Problem A. Easy Jump

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	512 mebibytes

Waiting for the silk song, Grammy decides to challenge the Path of Pain in White Palace. After 4 to 5 hours of painful efforts, she achieves it. And she suddenly wonders about the expected time for her to pass it because she would feel lucky if the time she used is less than the expected time. The whole process could be modeled as below.

There are n stages in the Path of Pain. For stage i, Grammy can spend 1 unit of time to try it and the probability of passing it is p_i. If she passes it, she will go to stage $i+1$ immediately (or just finish the challenge if $i = n$). Otherwise, she will take 1 unit of damage and back to stage i. For understanding convenience, we use hp and mp to represent mask and soul. If Grammy takes 1 unit of damage, her hp will decrease by 1. When the hp becomes 0, she will die and have to start over. Since Grammy is a lazy girl, she will never let her hp become 0 even if she is at stage 1. At any time, Grammy can do one of the following things.

1. Challenge the stage, costing 1 unit of time, with probability p_i passing it, $1 - p_i$ failing and taking 1 unit of damage.
2. Focus, using 1 unit mp to heal 1 unit hp, costing $T1$ unit time.
3. Heal by hiveblood, costing $T2$ unit time to heal 1 unit hp. But hiveblood can work only every time after taking 1 unit of damage.

There are soul totems in some of the stages. For simplification, we assume that Grammy can fill up her soul whenever she is at a stage having a soul totem without costing any time. And she can fill up any number of times in a stage.

Given the hp upper bound H, mp upper bound S, assume that Grammy starts the challenge with full hp and mp, could you tell her the expected time she has to use to achieve the challenge? Note that since Grammy is lazy, she will choose the best strategy when challenging to minimize the expected time.

Input

The input contains only a single case.

The first line contains three positive integers n, H and S ($1 \le n \le 1\,000$, $2 \le H \le 9$, $0 \le S \le 6$), denoting the number of stages, the upper bound of hp and mp.

The second line contains n integers $P_1, P_2, \ldots, P_n$ $(1 \le P_i \le 99)$, denoting the moleculars of probabilities. For each stage, we define $p_i = \frac{P_i}{100}$.

The third line first comes an integer K, denoting there are K stages that have soul totems. Follows K distinct integers $a_1, a_2, \ldots, a_K$ $(1 \le K \le n, 1 \le a_i \le n)$, denoting the index of the stages which have soul totems.

The fourth line contains two integers $T1$ and $T2$ $(1 \le T1, T2 \le 100)$, denoting the cost of focus and healing by hiveblood.

Output

Output the minimum expected time. Your answer will be considered correct if its absolute or relative error does not exceed 10^{-6}.

Examples

standard input	standard output
1 2 0 50 0 1 2	4.000000000000
2 3 1 50 50 1 1 1 3	6.000000000000

Problem B. Terrible Additive Number Theory Problem

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

Define P_i as the i-th prime.

Find the number of solutions x such that $x = \prod_{i=l}^{r} P_i = 2^k P_{r+1} - 1$, where $l, r, k \in \mathbb{N}^+, 1 \le l \le r$, and $x \le n$.

Input

Input contains a single integer n $(1 \le n \le 10^{18})$

Output

Output a single integer, indicating the number of solutions less than or equal to n.

Example

standard input	standard output
100	0

Problem C. Race

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

Pigetown is a city with n crossings and m bidirectional roads. A huge race event is going to be held in Pigetown. There are k types of race tracks, and each road in the city can be viewed as a particular type of race track.

In the race, each participant should choose an integer i such that $1 \le i \le q$, start at crossing S_i, visit each type of race tracks the same number of times, and finally arrive at crossing T_i in order to finish the race.

Grammy wants to know if it is possible to finish the race when choosing each integer i. Write a program to help her solve the problem.

Input

The first line contains 4 integers $n, m, k, q (1 \le n, m, q \le 200\,000,\ 1 \le k \le 30)$, indicating the number of crossings, the number of roads, the number of race track types, the upper limit of chosen integer i, respectively.

