6131 Arranging Heaps

A mining company extracts terbium, a rare metal used for constructing lightweight magnets, from river sand. They mine the Long River at N mining points, each of them identified by its distance from the river source. At each mining point, a relatively small but highly valued heap of mineral ore is extracted from the river.

To collect the mineral ore, the company regroups the N produced heaps into a smaller number of K heaps, each located at one of the initial mining points. The newly formed heaps are then collected by trucks.

To regroup the N heaps, they use a barge, which in practice can carry any amount of mineral ore because it is very large. The barge starts at the river source and can only travel downriver, so the heap produced at a mining point X can be taken to a mining point Y only if $Y > X$. Each heap is moved completely to another mining point, or not moved at all. The cost of moving a heap of weight W from a mining point X to a mining point Y is $W \times (Y - X)$. The total cost of the regrouping is the sum of the costs for each heap movement. Notice that a heap which is not moved has no influence on the total cost.

Given the values for N and K, the N mining points, and the weight of the heap each mining point produced, write a program that calculates the minimum total cost to regroup the N initial heaps into K heaps.

Input

Each test case is described using several lines. The first line contains two integers N and K denoting respectively the number of initial heaps and the desired number of heaps after regrouping ($1 \leq K < N \leq 1000$). Each of the next N lines describes one of the initial heaps with two integers X and W indicating that the mining point X produced a heap of weight W ($1 \leq X, W \leq 10^6$). Within each test case the heaps are given in strictly ascending order considering their mining points.

Sample Output

For each test case output a line with an integer representing the minimum total cost to regroup the N initial heaps into K heaps.

Sample Input

```
3 1
20 1
30 1
40 1
3 1
11 3
12 2
13 1
6 2
10 15
12 17
16 18
18 13
```

```
30 10
32 1
6 3
10 15
12 17
16 18
18 13
30 10
32 1
```

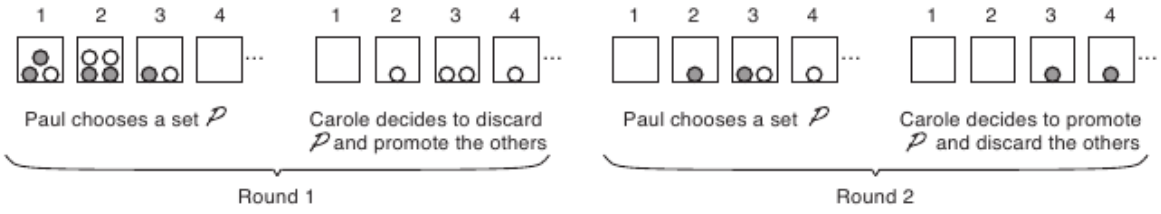
Sample Output

```
30
8
278
86
```

6132 Boxes and Stones

Paul and Carole like to play a game with S stones and B boxes numbered from 1 to B. Before beginning the game they arbitrarily distribute the S stones among the boxes from 1 to $B-1$, leaving box B empty. The game then proceeds by rounds. At each round, first Paul chooses a subset P of the stones that are in the boxes; he may choose as many stones as he wants from as many boxes as he wants, or he may choose no stones at all, in which case P is empty. Then, Carole decides what to do next: she can either *promote* the subset P and *discard* the remaining stones (that is, those stones not chosen by Paul in the first step); or she may *discard* the subset P and *promote* the remaining stones.

To *promote* a given subset means to take each stone in this subset and move it to the box with the next number in sequence, so that if there was a stone in this subset inside box b, it is moved to box $b+1$. To *discard* a given subset means to remove every stone in this subset from its corresponding box, so that those stones are not used in the game for the remaining rounds. The figure below shows an example of the first two rounds of a game.



Paul and Carole play until at least one stone reaches box number B, in which case Paul wins the game, or until there are no more stones left in the boxes, in which case Carole wins the game. Paul is a very rational player, but Carole is a worthy rival because she is not only extremely good at this game, but also quite lucky. We would like to know who is the best player, but before that we must first understand how the outcome of a game depends on the initial distribution of the stones. In particular, we would like to know in how many ways the S stones can initially be distributed among the first $B-1$ boxes so that Carole can be certain that she can win the game if she plays optimally, even if Paul never makes a mistake.

Input

Each test case is described using one line. The line contains two integers S ($1 \leq S \leq 200$) and B ($2 \leq B \leq 100$), representing respectively the number of stones and the number of boxes in the game.

