The 2020 ICPC Caribbean Finals Qualifier

Real contest problemset

Problem set developers:

Alberto Gonzalez Rosales (Cuba)
Alexander Bestard (Cuba)
Ariel Cruz Cruz (Cuba)
Aurora Gil Pons (Cuba)
Carlos Joa (Dominican Republic)
Ernesto David Peña Herrera (Cuba)
Jose Carlos Gutiérrez Pérez (Cuba)
Marcelo Fornet Fornés (Cuba)
Reynaldo Gil Pons (Cuba)
Roberto Abreu (Dominican Republic)
Rubén Alcolea Núñez (Cuba)

June 26^{th}, 2021

Problem A. Lost Key

Juan designed a very unorthodox encryption algorithm. This works as follows: Given a string $A = a_0a_1a_2 \dots a_{n-1}$, and a key $K = k_0k_1k_2 \dots k_{m-1}$, the algorithm encrypts the string A, in a resultant string $B = b_0b_1b_2 \dots b_{n-1}$, where $b_i = a_i + k_{i \bmod m}$. We define the sum of two characters x' and y' as the letter that is at position z' in the alphabet, where z' is the result of summing the number of the position of the letter x' in the alphabet plus the number of the position of the letter y' in the alphabet modulo 26 (cyclically). All positions in the alphabet are indexed starting from 0. Example: The encryption of the string $A =$ *"caribe"* with the key $K =$ *"icpc"* will return the string $B =$ *"kcgkjg"*, through the following steps:

1. Letter **k** is at position $10 = (2 + 8)_{\bmod 26}$, **c** is at position 2 and **i** is at position 8.
2. Letter **c** is at position $2 = (0 + 2)_{\bmod 26}$, **a** is at position 0 and **c** is at position 2.
3. Letter **g** is at position $6 = (17 + 15)_{\bmod 26}$, **r** is at position 17 and **p** is at position 15.
4. Letter **k** is at position $10 = (8 + 2)_{\bmod 26}$, **i** is at position 8 and **c** is at position 2.
5. Letter **j** is at position $9 = (1 + 8)_{\bmod 26}$, **b** is at position 1 and **i** is at position 8.
6. Letter **g** is at position $6 = (4 + 2)_{\bmod 26}$, **e** is at position 4 and **c** is at position 2.

Juan started encrypting a text yesterday and today he needs to continue. However, he has lost K. Help Juan to find the key, so he can continue working. Juan will give you a string, A, and its corresponding encryption, B.

Input

First line contains two strings, composed by lowercase letters, separated by a space, A and B ($|A| = |B|, 1 \le |A| \le 100$). Second line contains $m (1 \le m \le 4, m \le |A|)$, the size of the lost key.

Output

Print the key, a string of lowercase letters, if you could find it, or -1, if it is impossible to get B from A with a key of size m.

Example

standard input	standard output
caribe kcgkjg 4	icpc
latin lbfop 3	-1
zombie zamniq 2	am

Problem B. Maximum GCD Sum

For two positive integers a and b, let $gcd(a, b)$ be the greatest common divisor of a and b, that is, the largest integer that divides both a and b (leaving no remainder). Let's extend the definition of gcd to a sequence $S = s_1, s_2, ..., s_k$, where $k > 2$, recursively as follows: $gcd(S) = gcd(s_1, gcd(s_2, ..., s_{k-1}, s_k))$. That is, gcd of a sequence is the gcd of the first element in the sequence and the gcd of the rest of the sequence. For a sequence S consisting of at least two integers, let's define a function $F(S) = |S| \times gcd(S)$, where $|S|$ is the length of the sequence. Given an array A, determine what is the maximum possible value of $F(S)$ where S is a subsequence of A with at least two elements. A sequence S is a subsequence of an array A if S can be obtained from A by erasing several (possibly zero or all) elements.