In the next m lines, each line contains 3 integers $u, v, t (1 \le u, v \le n, 1 \le t \le k)$, indicating that there is a bidirectional road between crossing u and crossing v with type t.

In the next q lines, each line contains 2 integers $S_i, T_i (1 \le S_i, T_i \le n)$, indicating one possible combination of starting point and ending point.

Output

Output q lines.

In the i-th line, if it is possible to finish the race while choosing integer i, output "Yes", otherwise output "No"(Without quotes).

Example

standard input	standard output
7 9 3 4 1 2 1 2 3 1 3 1 2 1 4 3 5 6 2 6 7 1 6 7 3 7 7 2 5 5 1 6 7 1 4 2 4 2 5	Yes No Yes No

Problem D. Candy Machine

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	512 mebibytes

JB loves candy very much.

One day, he finds a candy machine with N candies in it. After reading the instructions of the machine, he knows that he can choose a subset of the N candies. Each candy has a sweet value. After JB chooses the subset, suppose the average sweet value of the chosen candies is X, all the candies with sweet value strictly larger than X will belong to JB. After JB makes the choice, the machine will disappear, so JB only has one opportunity to make a choice.

JB doesn't care how sweet the candies are, so he just wants to make a choice to maximize the number of candies he will get. JB has been fascinated by candy and can't think, so he needs you to help him.

Input

The first line contains one integer N ($1 \le N \le 10^6$), denoting the number of candies in the machine.

The second line contains N integers $a_1, a_2, \ldots, a_N$ ($1 \le a_i \le 10^9$), denoting the sweet values of the candies.

Output

One integer, denoting the maximum number of candies JB can get.

Example

standard input	standard output
5 1 2 3 4 5	2

Problem E. The Profiteer

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 mebibytes

BaoBao has a store. There are n items in the store, labeled by $1, 2, \ldots, n$. The value of the i-th item is v_i, and the price of it is a_i dollars. JB is planning to visit BaoBao's store tomorrow. JB always buys items optimally. Assume JB has t dollars, he will buy a set of items such that the total value is maximized and the total price is no more than t.

The profiteers cheated people right and left. BaoBao knows JB is rich, so he decides to choose a pair of integers l and r, where $1 \le l \le r \le n$, and raises the prices of all the items indexed in $[l, r]$. When JB comes tomorrow, he will need to pay b_i dollars instead of a_i dollars for the i-th item, where $l \le i \le r$.

However, BaoBao doesn't know how rich JB is, he only knows t is an integer uniform randomly chosen in $[1, k]$. BaoBao doesn't want JB to buy so many good items, he is now wondering how many pairs of integers l and r he can choose such that the expected total value of JB's shopping list $\frac{f(1)+f(2)+\cdots+f(k)}{k}$ will not exceed E, where $f(t)$ denotes the total value of the shopping list when JB has t dollars. Please write a program to help BaoBao.

Input

The input contains only a single case.

The first line contains three integers n, k and E ($1 \le n, k \le 200\,000$, $n \times k \le 10^7$, $1 \le E \le 10^9$).

Each of the following n lines contains three integers v_i, a_i and b_i ($1 \le v_i \le 10\,000$, $1 \le a_i < b_i \le k$), denoting the value, the initial price and the raised price of the i-th item.

Output

Print a single line containing an integer, denoting the number of valid pairs of integers l and r.

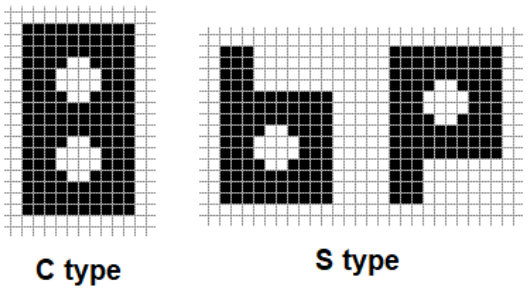
Examples

standard input	standard output
4 5 3 3 2 4 1 2 3 2 1 2 3 1 3	1
4 5 4 3 2 4 1 2 3 2 1 2 3 1 3	3

Problem F. BpbBppbpBB

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	512 mebibytes

Grammy has learned how to engrave stamps recently. She engraved two types of special stamps, type C has a capital letter "B" on it, and type S has a small letter "b" or a small letter "p" on it. The shapes and sizes of the stamps are illustrated in the following picture.



