's stack contains two strings and has the form of "`a b`". You have to write a program for the given machine; the program's progress should result in an s string located at the stack's end (at the end of the program's progress the stack can have more than one string left). The program should contain no more than $3 \times |s|$ commands ($|s|$ — is the s string's length). Of course, the program's progress shouldn't lead to the `CRASH` event.

Input

The first line contains the s string. It only consists of lowercase Latin letters "`a`" and "`b`" and has the length from 1 to 10^5 characters.

Output

Print on the first output line number k the number of commands in the program ($0 \leq k \leq 3 \times |s|$). Print on the next k lines k commands, one command per line. The acceptable commands are "`copy`", "`swap`", "`roll`" and "`fuse`". As the result of the program's progress the last element in the stack should be string s (the stack can have more than one string left). The program shouldn't lead to the `CRASH` event. If there are several acceptable decisions, print any of them. See the samples for clarifications.

Example

standard input	standard output
ababa	9 swap copy roll fuse copy fuse copy roll fuse

Note

During the sample program's progress the virtual machine's stack changes in the following manner:

`a b` → `b a` → `b a a` → `a a b` → `a ab` → `a ab ab` → `a abab` → `a abab abab` → `abab abab a` → `abab ababa`

Problem C. Check the String

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	64 megabytes

Suppose we have a virtual machine that is described in the statement of problem B.

Given string s and a program in the form of a list of n commands for the given machine (each command is one from the list "`copy`", "`swap`", "`roll`", "`fuse`").

Initially, the virtual machine stack contains two strings and looks like "`a b`".

All commands from the list are sequentially executed on the virtual machine.

Determine if all the commands from the list were executed correctly, or if the execution of one of them resulted with the `CRASH` event. If all program commands were executed correctly — additionally determine whether the string at the end of the machine's stack after the execution of all commands is equal to the s string or not.

Input

The first line contains the string s. It consists only of lowercase Latin letters "`a`" and "`b`" and has a length from 1 to 10^5 characters.

The second line contains an integer n ($0 \le n \le 10^5$).

The next n lines contain the program commands, one per line, ordered in the order of execution on the virtual machine. Each of the commands is one from the list "`copy`", "`swap`", "`roll`", "`fuse`".

Output

Print exactly one of the following messages:

`CRASH` — if the `CRASH` event occurred during the execution of the program.

`YES` — if after executing all the commands the string at the end of the machine's stack is equal to s.

`NO` — if after executing all the commands the string at the end of the stack is not equal to s.

Examples

standard input	standard output
ababa 9 swap copy roll fuse copy fuse copy roll fuse	YES
a 0	NO
aaba 1 roll	CRASH

Problem D. Diophantine Equation

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	64 megabytes

Bikarp has a square of a positive integer n^2. He wants to split it into a sum of two cubes of positive integers. In other words, Bikarp wants to solve the following Diophantine equation

$$n^2 = x^3 + y^3$$

in positive integers, where n is fixed.

Find a solution of this equation or determine that it doesn't exist.

Input

The first line contains integer T — the number of test samples ($1 \le T \le 3000$).

The i-th of the following T lines contains a single integer n ($1 \le n \le 10^9$).

Output

Output T lines. The i-th of them should contain the answer for the i-th test sample: either "`impossible`", if n cannot be decomposed, or two positive integers x and y.

If some test sample has several solutions — output any of them.

Example

standard input	standard output
4 1 2 3 4	impossible impossible 2 1 2 2

Problem E. Emperor's Palace

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	64 megabytes

The great Emperor Emelyan I is building a palace for himself. To emphasize the greatness of the Emperor, the palace must be huge — it must be visible from space. In addition, the palace must be in the shape of the letter E.

More formally, the palace should be a union of four rectangles: the main building of size $a \times 1$ and three wings of size $1 \times b$, adjacent to the main building from the east, where a and b are positive integers numbers and $a \geq 5$. The north wing should be adjacent to the upper right corner of the main building, and the south wing should be adjacent to the lower right. The middle wing should not adjoin either the north or south wing, but should adjoin with its upper left corner to the main building at an integer distance from the upper right corner of the main building.

In other words, the palace must completely occupy the following squares 1×1 of some rectangle of size $a \times (b+1)$: all squares adjacent to the left, upper or lower side of the rectangle, plus some line inside the rectangle, which is not adjacent to the top and bottom line.

The palace covers an area of $a + 3b$.