Input

The first line of the input contains integer n ($2 \leq n \leq 10^5$), representing the length of the array A. The second line contains n positive integers a_1, a_2, ..., a_n not exceeding 10^6, corresponding to the elements in the sequence A.

Output

Print a single line with the maximum value of $F(S)$ where S is a subsequence of array A with at least two elements.

Example

standard input	standard output
7 10 6 1 30 5 42 6	24
2 3 10	2
6 1 20 30 50 70 5	40

Problem C. Find the point

Consider an infinite 2-dimensional grid with some marked cells. We are interested in finding the minimum sum of Manhattan distances from all marked cells to some not marked cell. We define Manhattan distance between two cells with coordinates a_1, b_1 and a_2, b_2 as $|a_1 - a_2| + |b_1 - b_2|$.

Input

The first line of input contains a single integer $n (1 \leq n \leq 100000)$, the number of marked cells in the grid. Next n lines contains the coordinates a_i and b_i of marked cells in the grid $(-2^{30} \leq a_i, b_i \leq 2^{30})$.

Output

You should output a single line with the minimum sum of Manhattan distances.

Example

standard input	standard output
4 1 -3 0 1 -2 1 1 -1	10
5 2 2 0 2 4 2 2 0 4 0	11
9 0 0 0 1 0 2 1 0 1 1 1 2 2 0 2 1 2 2	24

Problem D. Bracket Sequence

Given two sequences of brackets a and b of length n, such that a is not greater than b, you must count the number of balanced bracket sequences of length n not less than a and not greater than b. We assume comparisons are made in a lexicographical sense. We define balanced bracket sequences as follows:

- The empty sequence is a balanced bracket sequence.
- If s is a balanced bracket sequence, then so is (s).
- If s and t are balanced bracket sequences, then so is st.

Given two sequences of brackets x, y of length n, we say x is lexicographically smaller than y if they share a prefix of size $t \in [0, n-1]$ and $x[t+1] < y[t+1]$. You may assume character '(' is smaller than character ')'.

Input

The first line of input contains the integer n ($1 \leq n \leq 10^3$), the size of the sequences. Next two lines contain respectively sequences a and b, composed only by characters '(' and ')'. It is guaranteed that sequence b is not lexicographically smaller than a.

Output

Print a line with the answer. Since this number may be very large you should output the answer modulo $10^9 + 7$.

Example

standard input	standard output
4 ((((()))	2
3 ()))()	0
12 ((()))(()()) ((()))(()())	1
38 (((((((((((((((((((((((((((((((((((((( ))))))))))))))))))))))))))))))))))))))	767263183

Problem E. Counting Multiples

You are given an array A with n integers. You are asked to perform q queries on this array. Queries can be of two types:

- **q i j:** Within the interval $[i, j]$, find the number of **distinct** multiples of 2 and 3, but not both.
- **u p v:** Update the value of the array at index p with the value v, that is set $A[p] = v$.

You must perform these queries in the order given in the input. In particular, each query of type **'q'** must consider all updates of type **'u'** that precede the query. Write a program to solve this problem efficiently.

Input

First line contains two integers n and q ($1 \leq n \leq 10^5$, $1 \leq q \leq 10^5$), the number of elements of the array and the number of queries to answer, respectively. The second line contains n integers representing the initial content of the array. Each $A[i]$ value of the array satisfies the constraint $1 \leq A[i] \leq 10^9$. The next q lines contain a character **'q'** or **'u'** and two integers, separated by spaces.

- If the character is **'q'**, the query is in the format **q i j**, where $1 \leq i \leq j \leq N$.
- If the character is **'u'**, the query is in the format **u p v**, where $1 \leq p \leq N$ and $1 \leq v \leq 10^9$.

Output

For each query of type **'q'**, print a line with the answer. You must answer the queries in the order given in the input.