Grammy stamped these letters (with rotations) on a grid paper without overlapping, the letters can only be pressed at the piece of paper if it lies totally inside the piece of paper. However, Grammy forgot how many times she used each type of stamps. Please count the letters and help her to remember them.

The black part of the stamps may be adjacent but may not overlap.

Note that the stamps can be rotated to a multiple of 90 degrees.

Input

The first line consists of two integers n, m ($1 \le n, m \le 1\,000$), representing the size of the paper.

In the following n lines, each line consists of m characters, representing the current state of the paper. "#" stands for a black square and "." stands for a white square.

Output

Output two integers, denoting the number of type C stamps and the number of type S stamps, respectively.

Examples

standard input	standard output
10 17 ################## ################## ################## ####..#####..#### ###....###....### ###....###....### ####..#####..#### ################## ################## ##################	1 0
14 11 .########## .########## .########## .####..#### .###....### .###....### .####..#### .########## .########## .########## .###....... .###....... .###....... .###.......	0 1
20 14 .##########... .##########... .##########... .####..####... .###....###... .###....###... .####..####... .##########... .##########... .##########... .############## .############## .############## .#######..#### ....###....### ....###....### ....####..#### ############### ############### ###############	0 2

Problem G. Dynamic Reachability

Input file:	*standard input*
Output file:	*standard output*
Time limit:	12 seconds
Memory limit:	512 mebibytes

You are given a directed graph with n vertices and m edges, the vertices of which are labeled by $1, 2, \ldots, n$. The color of each edge is either black or white. Initially, all the m edges are colored black.

You need to perform q operations. Each operation is one of the following:

- "1 k" ($1 \le k \le m$): Change the color of the k-th edge in the input from black to white and vice versa.
- "2 u v" ($1 \le u, v \le n$, $u \ne v$): You need to answer whether vertex u can reach vertex v without passing any white edge.

Input

The input contains only a single case.

The first line contains three integers n, m and q ($2 \le n \le 50\,000$, $1 \le m, q \le 100\,000$), denoting the number of vertices, the number of edges, and the number of operations.

Each of the following m lines contains two integers u_i and v_i ($1 \le u_i, v_i \le n$, $u_i \ne v_i$, $1 \le i \le m$), denoting a directed edge from vertex u_i to vertex v_i.

Each of the next q lines describes an operation in formats described in the statement above.

Output

For each query, print a single line. If vertex u can reach vertex v without passing any white edge, print "YES". Otherwise, print "NO".

Example

standard input	standard output
5 6 7 1 2 1 3 2 4 3 4 3 5 4 5 2 1 5 2 2 3 1 3 1 4 2 1 4 1 3 2 1 5	YES NO NO YES

Problem H. Barbecue

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1.5 seconds
Memory limit:	512 mebibytes

Putata and Budada are playing a new game. In the beginning, Putata has a note with a string consists of lowercase letters on it. In each round, the player who has the note must rip off a character from the beginning or the end of the note, then pass it to the other player. If at any moment, the string on the note is a palindrome, then the player who has the note loses. Notice that both before or after the player ripping off a character from the note, the player is considered to have the note. A string $s_1 s_2 \ldots s_n$ of length n is considered to be a palindrome if for all integers i from 1 to n, $s_i = s_{n-i+1}$.

However, when Putata found the note, he found that someone have played on this note before. Since both Putata and Budada are clever and will always choose the best way to make themselves win, they wonder who will win the game, and they ask you for help. Formally, you are given a string of length n and you have to answer q queries, each query is described by two integers l and r, which means you have to determine who will win if Putata and Budada play the game described above on string $s_l s_{l+1} \ldots s_r$.