Output

For each test case output a line with an integer representing the number of ways in which the S stones may be distributed among the first $B-1$ boxes so that Carole is certain that she can win the game. Because this number can be very large, you are required to output the remainder of dividing it by 10^9+7.

Sample Input

```
2 3
8 4
42 42
```

Sample Output

```
2
0
498467348
```

6133 Cellphone Typing

A research team is developing a new technology to save time when typing text messages in mobile devices. They are working on a new model that has a complete keyboard, so users can type any single letter by pressing the corresponding key. In this way, a user needs P keystrokes to type a word of length P.

However, this is not fast enough. The team is going to put together a dictionary of the common words that a user may type. The goal is to reduce the average number of keystrokes needed to type words that are in the dictionary. During the typing of a word, whenever the following letter is uniquely determined, the cellphone system will input it automatically, without the need for a keystroke. To be more precise, the behavior of the cellphone system will be determined by the following rules:

1. The system never guesses the first letter of a word, so the first letter always has to be input manually by pressing the corresponding key.

2. If a non-empty succession of letters $c_1c_2\ldots c_n$ has been input, and there is a letter c such that every word in the dictionary which starts with $c_1c_2\ldots c_n$ also starts with $c_1c_2\ldots c_nc$, then the system inputs c automatically, without the need of a keystroke. Otherwise, the system waits for the user.

For instance, if the dictionary is composed of the words '`hello`', '`hell`', '`heaven`' and '`goodbye`', and the user presses '`h`', the system will input '`e`' automatically, because every word which starts with '`h`' also starts with '`he`'. However, since there are words that start with '`hel`' and with '`hea`', the system now needs to wait for the user. If the user then presses '`l`', obtaining the partial word '`hel`', the system will input a second '`l`' automatically. When it has '`hell`' as input, the system cannot guess, because it is possible that the word is over, or it is also possible that the user may want to press '`o`' to get '`hello`'. In this fashion, to type the word '`hello`' the user needs three keystrokes, '`hell`' requires two, and '`heaven`' also requires two, because when the current input is '`hea`' the system can automatically input the remainder of the word by repeatedly applying the second rule. Similarly, the word '`goodbye`' needs just one keystroke, because after pressing the initial '`g`' the system will automatically fill in the entire word. In this example, the average number of keystrokes needed to type a word in the dictionary is then $(3 + 2 + 2 + 1)/4 = 2.00$.

Your task is, given a dictionary, to calculate the average number of keystrokes needed to type a word in the dictionary with the new cellphone system.

Input

Each test case is described using several lines. The first line contains an integer N representing the number of words in the dictionary ($1 \leq N \leq 10^5$). Each of the next N lines contains a non-empty string of at most 80 lowercase letters from the English alphabet, representing a word in the dictionary. Within each test case all words are different, and the sum of the lengths of all words is at most 10^6.

Output

For each test case output a line with a rational number representing the average number of keystrokes needed to type a word in the dictionary. The result must be output as a rational number with exactly two digits after the decimal point, rounded if necessary.

Sample Input

```
4
hello
hell
heaven
goodbye
3
hi
he
h
7
structure
structures
ride
riders
stress
solstice
ridiculous
```

Sample Output

```
2.00
1.67
2.71
```

6134 Different Digits

The inhabitants of Nlogonia are very superstitious. One of their beliefs is that street house numbers that have a repeated digit bring bad luck for the residents. Therefore, they would never live in a house which has a street number like 838 or 1004.

The Queen of Nlogonia ordered a new seaside avenue to be built, and wants to assign to the new houses only numbers without repeated digits, to avoid discomfort among her subjects. You have been appointed by Her Majesty to write a program that, given two integers N and M, determines the maximum number of houses that can be assigned street numbers between N and M, inclusive, that do not have repeated digits.

Input

Each test case is described using one line. The line contains two integers N and M, as described above ($1 \leq N \leq M \leq 5000$).

Output

For each test case output a line with an integer representing the number of street house numbers between N and M, inclusive, with no repeated digits.

Sample Input

```
87 104
989 1022
22 25
1234 1234
```

Sample Output

```
14
0
3
1
```

6135 Environment Protection

Arsenic & Cyanide Mining (ACM) is a corporation that has recently decided to start developing its mines in the lands near your hometown. As a member of the citizen's regulatory committee for ACM's operations, your task is to control how much the corporation can mine from those lands, so that you get to keep the jobs and other benefits without sacrificing the environment and the health of the local residents.