Example

<table>
<tr><th>standard input</th><th>standard output</th></tr>
<tr><td><pre>5 6
12 4 2 3 8
q 1 2
u 3 30
q 3 5
u 1 10
u 3 9
q 1 5</pre></td><td><pre>1
2
5</pre></td></tr>
<tr><td><pre>5 9
10 2 32 18 54
q 1 5
u 1 5
q 1 5
u 2 4
q 1 5
u 3 3
q 1 5
u 4 2
q 1 5</pre></td><td><pre>3
2
2
2
3</pre></td></tr>
</table>

Problem F. Antipalindromes

A binary string is considered antipalindrome if it is not empty and it is fulfilled for each i that $s_i \neq s_{n-i+1}$. For example, 01, 0011, 101010 and 011001 are antipalindromes, but 0, 1110, 11, are not. Given a binary string, determine how many substrings are antipalindromes. If the amount of antipalindromes substrings is greater than 10^5, print 10^5 instead.

Input

A line with a number $n(1 \leq n \leq 10^5)$, the size of the string. In second line, a binary string of n characters.

Output

Print the amount of substrings that are antipalindromes, in case being greater than 10^5, print 100000.

Example

standard input	standard output
4 0101	4
4 1001	2

Problem G. Evolution

You are given a list of n numbers. Each number will be 0 or 1. There must be applied m iterations of the same procedure. For each iteration the value of each number will change according to:

- If the two adjacent numbers in the previous iteration are equal, the value of the number will be 0.
- Otherwise, the value of the number will be 1.

As the first and last values of the list have only one adjacent number, consider 0 to be the other adjacent number.

Input

First line contains 2 integers n and m ($1 \leq n \leq 1000$, $1 \leq m \leq 1000$) the amount of numbers on the list and the amount of iterations to applied respectively. Second line contains n numbers separated by spaces.

Output

Print the list of numbers after m iterations.

Example

standard input	standard output
8 1 1 0 0 0 0 1 0 0	0 1 0 0 1 0 1 0
8 2 1 1 1 0 1 1 1 1	0 0 0 0 0 1 1 0
5 3 0 0 0 1 0	1 0 1 0 0

Problem H. Round Table

Consider $2 \cdot n$ people with n enemy pairs. Each person has **exactly one** enemy and the enmity is bi-directional, meaning that, if a is the enemy of b, then b is the enemy of a. Enemy pairs are defined as follows, for all j ($1 \leq j \leq n$), the person $2 \cdot j - 1$ and the person $2 \cdot j$ are enemies. You want to calculate in how many ways it is possible to distribute them on a large circular table with $2 \cdot n$ seats, so that there are no 2 enemies sitting next to each other. The seats are numbered from 1 to $2 \cdot n$. The seats of positions i and $i + 1$ are adjacent for all $1 \leq i < 2 \cdot n$. The seats 1 and $2 \cdot n$ are also adjacent. Two distributions are considered different if they have at least one seat occupied by different people in each distribution. For example: We have 4 people and the enemy pairs are $(1, 2)$ and $(3, 4)$. There are 8 different ways to place them around the table:

- $(1, 3, 2, 4)$
- $(2, 3, 1, 4)$
- $(3, 1, 4, 2)$
- $(4, 1, 3, 2)$
- $(1, 4, 2, 3)$
- $(2, 4, 1, 3)$
- $(3, 2, 4, 1)$
- $(4, 2, 3, 1)$

Input

The first line of the input contains an integer t ($1 \leq t \leq 10^5$), the number of cases to process. The next t lines describe the test cases, each with an integer n ($1 \leq n \leq 10^5$), the number of enemy pairs.

Output

For each test case you should print a line with the number of possible distributions. Since this number can be very large, print the remainder left by dividing it by $10^9 + 7$.