Input

The first line contains two integers n, q ($1 \le n, q \le 1\,000\,000$), denoting the length of the string and the number of queries.

The second line contains a string s of length n, consisting of lowercase English letters.

Each of the following q lines contains two integers l and r ($1 \le l \le r \le n$), describing a query.

Output

For each query, print a single line. If Putata wins the game in one query, output "`Putata`" (without quotes). Otherwise output "`Budada`".

Example

standard input	standard output
7 3 potatop 1 3 3 5 1 6	Putata Budada Budada

Problem I. Easy Fix

Input file:	*standard input*
Output file:	*standard output*
Time limit:	3 seconds
Memory limit:	512 mebibytes

Since Grammy plays Hollow Knight day and night and forgets the homework Tony gives her, she already has no time to do it. As a talented programmer and good friend of Grammy, you decide to help her. The problem is described as follows.

Given a permutation $p = p_1, p_2, \ldots, p_n$. We define A_i as the number of j satisfying that $j < i \wedge p_j < p_i$, B_i as the number of j satisfying that $j > i \wedge p_j < p_i$, and $C_i = \min(A_i, B_i)$.

There are m queries. For the i-th query, you should output the value of $\sum_{i=1}^{n} C_i$ if we swap p_u and p_v. Note that we will recover the permutation p after each query which means queries are independent of each other.

Input

The input contains only a single case.

The first line contains one positive integer n ($1 \le n \le 100\,000$). It is guaranteed that p is a permutation of $1, 2, \ldots, n$.

The second line contains n **distinct integers** $p_1, p_2, \ldots, p_n$ ($1 \le p_i \le n$).

The third line contains one positive integer m ($1 \le m \le 200\,000$).

The following m lines describe m queries. The i-th line contains two integers u and v ($1 \le u, v \le n$), denoting the parameter of the i-th query. Note that u may be equal to v.

Output

The output contains m lines. Each line contains one integer, denoting the answer to the i-th query.

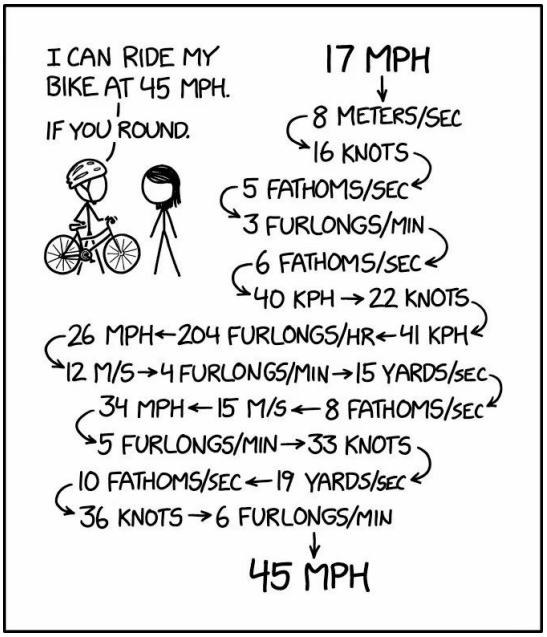
Examples

standard input	standard output
7 1 6 2 7 5 4 3 7 1 7 2 6 3 5 4 4 1 1 2 1 3 7	7 6 6 7 7 6 8
5 5 3 1 2 4 3 3 1 2 5 3 3	3 0 0

Problem J. Rounding Master

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

Grammy has obtained master degree in rounding(she awarded herself). She can use her rounding techniques to obtain a super large number by changing a unit and round.



In particular, she has a number x, which initially equals to 1. She will perform the following operation k times, and finally make her number $x \geq n$. In each operation, she will multiply x by $q(q > 0)$, and round it. Rounding a number w means to find integer a such that $a \leq w < a + 1$, and if $w \geq a + 0.5$, then change w into $a + 1$, otherwise change w into a.

Can you help her to choose the minimum q such that after k operations, x will be greater than or equal to n.