The area for the construction of the future palace is a rectangle of size $h \times w$, divided into cells of size 1×1. Some cells are occupied by the Emperor's gardens, which cannot be built on. The rest of the cells are free and you can build on them.

Determine the largest area of the palace that can be built on empty spaces.

Input

The first line contains two integers h and w ($5 \leq h, w \leq 10^6$, $h \times w \leq 5 \times 10^6$).

Each of the following h lines contains w characters. The j-th character in the i-th of these lines is equal to "`#`" if the cell with coordinates (i, j) is occupied and "`.`" — if this cell is free.

Output

Print one integer — the largest possible area of the palace. If it is impossible to build a palace, print "`-1`".

Examples

standard input	standard output

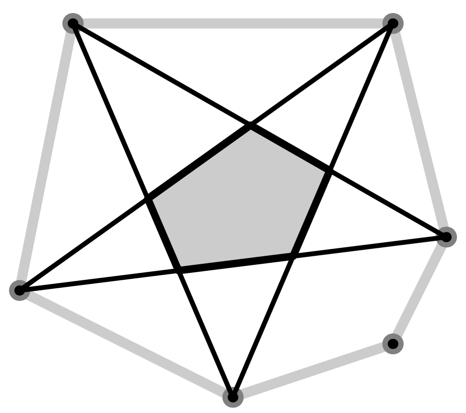
`6 6`
`#.....`
`....#.`
`..####`
`....##`
`..##..`
`......`
	`14`

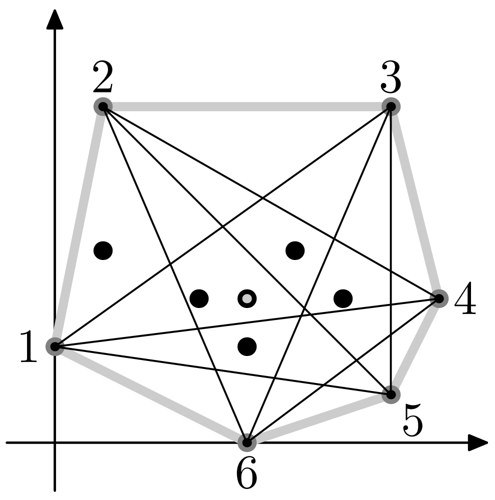
`6 6`
`#.....`
`...##.`
`..####`
`....##`
`..##..`
`......`
	`12`

Problem F. Five-pointed Queries

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	64 megabytes

Nowadays, the 5G mobile communication standard is being introduced everywhere. But progress does not stand still, and the researchers of the Lucifer Laboratory are working hard to develop a new communication standard. The development turns out to be so innovative that it was decided to call the new standard not 6G, but immediately 666G. Researchers claim that this technology will make it possible to call Satan himself.





The new technology can be described as follows. The k comminication towers, which can be considered points on the plane, are located at the vertices of a convex k-gon. Any five pairwise different towers are the tops of a five-pointed star and allow serving subscribers inside a pentagon, which is bounded by edges of the five-pointed star.

You task is to test the load on communication towers using a model example. Let there be n subscribers who can be considered as points on the plane inside the convex k-gon formed by the k communication towers. We will assume that subscribers do not change their location. For each subscriber it is known whether he is active or not.

Further, there are q events of two types, ordered chronologically:

1. Subscriber numbered t changes its activity. That is, if the subscriber was active, then he becomes inactive and vice versa.
2. For some five pairwise different communication towers, it is necessary to determine the number of **active** subscribers who are served by these five towers.

You need to simulate all events and respond to all requests of type 2.

Input

The first line contains an integer k — the number of communication towers ($5 \le k \le 30$).

The i-th of the following k lines contains two integers X_i and Y_i — coordinates of the i-th communication tower. It is guaranteed that the communication towers are at the vertices of a strictly convex k-gon, that is, no three towers are on the same straight line. The towers are listed in clockwise order of traversing this k -gon.

Further on a separate line is an integer n — the number of subscribers ($1 \le n \le 50\,000$).

The i-th of the following n lines contains three integers x_i, y_i and z_i — the coordinates of the i -th subscriber and his activity. $z_i = 1$ means that the i -th subscriber is active, $z_i = 0$ — that he is not. It is guaranteed that all subscribers are strictly inside the convex k-gon formed by communication towers. No subscriber is on the segment that connects any two communication towers.