Example

standard input	standard output
4 1 2 4 99999	0 8 11904 720034323

Problem I. Weighted Components

There is a matrix X with $n \times m$ cells where each cell has an associated weight. A subset S of cells is considered connected if, between every pair of cells in S, there is a path of neighboring cells, also in S, that joins them. Two cells are considered neighboring if they share a side. The weight of a subset S is the sum of the weights of the cells that are in S. You are asked to answer several queries, where each query has a weight of k. For each query, determine whether there is a connected subset whose weight is between k and $2 \cdot k$ (inclusive).

Input

First line contains two integers n and m ($1 \leq n \times m \leq 10^5$). Then n lines each with m integers. The j-th number in the i-th line represents the weight of the cell $X[i, j]$ ($1 \leq X[i, j] \leq 10^9$). Then a line with an integer q ($1 \leq q \leq 10^5$), the number of queries and then q lines with an integer k ($1 \leq k \leq 10^{14}$), indicating the weight of each query.

Output

For each question, print "YES" if the solution exists, or "NO" otherwise.

Example

standard input	standard output
3 4 1 1 9 2 9 9 9 9 1 1 9 1 10 1 2 3 4 5 6 7 8 9 10	YES YES NO NO YES YES YES YES YES YES
5 5 8 9 1 9 4 941 942 8 4 2 5 939 921 978 914 932 1 8 9 4 7 3 6 5 10 4 450 58 20 10	NO NO YES YES

Problem J. Coin

Two players, A and B, compete on an undirected graph with N nodes numbered 1 to N. Player A must move a coin from node 1 to node N. On each turn, player B selects k nodes, and locks them; however, she cannot choose node N nor the node where the coin is currently located at. Then, player A may move the coin to any adjacent (unlocked) node of the coin. Locked nodes are kept locked until the end of the turn and then are unlocked. We want to find the smallest k such that player B can prevent A from moving the coin to node N.

Input

The first line contains two integers N and M ($2 \le N \le 10^5, 0 \le M \le 10^5$), the number of nodes and the number of edges respectively. Then M lines, each with two integers u and v ($1 \le u, v \le N$) the description of each edge.

Output

Print the value of the smallest k such that player B can prevent A from moving the coin to node N. If there is no such k, print -1.

Example

standard input	standard output
6 6 1 2 1 3 2 4 3 5 4 6 5 6	1
6 8 1 2 1 3 2 4 3 5 4 6 5 6 3 4 2 5	2
4 6 1 2 1 3 1 4 2 3 2 4 3 4	-1

Problem K. String Operations

There is a list A of n strings and q queries of the following types:

- 1 i s: Set string at position i to s. That is, $A[i] = s$.
- 2 i j s: Determine whether there is any string located in the interval $[i, j]$ that is a prefix of the string s.
- 3 i j s: Determine whether the string s is a prefix of some string located in the interval $[i, j]$.

For all queries, it holds that $1 \leq i \leq j \leq n$ and s is a string of lowercase English characters.

Input

The first line contains an integer $1 \leq n \leq 10^5$, the number of strings in A. Then, n lines follow, where the i-th line corresponds to the i-th string in A. The next line contains an integer $1 \leq q \leq 10^5$, the number of queries to process. Then, q lines follow, each having a query in the format described above. The sum of the lengths of the strings present in array A and in all queries does not exceed $5 \cdot 10^5$.

Output

For every query of type 2 or 3 answer 'YES' or 'NO' if the condition presented in the corresponding query holds or not.

Example

standard input	standard output
5 abc a ab a c 3 2 1 2 a 1 2 b 3 2 3 b	YES YES
5 abc ab a xy wxz 6 2 1 2 abcdario 2 2 2 icpc 3 1 3 ab 3 1 3 x 1 2 xp 3 1 3 x	YES NO YES NO YES

Problem L. Competition

There are n contestants participating in a round robin competition. Each pair of contestants face each other only once. In each match there is only one winner, (**there are no ties**). Given a subset of contestants X and the winners of the matches between these contestants, it is said that the best player of X cannot be determined if there is no contestant with more wins than the rest. It is desired to find, given the results of all the matches made in the competition, a set of 3 contestants where the best contestant cannot be determined.