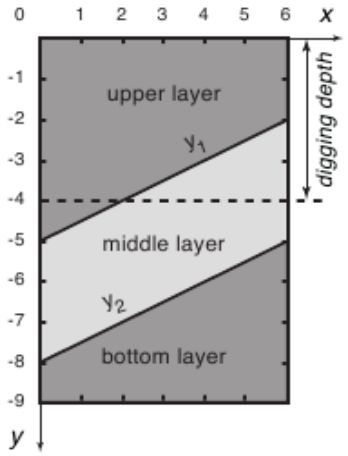
The ACM has plans to mine several rectangular patches of land. A patch of land has width W, can be dug up to a maximum depth D, and has a flat surface which we consider to be at depth 0. The minerals in a patch are organized in three layers, which may vary in their depth along the width of the patch, but always have the same profile along its whole length. This is why the ACM is only interested in the profile along the width of each patch, and has performed exploratory work in order to precisely determine its shape. As a result, they discovered that the two interfaces between the three layers of minerals can be represented by two functions $y_1(x)$ and $y_2(x)$, where the first describes the boundary between the top layer and the middle layer, and the second describes the boundary between the middle layer and the bottom layer. These functions are always such that

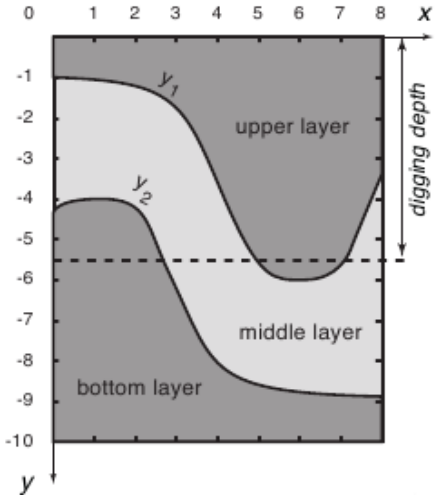
$$-D < y_2(x) < y_1(x) < 0 \qquad \text{for} \qquad 0 \le x \le W,$$

so that the layers' boundaries never touch each other. Besides, each function has the form $y_i(x) = p_i(x)/q_i(x)$, where

$$p_i(x) = \sum_{k=0}^{K} P_{i,k} x^k \qquad \text{and} \qquad q_i(x) = \sum_{k=0}^{K} Q_{i,k} x^k,$$

for $i = 1, 2$ and a certain integer K. The figure below shows the profiles of two patches of land in the way the ACM represents them. The patch on the left has width $W = 6$ and depth $D = 9$, while the patch on the right has $W = 8$ and $D = 10$. The boundaries of the layers of each patch are described by the functions defined below them.





$$y_1(x) = \frac{-10 + 1x}{2 + 0x}$$
$$y_2(x) = \frac{-16 + 1x}{2 + 0x}$$

$$y_1(x) = \frac{-1392 + 864x - 216x^2 + 24x^3 - 1x^4}{1312 - 864x + 216x^2 - 24x^3 + 1x^4}$$
$$y_2(x) = \frac{-73 + 36x - 54x^2 + 36x^3 - 9x^4}{17 - 4x + 6x^2 - 4x^3 + 1x^4}$$

The ACM will dig everything in a patch of land up to a certain digging depth d, and then sell all the minerals thus obtained to make a profit. However, the minerals in the top and the bottom layers are essentially worthless, so the profit of the whole operation comes exclusively from those minerals in the middle layer. In fact, the profit is proportional to the area A of the middle layer in the profile that is at depth at most d. Given the description of a patch of land and an integer A, you would like to know the digging depth d you should allow the ACM to dig the patch so that they get an area of minerals from the middle layer in the profile of exactly A. In the figure above you can see the answer for the two test cases in the sample input. For the patch on the left, in order to get an area $A = 4$ the digging depth must be $d = 4.00000$, while for the patch on the right an area $A = 14$ requires a digging depth $d = 5.51389$.

Input

Each test case is described using five lines. The first line contains four integers W, D, A and K, where W is the width of the patch of land the ACM wants to mine ($1 \leq W \leq 8$), D is its depth ($1 \leq D \leq 10$), A is the area of the middle layer in the profile that the ACM must get ($1 \leq A \leq W \times D$), and K allows the definition of the interfaces $y_1(x)$ and $y_2(x)$ as explained above ($0 \leq K \leq 8$). Each of the other lines contains $K + 1$ integers between -10^8 and 10^8, inclusive. The second line contains the coefficients of $p_1(x)$ from $P_{1,0}$ to $P_{1,K}$. The third line contains the coefficients of $q_1(x)$ from $Q_{1,0}$ to $Q_{1,K}$. The fourth line contains the coefficients of $p_2(x)$ from $P_{2,0}$ to $P_{2,K}$. The fifth line contains the coefficients of $q_2(x)$ from $Q_{2,0}$ to $Q_{2,K}$. Within each test case A is strictly less than the total area of the middle layer in the profile, and there exists a single value d such that a digging depth d yields an area of minerals from the middle layer in the profile of exactly A. Besides, $q_1(x) = 0$, $q_2(x) = 0$ and $-D < y_2(x) < y_1(x) < 0$, for $0 \leq x \leq W$.