Input

The first line contains two integers $n, k(1 \leq n, k \leq 10^{18})$, representing the final target and the number of operations.

Output

Output a positive real number q, representing the answer. You answer will be considered correct if its absolute or relative error does not exceed 10^{-6}.

Example

standard input	standard output
18 4	2.125000000000

Problem K. Resource Calculator

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

Grammy is playing her favorite video game. The characters in that game have multiple ascension levels, normal levels, and 3 talent levels.

A Character can upgrade its normal level by gaining experience. We assume that the only way for a character to gain experience is to feed it with coins and the following 3 types of experience materials. A "Wanderer's Advice" can provide 1 000 experience to a character, while an "Adventurer's Experience" can provide 5 000, a "Hero's Wit" can provide 20 000. Whenever a character gain 1 experience, 0.2 coins will be spent as upgrade cost. The amount of experience needed is in the following table.

Level	To Next	Level	To Next	Level	To Next
1	1000	31	30650	61	108950
2	1325	32	32250	62	112050
3	1700	33	33875	63	115175
4	2150	34	35550	64	118325
5	2625	35	37250	65	121525
6	3150	36	38975	66	124775
7	3725	37	40750	67	128075
8	4350	38	42575	68	131400
9	5000	39	44425	69	134775
10	5700	**40**	46300	**70**	138175
11	6450	41	50625	71	148700
12	7225	42	52700	72	152375
13	8050	43	54775	73	156075
14	8925	44	56900	74	159825
15	9825	45	59075	75	163600
16	10750	46	61275	76	167425
17	11725	47	63525	77	171300
18	12725	48	65800	78	175225
19	13775	49	68125	79	179175
20	14875	**50**	70475	**80**	183175
21	16800	51	76500	81	216225
22	18000	52	79050	82	243025
23	19250	53	81650	83	273100
24	20550	54	84275	84	306800
25	21875	55	86950	85	344600
26	23250	56	89650	86	386950
27	24650	57	92400	87	434425
28	26100	58	95175	88	487625
29	27575	59	98000	89	547200
30	29100	**60**	100875	**90**	MAX

A Character can upgrade its ascension level at normal level $20, 40, 50, 60, 70$, and 80. Before upgrading its ascension level at the corresponding normal level, the character cannot gain any more normal experience. If the amount of experience that a character can gain is less than the amount that an experience material can provide, the overflown part of experience are wasted, and will not spend coins. Additionally, extra ascension materials and coins are needed for upgrading the character's ascension level.

The first type of ascension materials is gems (Agnidus Agate, Prithiva Topaz, Shivada Jade, Vajrada

Amethyst, Varunada Lazurite, Vayuda Turquoise, and Brilliant Diamond). Each type of gemstone has 4 rareness levels: sliver, fragment, chunk, and gemstone.

The second type of ascension materials is boss drops, which has nothing special.

The third type of ascension materials is mob drops. Mob drops also have different rareness levels: common, rare, and epic.

The last type of ascension materials is local specialties, which is (probably) the character's favorite item in the world.

The amount of materials needed is in the following table.

Ascension level	Normal level	Gemstones	Boss Drops	Mob Drops	Specialties	Coins
0→1	20	1 Sliver	0	3 Common	3	20000
1→2	40	3 Fragments	2	15 Common	10	40000
2→3	50	6 Fragments	4	12 Rare	20	60000
3→4	60	3 Chunks	8	18 Rare	30	80000
4→5	70	6 Chunks	12	12 Epic	45	100000
5→6	80	6 Gemstones	20	24 Epic	60	120000

The 3 talent levels are upgraded mutually independently. In order to upgrade a talent level, 4 types of different materials and coins are needed.

The first type of talent level-up materials is normal mob drops, which is ths same as the third type of ascension materials.

The second type of talent level-up materials is talent books, which has 3 different rarities: Teachings, Guides, and Phylosophies.

The third type of talent level-up materials is weekly boss drops, which has nothing special.