Input

The first line contains an integer $3 \leq n \leq 2000$. The amount of contestants in the competition. The next n lines represent the results of player $i{-}th$ against the other contestants. If the j-th character in the i-th line is 1, it means that the contestant i defeated the contestant j; if it is a 0 it means they lost. For each pair of distinct contestants i and j, it is guaranteed that either i defeats j or j defeats i (but not both).

Output

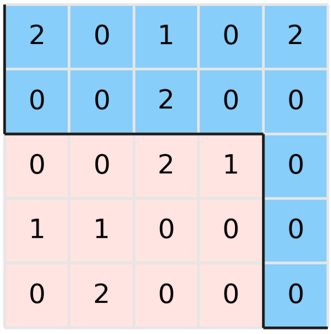
Print three integers a, b, c ($1 \leq a, b, c \leq n$), the indices of the players that satisfy that a better player cannot be selected among them. If there are multiple answers, print any. If there is no solution to the problem, print -1.

Example

standard input	standard output
4 0110 0000 0101 1100	1 3 4
4 0111 0000 0101 0100	-1

Problem M. Even Split

You are given a matrix A where each cell has a value of 0, 1 or 2. We want to divide the matrix into two parts, from the vertex at position $(0, 0)$ to the vertex at (n, m) by a sequence of edges, in such a way that edge a precedes edge b, if b is to the right or below of a and they share the vertex on the right or below of edge a. Is it possible to do the division in such a way that on both sides the sum of the values of the cells are the same?



Example testcase 3

Input

First line contains 2 integers n y m $(1 \leq n, m \leq 1000)$, the amount of rows and columns of the matrix respectively. Then n lines, each one with m integers $A_{i,j}$ $(0 \leq A_{i,j} \leq 2)$, indicating the value of the cell j-*th* on the row i-*th*.

Output

Print 'YES' if it is possible to find a path that satisfies the conditions described above; otherwise print 'NO'. If the answer is 'YES', print a second line with a string of length $n + m$ consisting of characters 'R' (right) or 'D' (down) that describes a possible path. If there are multiple paths satisfying the above conditions, print any of them.

Example

standard input	standard output
2 2 2 2 1 1	YES RDDR
3 3 0 1 1 2 0 0 2 0 0	NO
5 5 2 0 1 0 2 0 0 2 0 0 0 0 2 1 0 1 1 0 0 0 0 2 0 0 0	YES DRRDRRDDDR
2 3 0 0 0 0 0 0	YES DDRRR

Problem N. Arithmetic Mean

Given four integer values, determine if one of them is equal to the mean of other two values. Remember that the mean of two values a and b is $\frac{a+b}{2}$.

Input

The first line of the input contains the four integers a, b, c and d $(1 \le a, b, c, d \le 100)$

Output

If there exists one value that is equal to the mean of other two values, then print "YES" without quotes, otherwise print "NO" without quotes.

Example

standard input	standard output
3 10 1 2	YES
100 100 100 100	YES
10 21 21 15	NO

Problem O. Polygon

You are given a **convex** polygon with n vertices (no three of them lie on the same line) and a triangulation for that polygon. You should answer q queries of the type x_1, y_1, x_2, y_2. For each query you should output the minimum number of diagonals you need to cross when walking from point (x_1, y_1) to point (x_2, y_2) without stepping out of the polygon.