Output

For each test case output a line with a rational number representing the depth d that the ACM should be allowed to dig the patch of land so that they get an area of minerals from the middle layer in the profile of exactly A. The result must be output as a rational number with exactly five digits after the decimal point, rounded if necessary.

Sample Input

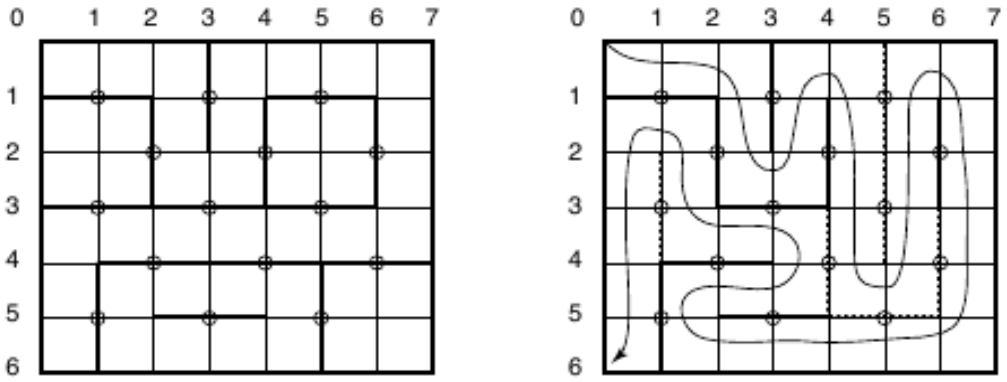
```
6 9 4 1
-10 1
2 0
-16 1
2 0
8 10 14 4
-1392 864 -216 24 -1
1312 -864 216 -24 1
-73 36 -54 36 -9
17 -4 6 -4 1
```

Sample Output

```
4.00000
5.51389
```

6136 Fix the Pond

A shrimp farm uses a rectangular pond built as a grid with $2N$ rows and $2N+1$ columns of square cells, for a given integer N. Each cell side is one meter long. The pond has exactly $(2N-1) \times N$ barriers of length two meters, used to temporarily isolate smaller sections inside the pond for breeding different kinds of shrimp. The barriers have their middle points fixed precisely at the integer coordinates (a, b), for all $0 < a < 2N$ and $0 < b < 2N+1$, where both a and b are odd, or both are even. Each barrier can be rotated around its middle point to change the pond configuration; however, by being rotated, a barrier switches between only two possible positions, always parallel to the pond sides, vertical or horizontal. The left part of the figure below shows a pond configuration, with $N = 3$.



At the end of every season the pond is closed for maintenance and cleaning. It must then be reconfigured so that a special machine can sweep the pond floor. The machine starts its work at the top left cell, and needs to pass through every cell exactly once, finishing in the bottom left cell. The right part of the figure shows one such reconfiguration, where six barriers were switched. For this example, though, four barrier switches would have been enough.

You must write a program that given a pond configuration, determines the minimum number of barrier switches needed to reconfigure the pond as specified above. There is always at least one possible way to reconfigure the pond as specified.

Input

Each test case is described using several lines. The first line contains an integer N indicating that the pond has $2N$ rows and $2N+1$ columns ($1 \leq N \leq 300$). Each of the next $2N-1$ lines contains a string of N characters describing the orientation of the barriers. In the i-th line, the j-th character indicates the orientation of the barrier whose middle point is at coordinates $(i, 2j-1)$ if i is odd, or $(i, 2j)$ if i is even, for $i = 1, 2, \ldots, 2N-1$ and $j = 1, 2, \ldots, N$. The character is the uppercase letter '`V`' if the orientation is vertical, or the uppercase letter '`H`' if it is horizontal.

Output

For each test case output a line with an integer representing the minimum number of barrier switches needed to reconfigure the pond as specified.