Further on a separate line is an integer q — the number of events ($1 \le q \le 50\,000$).

The i-th of the following q lines describes the i-th event. An event of type 1 is written as "1 t", where t is the number of the subscriber for which the activity is changing ($1 \le t \le n$). An event of type 2 is written as "2 a_i b_i c_i d_i e_i", where a_i, b_i, c_i, d_i and e_i — tower indexes for which you need to determine the number of active served subscribers ($1 \le a_i < b_i < c_i < d_i < e_i \le k$).

It is guaranteed that there is at least one request of the type 2.

The coordinates of all points are integers not exceeding 5×10^5 in absolute value. No two points in the input are equal.

Output

Print p lines, where p is the number of events of type 2. In the i-th line, print one integer - the answer to the i-th query of the type 2.

Example

standard input	standard output

```
6
0 2
1 7
7 7
8 3
7 1
4 0
6
1 4 1
3 3 1
4 3 0
4 2 1
5 4 1
6 3 1
8
2 1 2 3 4 5
2 1 2 3 4 6
1 3
2 1 2 3 4 6
2 1 2 3 5 6
2 1 2 4 5 6
2 1 3 4 5 6
2 2 3 4 5 6
```
	```
	2
	2
	3
	3
	1
	0
	1
	```

Note

The point configuration in the example is shown on picture above.

Problem G. Gas and Minerals

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	64 megabytes

An Artifact that could turn the tide of the war was discovered in one of the distant Terran colonies. Meanwhile, the intelligence service reports that a Zerg swarm is moving towards the colony. It is necessary to protect the Artifact at all costs before the arrival of reinforcements.

You have m units of minerals and g units of Vespen gas. In addition, there are n types of defensive buildings available for construction. A building of type i requires a_i units of minerals and b_i units of gas to construct, and increases the defenses of the base by c_i units. You can construct any number of buildings (including zero) of any type, provided that the total costs of minerals and gas for all buildings will not exceed m and g, respectively.

Determine what the maximum total building defense capability can be achieved under the given constraints.

Input

The first line contains three integers m, g and n — the number of available units of minerals and gas, respectively, and the number of building types ($0 \le m \le 1000$, $0 \le g \le 1000$, $1 \le n \le 10$).

The i th of the following n lines contains three integers a_i, b_i and c_i — the amount of units of mineral and gas are needed to construct a building of the i-th type and its defenses ($1 \le a_i \le 100$, $0 \le b_i \le 100$, $0 \le c_i \le 100$).

Output

Print a single integer — the maximum total building defense capability that can be achieved.

Examples

standard input	standard output

`10 10 3`
`7 0 6`
`6 2 7`
`2 5 5`
	`12`

`11 10 3`
`7 0 6`
`6 2 7`
`2 5 5`
	`16`

Note

In the first example, the optimum is provided by the construction of one building of type 2 and one building of type 3.

In the second example, it is most profitable to construct one building of type 1 and two buildings of type 3.

Problem I. Infection

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	64 megabytes

An emergency happened in one secret organization. In the middle of the working day, one of the employees was hospitalized with symptoms of an extremely dangerous *colonavirus* infection. In this regard, the management of the organization wants to establish which employees can still be infected, but the symptoms of the disease have not yet shown themselves.

There are n employees in the organization, who can be numbered with integers from 1 to n. From the recordings of CCTV cameras, the organization's management established when which employees contacted each other. In addition, management took into account the following assumptions:

- At the beginning of the working day, exactly one of the employees was infected, and each of the initial states could happen with a probability of $1/n$.
- If two employees come into contact with each other, and one of them is infected and the other is not, then a healthy employee becomes infected with a probability of $1/2$. If both employees are healthy, or both are infected, nothing happens.
- If an employee is infected, he cannot suddenly recover, that is, he remains infected until the end.
- It is known that the employee numbered k was eventually infected.

A chronological list of employees' contacts is given. Determine for each employee the probability of being infected according to the assumptions described above.

Input

The first line contains three integers n, k and m — the number of employees, the number of the infected employee and the number of contacts, respectively ($2 \le n \le 15$, $1 \le k \le n$, $1 \le m \le 50$).

The i-th of the following m lines contains two integers x_i and y_i — indexes of employees who participated in the i-th contact ($1 \le x_i, y_i \le n$, $x_i \ne y_i$).