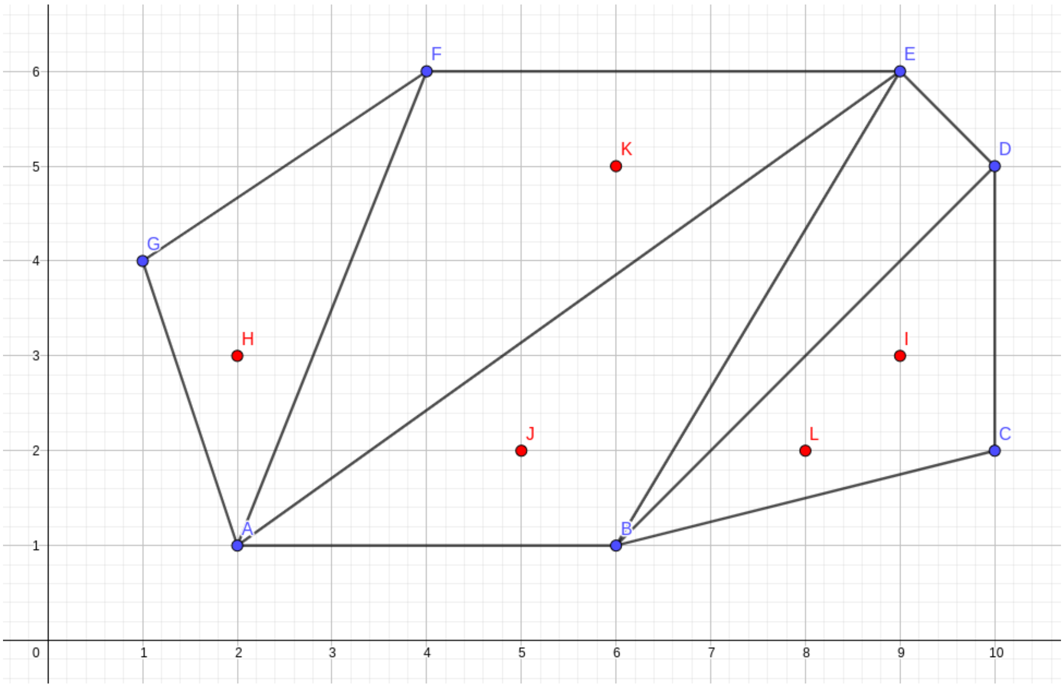


Image is related to sample input

Example:

1. If we are at point H inside $\triangle AFG$ and want to get to point I in $\triangle BCD$, best route implies crossing all 4 given diagonals ($\overline{AF}$, $\overline{AE}$, $\overline{BE}$, $\overline{BD}$).
2. If we are at point J inside $\triangle ABE$ and want to get to point K in $\triangle AEF$, since points are in adjacent triangles, it is sufficient to cross the diagonal $\overline{AE}$.
3. If we are at point L inside $\triangle BCD$ and want to get to point I in $\triangle BCD$, since both points are inside the same triangle, the amount of diagonals to cross is 0.

Input

First line of input contains an integer n ($3 \le n \le 10^5$) describing the number of vertices of the polygon. N lines follow, the i-th of them with two integers x_i, y_i ($-10^9 \le x_i, y_i \le 10^9$), the coordinates of the i-th vertex of the polygon, given in counter-clockwise order. Next $n-3$ lines describes the triangulation, giving the list of diagonals that make up the triangulation. Each of those lines contains two integers a_i, b_i ($1 \le a_i, b_i \le n$), indicating there is a diagonal from the a_i-th vertex to the b_i-th vertex, according to the order in which vertices were given in the input. It is guaranteed that the given list of diagonals make up a valid triangulation for the polygon. Next line gives you the integer q ($1 \le q \le 10^5$), the number of queries to process. And the final q lines contain four integers x_1, y_1, x_2, y_2 describing the query. It is guaranteed that both points are inside the polygon and do not lie on a given diagonal nor on a side of the polygon.

Output

For each query output a line with answer for that query.

Example

standard input	standard output
7 2 1 6 1 10 2 10 5 9 6 4 6 1 4 1 6 1 5 2 4 2 5 3 2 3 9 3 5 2 6 5 8 2 9 3	4 1 0
4 100 100 200 100 200 200 100 200 1 3 3 150 140 150 160 190 140 190 160 110 140 110 160	1 0 0