Sample Input

```
3
HVH
VVV
HHH
HHH
VHV
1
H
1
V
```

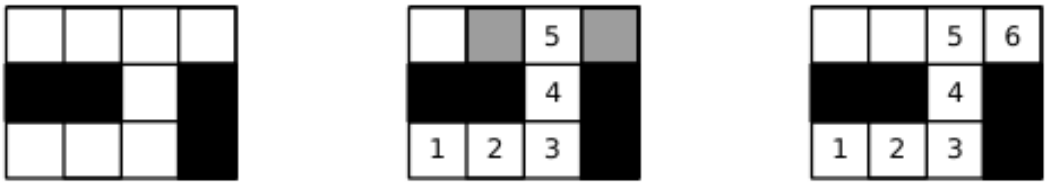
Sample Output

```
4
0
1
```

6137 Game of Tiles

The Game of Tiles is a game for two players played over a rectangular board in the form of a table of R rows and C columns of square cells called tiles. At the beginning of the game, some of the tiles may be painted black and the rest remain white. Then, Player 1 and Player 2 alternate turns making a move and the first one that cannot make a valid move loses the game. The first move of the game is done by Player 1 and consists of choosing a white tile and writing the number 1 on it. After that, each subsequent move i consists of writing number i on an unused white tile that is adjacent horizontally or vertically (but not diagonally) to the tile numbered $i - 1$. Note that Player 1 always writes odd numbers and Player 2 always writes even numbers.

The following figure shows three examples of possible configurations of a board with $R = 3$ and $C = 4$ during a game. On the left it shows the initial configuration. On the center it shows an intermediate state, where cells in gray mark the possible moves for Player 2. And on the right it shows the configuration when the game is won by Player 2, who chose the appropriate move.



Your task is to write a program that given the initial configuration of the board, determines which player will win, if both of them play optimally.

Input

Each test case is described using several lines. The first line contains two integers R and C representing respectively the number of rows and columns of the board ($1 \leq R, C \leq 50$). The i-th of the next R lines contains a string B_i of C characters that describes the i-th row of the initial board. The j-th character of B_i is either '`.`' (dot) or the uppercase letter '`X`', representing that the tile at row i and column j is respectively white or black. Within each test case at least one of the tiles is white.

Output

For each test case output a line with an integer representing the number of the player (1 or 2) who will win the game if both of them play optimally.

Sample Input

```
3 4
....
XX.X
...X
3 4
....
.X.X
...X
```

```
3 4
....
.X.X
....
1 1
.
1 11
....X......
```

Sample Output

```
2
1
1
1
2
```

6138 Hours and Minutes

Heidi has a discrete analog clock in the shape of a circle, as the one in the figure. Two hands rotate around the center of the circle, indicating hours and minutes. The clock has 60 marks placed around its perimeter, with the distance between consecutive marks being constant.

The minute hand moves from its current mark to the next exactly once every minute. The hour hand moves from its current mark to the next exactly once every 12 minutes, so it advances five marks each hour.

We consider that both hands move discretely and instantly, which means they are always positioned exactly over one of the marks and never in between marks.

At midnight both hands reach simultaneously the top mark, which indicates zero hours and zero minutes. After exactly 12 hours or 720 minutes, both hands reach the same position again, and this process is repeated over and over again. Note that when the minute hand moves, the hour hand may not move; however, when the hour hand moves, the minute hand also moves.

Heidi likes geometry, and she likes to measure the minimum angle between the two hands of the clock at different times of the day. She has been writing some measures down, but after several years and a long list, she noticed that some angles were repeated while some others never appeared. For instance, Heidi's list indicates that both at three o'clock and at nine o'clock the minimum angle between the two hands is 90 degrees, while an angle of 65 degrees does not appear in the list. Heidi decided to check, for any integer number A between 0 and 180, if there exists at least one time of the day such that the minimum angle between the two hands of the clock is exactly A degrees. Help her with a program that answers this question.

Input

Each test case is described using one line. The line contains an integer A representing the angle to be checked ($0 \leq A \leq 180$).

Output

For each test case output a line containing a character. If there exists at least one time of the day such that the minimum angle between the two hands of the clock is exactly A degrees, then write the uppercase letter '`Y`'. Otherwise write the uppercase letter '`N`'.