The last type of talent level-up materials is "Crown of Insight", which is only used in the last talent level upgrade.

The amount of materials needed is in the following table.

Talent Level	Coins	Mob Drops	Talent Level-Up Materials		
			Talent Books	Weekly Boss Drops	Crown of Insight
1→2	12500	6 Common	3 Teachings	0	0
2→3	17500	3 Rare	2 Guides	0	0
3→4	25000	4 Rare	4 Guides	0	0
4→5	30000	6 Rare	6 Guides	0	0
5→6	37500	9 Rare	9 Guides	0	0
6→7	120000	4 Epic	4 Philosophies	1	0
7→8	260000	6 Epic	6 Philosophies	1	0
8→9	450000	9 Epic	12 Philosophies	2	0
9→10	700000	12 Epic	16 Philosophies	2	1

Grammy has a character with ascension level a_0, normal level l_0, and talent levels t_{10}, t_{20}, t_{30}, and she wants to upgrade the character to ascension level a, normal level l, and talent levels t_1, t_2, t_3. If the character gains experience after leveling up to normal level l, the extra experience gained is also considered as wasted experience and will not spend coins.

Grammy wants to ask you about the amount of materials needed. If there are multiple ways to use experience materials, choose a way that minimizes total experience wasted. If there are still multiple ways, choose a way that minimizes the amount of experience materials used.

Input

Each test contains multiple test cases. The first line contains a single integer T ($1 \le T \le 200\,000$) — the

number of test cases. Description of the test cases follows.

The only line of each test case contains 10 integers $a_0, l_0, t_{10}, t_{20}, t_{30}, a, l, t_1, t_2, t_3$ ($0 \le a_0 \le a \le 6, 1 \le l_0 \le l \le 90, 1 \le t_{i0} \le t_i \le 10$). It is guaranteed that the normal level can be reached while in the corresponding ascension level.

Output

For each test case print the answer in the following format.

Print 5 integers in the first line, indicating the number of coins needed, the number of local specialties needed, the number of boss drops needed, the number of weekly boss drops needed, and the number of "Crown of Insight" needed, respectively. It can be proved that under given constraints, the number of coins needed is an integer.

Print 3 integers in the second line, indicating the number of "Wanderer's Advice" needed, the number of "Adventurer's Experience" needed, the number of "Hero's Wit" needed, respectively.

Print 4 integers in the third line, indicating the number of Slivers, Fragments, Chunks, Gemstones needed, respectively.

Print 3 integers in the fourth line, indicating the number of common mob drops, rare mob drops, epic mob drops needed, respectively.

Print 3 integers in the fifth line, indicating the number of teachings, guides, and phylosophies needed, respectively.

Example

standard input	standard output
2 0 1 1 1 1 0 20 4 5 6 0 20 3 3 3 1 30 6 6 6	286535 0 0 0 0 1 0 6 0 0 0 0 18 42 0 9 39 0 340085 3 0 0 0 3 2 10 1 0 0 0 3 57 0 0 57 0

Problem L. Frog

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	512 mebibytes

Grammy spotted a frog at the border of a circular pillar. The pillar is centered at $(0, 0)$ and has radius 1. The frog can jump to a distance of exactly 1. Grammy wants the frog to move to her desired destination point at the border of the pillar. Please help Grammy to find a route for the frog with minimum number of jumps.

Note that the frog cannot be strictly inside the pillar at any time.

Input

The input contains multiple test cases.

The first line contains a single integer T ($1 \le T \le 10\,000$), indicating the number of test cases.

The only line of each testcase consists of two integers d_s, d_t ($0 \le d_s, d_t \le 359$), indicating that the frog's starting position is $(\cos\frac{\pi d_s}{180}, \sin\frac{\pi d_s}{180})$, and the frog's destination is $(\cos\frac{\pi d_t}{180}, \sin\frac{\pi d_t}{180})$.

Output

For each test case, print one or several lines in the following format.

The first line contains a single integer k, indicating the minimum number of jumps in this test case.