All contacts in the list are given in chronological order

Output

Print n lines. On the i-th line print the probability of infection of the i-th employee as an irreducible fraction a/b. See the example for a more precise understanding.

Examples

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

3 2 2
1 2
2 3
	1/2
	1/1
	5/8

4 1 4
1 2
2 3
3 4
4 1
	1/1
	19/37
	17/37
	27/37

Problem J. Jumping Path

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	64 megabytes

Popeye the Sailor loves to eat spinach. He also loves to smoke his corn-made pipe. And which he constantly smokes.

Popeye lives in the Sweethaven village. On the main street of Sweethaven, which can be represented as a straight line, there are n public places, which can be considered as points on a straight line located at coordinates $x_1, x_2, \cdots, x_n$, respectively.

Popeye needs to get from the A point on the main street to the B point. Everything would have been simple, if not for the law that passed Sweethaven's authority: now smoking nearer than r from a public place is prohibited. Fortunately, Popeye has a pole length $R \geq r$,, with which he can jump over forbidden zones.

Popeye is initially located at point A. He can move from x to y on foot in $|x-y|$ time. Also, at any time, he can use the pole and move from point x to point $x+2R$ or $x-2R$, moving along a semicircle of radius R, while he spends πR time. At the end of the path, Popeye must be at point B, and at no point on the trajectory of Popeye can be closer than r to any public place.

Determine the shortest time it takes Popeye to get from A to B. Or determine that it is impossible to get from A to B under the given constraints, so Popeye will have to use the power of spinach.

Input

The first line contains five integers n, r, R, A and B ($1 \leq n \leq 500$, $1 \leq r \leq R \leq 10^6$, $-10^9 \leq A, B \leq 10^9$). The second line contains n integers $x_1, x_2, \cdots, x_n$ ($-10^9 \leq x_i \leq 10^9$, $1 \leq i \leq n$). All x_i are pairwise distinct. It is guaranteed that the points A and B are different and are not located in any of the forbidden zones.

Output

Print one real number — the smallest time. The answer will be counted if it differs from the jury's answer by no more than 10^{-6} in absolute or relative value. If it is impossible to get from A to B, print -1.

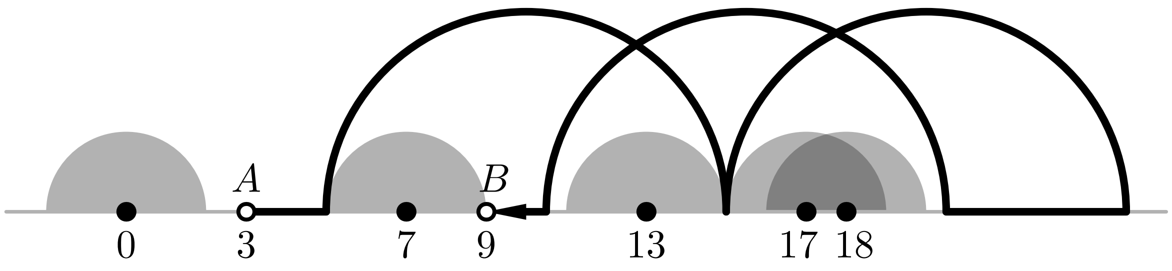
Example

standard input	standard output

`5 2 5 3 9`
`13 0 17 7 18`
	`55.1238898038`

Note

For an example from the statement, one of the optimal trajectories of movement looks as follows:

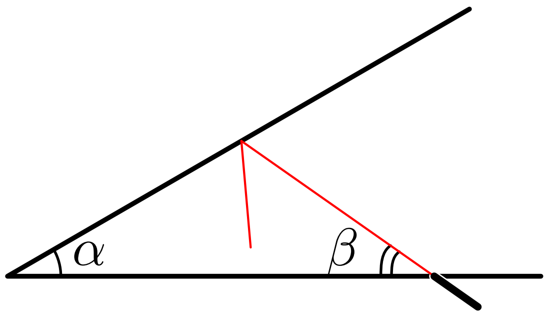


Elapsed time — $8 + 15\pi$.

Problem L. Laser Beam

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	64 megabytes

There are two infinite flat mirrors located at an angle α relative to each other so that they can be considered as rays on the plane when viewed from the side. Through a tiny hole in one of the mirrors, a laser beam is launched at an angle β as shown in the figure below:



Your task is to count the number of reflections of the laser beam from the mirrors before it goes to infinity. The angle of incidence of the beam on the mirror always coincides with the angle of reflection. The hole through which the beam is launched is extremely small, so we can assume that if the beam suddenly hits the hole, it will still be completely reflected according to the usual rules.