Sample Input

```
90
65
66
67
128
0
180
```

Sample Output

```
Y
N
Y
N
N
Y
Y
```

6139 Interval Product

It's normal to feel worried and tense the day before a programming contest. To relax, you went out for a drink with some friends in a nearby pub. To keep your mind sharp for the next day, you decided to play the following game. To start, your friends will give you a sequence of N integers $X_1, X_2, \ldots, XN$. Then, there will be K rounds; at each round, your friends will issue a command, which can be:

- a *change* command, when your friends want to change one of the values in the sequence; or
- a *product* command, when your friends give you two values I, J and ask you if the product $X_I \times X_{I+1} \times \ldots \times X_{J-1} \times X_J$ is positive, negative or zero.

Since you are at a pub, it was decided that the penalty for a wrong answer is to drink a pint of beer. You are worried this could affect you negatively at the next day's contest, and you don't want to check if Ballmer's peak theory is correct. Fortunately, your friends gave you the right to use your notebook. Since you trust more your coding skills than your math, you decided to write a program to help you in the game.

Input

Each test case is described using several lines. The first line contains two integers N and K, indicating respectively the number of elements in the sequence and the number of rounds of the game ($1 \leq N, K \leq 10^5$). The second line contains N integers X_i that represent the initial values of the sequence ($-100 \leq X_i \leq 100$ for $i = 1, 2, \ldots, N$). Each of the next K lines describes a command and starts with an uppercase letter that is either '`C`' or '`P`'. If the letter is '`C`', the line describes a *change* command, and the letter is followed by two integers I and V indicating that X_I must receive the value V ($1 \leq I \leq N$ and $-100 \leq V \leq 100$). If the letter is '`P`', the line describes a *product* command, and the letter is followed by two integers I and J indicating that the product from X_I to X_J, inclusive must be calculated ($1 \leq I \leq J \leq N$). Within each test case there is at least one *product* command.

Output

For each test case output a line with a string representing the result of all the *product* commands in the test case. The i-th character of the string represents the result of the i-th *product* command. If the result of the command is positive the character must be '`+`' (plus); if the result is negative the character must be '`-`' (minus); if the result is zero the character must be '`0`' (zero).

Sample Input

```
4 6
-2 6 0 -1
C 1 10
P 1 4
C 3 7
P 2 2
C 4 -5
P 1 4
5 9
1 5 -2 4 3
```

```
P 1 2
P 1 5
C 4 -5
P 1 5
P 4 5
C 3 0
P 1 5
C 4 -5
C 4 -5
```

Sample Output

```
0+-
+-+-0
```

6140 Joining Couples

Air traffic regulations in Nlogonia require that each city must register exactly one outbound flight to another city. Passengers can use this flight only in the direction registered, that is, there may be a flight registered from city X to city Y and no flight registered from city Y to city X. Thus, the number of registered flights is equal to the number of cities. This rule, as one can imagine, makes air travel somewhat complicated, but tradition and a strong ruling by the Queen makes any changes difficult. Besides, some companies even make a profit from the problems caused by the rule.

The Association for Couple Matching (ACM) is setting up a new service to help customers find their long lasting soulmates: the Internet Connecting Program for Couples (ICPC). The service consists of computing the minimum total number of flights a couple needs to take to meet one another (perhaps in a city where neither of them lives in). Assuming the couple's starting cities are A and B, the agency will try to find a city C such that C is reachable by air travel from both A and B, and the sum of the number of flights needed to go from A to C and the number of flights needed to go from B to C is minimized. Note that C may be equal to A or B or both.

You will be given the list of all available flights, and a list of queries consisting of pairs of cities where the members of a couple live. For each query, you must compute the minimum total number of flights that are needed for them to meet.

Input

Each test case is described using several lines. The first line contains an integer N representing the number of cities ($2 \leq N \leq 10^5$). Cities are identified by different integers from 1 to N. The second line contains N integers F_i, where F_i indicates that the registered outbound flight from city i is to city F_i ($1 \leq F_i \leq N$, $F_i = i$ for $i = 1, 2, \ldots, N$). The third line contains an integer Q representing the number of queries ($1 \leq Q \leq 10^5$). Each of the next Q lines describes a query with two integers A and B indicating the couple's starting cities ($1 \leq A, B \leq N$). Within each test case, if it is possible to travel by air from city X to city Y, the maximum number of flights needed to do so is 10^4.

Output

For each test case output Q lines. In the i-th line write an integer with the answer to the i-th query. If the corresponding couple can meet by air travel, write the minimum total number of flights that the couple must take to meet one another; if it is impossible for the couple to meet by air travel, write the number '`-1`'.

Sample Input

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

```
4 7
7 4
6 2
2 1
```

Sample Output

```
1
2
0
-1
3
3
-1
1
```