The next $k+1$ lines contain the landing points for the frog, including its starting point and its destination point.

The i-th of the next $k+1$ lines contains 2 real numbers, indicating the coordinates of the frog's i-th landing point.

Your answer will be considered correct if all the following conditions are satisfied:

- The number of jumps is minimal.
- The distance between the first landing point and the starting point is less than 10^{-6}.
- The distance between the last landing point and the destination point is less than 10^{-6}.
- The distance d between any two consecutive landing points satisfy $1 - 10^{-6} < d < 1 + 10^{-6}$.
- The segment connecting any two consecutive landing points have a distance $d > 1 - 10^{-6}$ to $(0, 0)$.

Example

standard input	standard output
3 0 0 0 90 180 0	0 1.0000000000 0.0000000000 2 1.0000000000 0.0000000000 1.0000000000 1.0000000000 0.0000000000 1.0000000000 4 -1.0000000000 0.0000000000 -1.0000000000 -1.0000000000 -0.0000000000 -1.0000000000 1.0000000000 -1.0000000000 1.0000000000 -0.0000000000

Problem M. A=B

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	512 mebibytes

Marisa has learned an interesting language called A=B. She finds that this language has the advantages of simple syntax, easy to learn and convenient to code.

Here is the user manual of A=B:

(Note that it may differ from the original game "A=B". So please read the statement carefully.)

- Instruction set

  A=B's instruction set includes:

  1. `string1=string2`
     Find the leftmost occurrence of `string1` in the string and replace it with `string2`.
  2. `string1=(return)string2`
     If `string1` is found, replace the entire string with `string2` and end the program immediately.

- Program structure

  An A=B program consists of several lines of instructions. Each line must include exactly one equal sign ('`=`').

  Following characters are reserved: '`=`', '`(`', '`)`'.

- Execution order

  1. Read the input string.
  2. Starting from the topmost line, find the first line that can be executed.
  3. If found, execute that line and go to step 2.
  4. If none is found, return the current string as output.

Pixiv ID: 68375845

Marisa once introduced A=B to Alice. However, "You called this a programming language? You can't even write a program that can check if string t is a substring of string s!" said Alice.

Now Marisa comes to you for help. She wants you to design an A=B program for this problem and show A=B's efficiency.

Your program needs to meet the following requirements given by Marisa:

- Read the input string (the input format is s`S`t. '`S`' is the separator. s and t are two **non-empty** strings consists of characters '`a`', '`b`', '`c`'.)
- If t is a substring of s, the program should return 1 as output, else return 0 as output.
- The character set that your program can use is {'`a`'∼'`z`', '`A`'∼'`Z`', '`0`'∼'`9`', '`=`', '`(`', '`)`'}. Remember, '`=`', '`(`', '`)`' are reserved character in A=B and you can't use it in `string1` or `string2`.
- In the previous instruction's format, the length of `string1` and `string2` should be at most 3.
- Suppose the length of the input string is L, the number of instruction's execution can't exceed $\max(2L^2, 50)$ and the length of string during execution can't exceed $2L + 10$.
- The number of instructions in your A=B program can't exceed 100.

Input

Input an integer Tid ($0 \le Tid \le 2 \times 10^9$). It is used for generating test sets and may be no use to you.

Output

Output your A=B program containing several lines of instructions.

The number of tests will not exceed 20. In each test, the checker will use Tid in the input file to generate several lines of input strings and their corresponding answers. Your A=B program is considered correct if and only if for each input string in all tests, your A=B program gives the correct output.

It's guaranteed that for each input string in all tests, the length L satisfies $3 \le L \le 1\,000$.