Input

Given two integers α and β — angles given in degrees ($1 \le \alpha \le 179$, $1 \le \beta \le 179$).

Output

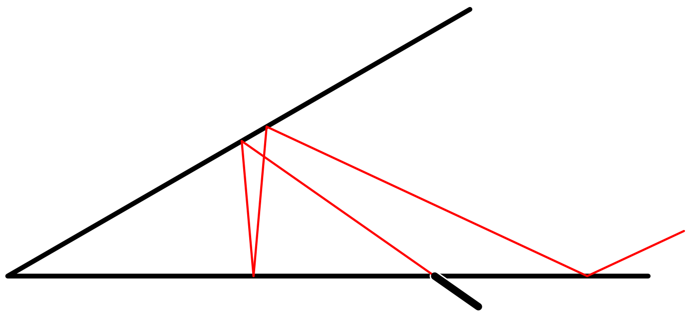
Print one integer — the number of reflections.

Examples

standard input	standard output
`30 35`	`4`
`90 90`	`0`
`30 60`	`3`

Note

In the first example, the beam is reflected as follows:



Problem M. Mirror Brackets

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	64 megabytes

Let's remind you what a correct bracket sequence (CBS) is.

- An empty string is CBS.
- If s is CBS, then `(`s`)` and `[`s`]` also are CBS.
- If s and t are CBS, then st also is CBS.
- If a sequence can't be obtained applying the rules given above, then it is not a CBS.

Vasya wrote a correct bracket sequence consisting of brackets "`(`", "`)`", "`[`" and "`]`". After that Vasya added to some places of the sequence characters "`b`", "`d`", "`o`", "`p`", "`q`" and "`x`". As a result, Vasya got some string z.

Now Vasya performs the following operations:

If z has a substring of the form `(`s`)` (there are no brackets in s), then Vasya replaces it with s string's mirror reflection relative to vertical axis. If z contains a substring of the form `[`t`]` (there are no brackets in t as well), then Vasya replaces it with t string's mirror reflection relative to horizontal axis. For example, substring "`(qbpoxd)`" will be replaced with "`bxoqdp`", and substring "`[qbpoxd]`" will be replaced with "`dpboxq`".

Vasya performs operations until there are no brackets left in the string.

Calculate by the given z string what result Vasya will have in the end.

Input

The only line contains the z string whose length does not exceed 10^5 characters, consisting of characters "`(`", "`)`","`[`", "`]`", "`b`", "`d`", "`o`", "`p`", "`q`" and "`x`". It is guaranteed that all the brackets in z organize a correct bracket sequence. It is also guaranteed that the input data contain at least one letter.

Output

Print the string Vasya will get in the end.

Examples

standard input	standard output
`(qbpoxd)[qbpoxd]`	`bxoqdpdpboxq`
`d[xd(bx)op]q()[]xx`	`dxqxqobqxx`

Problem N. Numbers

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	64 megabytes

A non-negative integer is K-digit, if it can be written using K digits, but $(K-1)$ digits are not enough for this. For example, 43 is a two-digit number, 2010 is four-digit, and 0 and 5 are one-digit numbers.

For given A and B count, how many K-digit non-negative integers exist, where K is not less than A and not greater than B.

Input

The only line contains integers A and B ($1 \le A \le B \le 1000$).

Output

Output the answer to the problem without excessive leading zeros.

Example

standard input	standard output
1 1	10
3 5	99900

Problem O. Overwhelming Vowels

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	64 megabytes

There are 26 letters in English alphabet, of which 5 (‘a’, ‘e’, ‘i’, ‘o’, ‘u’) are vowels, 20 are consonants and one is considered a semivowel, i.e. neither a vowel nor a consonant (letter ‘y’). A word is sonorous if the number of vowels in it exceeds the number of consonants. If a word contains some letter more than once, then each its occurence counts — for example, “alabama” is a sonorous word.

For a given word determine the least number of letters in it which must be replaced to obtain a sonorous word.

Input

A single line of input contains the word given. The length of the given word does not exceed 1000 letters. The word consists of lowercase letters.

Output

Output the least number of letters replacement of which makes the given word sonorous.

Example

standard input	standard output
`dog`	`1`
`alabama`	`0`
`y`	`1`