Examples

standard input	standard output
114514	514=(return)1 =514
1919810	S=Sakuya =(return)0
/* Sample input: caba Sample output: aabc Sample input: cbacab Sample output: aabbcc */	ba=ab ca=ac cb=bc
/* Sample input: bababb Sample output: b Sample input: aababbaa Sample output: a */	ba=ab ab= bb=b aa=a
/* Sample input: abc Sample output: true Sample input: cabc Sample output: false Sample input: ca Sample output: false */	b=a c=a aaaa=(return)false aaa=(return)true =(return)false
/* Sample input: 10111+111 Sample output: 11110 Sample input: 101+10110 Sample output: 11011 */	A0=0A A1=1A B0=0B B1=1B 0A=a 0B=b 1A=b 1B=ca A=a B=b ac=b bc=ca 0+=+A 1+=+B += 0c=1 1c=c0 c=1 a=0 b=1

Note

The first and second samples show how you should submit your answer. You should read an integer and then output your A=B program. This integer will be used to generate a fixed data set. The third to sixth samples give problems and their corresponding programs to help you get familiar with the A=B language. All these programs **may not** satisfy the requirements given by Marisa.

- Sample 1

  If the input is "`abcaSab`", the process will be:

  `abcSab` $\xrightarrow{line\ 2}$ `514abcSab` $\xrightarrow{line\ 1}$ `1`

  (Note that an empty string can be found at the beginning of any string.)

  If the input is "`abcaSccc`", the process will be:

  `abcSccc` $\xrightarrow{line\ 2}$ `514abcSccc` $\xrightarrow{line\ 1}$ `1`

  So this program will get a WRONG ANSWER verdict.

- Sample 2

  In the first instruction, `string2` is equal to "`Sakuya`", which has a length more than 3.

  So this program will get a WRONG ANSWER verdict.

- Sample 3

  Input: A string consists of '`a`', '`b`', '`c`'.

  Output: Sort the input in alphabetical order.

  Test1: `caba` $\xrightarrow{line\ 1}$ `caab` $\xrightarrow{line\ 2}$ `acab` $\xrightarrow{line\ 2}$ `aacb` $\xrightarrow{line\ 3}$ `aabc`

  Test2: `cbacab` $\xrightarrow{line\ 1}$ `cabcab` $\xrightarrow{line\ 2}$ `acbcab` $\xrightarrow{line\ 2}$ `acbacb` $\xrightarrow{line\ 1}$ `acabcb` $\xrightarrow{line\ 2}$

  `aacbcb` $\xrightarrow{line\ 3}$ `aabccb` $\xrightarrow{line\ 3}$ `aabcbc` $\xrightarrow{line\ 3}$ `aabbcc`

- Sample 4

  Input: A string consists of '`a`', '`b`'. The number of '`a`' and '`b`' are different.

  Output: The most common letter.

  Test1: `bababb` $\xrightarrow{line\ 1}$ `abbabb` $\xrightarrow{line\ 1}$ `ababbb` $\xrightarrow{line\ 1}$ `aabbbb` $\xrightarrow{line\ 2}$ `abbb` $\xrightarrow{line\ 2}$ `bb` $\xrightarrow{line\ 3}$ `b`

  Test2: `aababbaa` $\xrightarrow{\text{several times of line 1}}$ `aaaaabbb` $\xrightarrow{\text{several times of line 2}}$ `aa` $\xrightarrow{line\ 4}$ `a`

- Sample 5

  Input: A string consists of '`a`', '`b`', '`c`'.

  Output: Return true if the input contains exactly three letters. Otherwise, return false.

- Sample 6

  Input: Two binary numbers, separated by a '`+`'.

  Output: The sum of these two numbers. It is also in binary form.

  This program performs addition from low bit to high bit. For each step, the program first checks if the first number is empty (Line 15). Then consumes the lowest bit of the first number (Line 13 or 14) and then writes it at the back of the second number (Line 1 - 4). And then performs 1-bit addition (Line 5 - 8). Here, '`a`' and '`b`' represent '`0`' and '`1`', '`c`' represents the carry character. It may happen that in some steps the first number is non-empty but the second one is empty, so Line 9 - 10 are used to solve this case. Line 11 - 12 are to solve the carry in the previous step. At last, we transform '`a`', '`b`', '`c`' to their real number (Line 18 - 20) and solve the case that the second number is non-empty (Line 16 - 17).