Problem A. Chords

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

Consider a circle. There are $2N$ different points marked on it and numbered with integer numbers in range from 1 to N so that each number from the interval has two points corresponding to it.

Points with the same numbers are connected by segments. Thus we've got N chords. Futhermore, the chords are numbered as well: the chord number "i" connects the two different points with number "i". Apparently some of the chords may intersect. Now it would be nice to find out for each single chord the number of chords it intersects.

Input

In the first line of the input file there is a single integer number N ($1 \le N \le 10^5$). In the next line there are $2N$ integers in range from 1 to N — the numbers assigned to the points in the order of traversal. Each number is written exactly twice. All the numbers in the line are separated with spaces.

Output

The output file is supposed to contain exactly N lines: on the i-th line write the number of chords that i-th chord intersects.

Example

standard input	standard output
5 1 2 3 1 4 2 5 5 3 4	2 3 3 2 0

Problem B. Cyclic suffixes

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

Consider a string $S = s_1 s_2 s_3 \dots s_{n-1} s_n$ over an alphabet Σ. A *cyclic expansion* of order m over a string S is the string $s_1 s_2 s_3 \dots s_{n-1} s_n s_1 s_2 \dots$ of m characters; that is, we concatenate instances of S until the resulting string has sufficient length, and then take the first m characters as the result.

Let *cyclic string* $\tilde{S}$ be the infinite cyclic expansion over a string S.

Take a look at the suffixes of a cyclic string $\tilde{S}$. Obviously, there are no more than $|S|$ different ones among them: the $(n+1)$-th one is equal to the first (the cyclic string itself), the $(n+2)$-th is equal to the second, and so on. However, there may be even less different suffixes. For example, if $S =$ `abab`, the first suffixes of $\tilde{S}$ are:

$$\begin{aligned} \tilde{S}_1 &= \texttt{abababab}\dots \\ \tilde{S}_2 &= \texttt{bababababa}\dots \\ \tilde{S}_3 &= \texttt{abababab}\dots \\ \tilde{S}_4 &= \texttt{bababababa}\dots \end{aligned}$$

etc. Here, only two different suffixes can be found, while $|S| = 4$.

Let us order the first $|S|$ suffixes of $\tilde{S}$ lexicographically. If two suffixes are equal, the one with the lower index comes first. We are now interested in the following problem: what is the position of our original string $\tilde{S}$ after sorting?

The following example shows the ordering for $S =$ `cabcab`.

$$\begin{aligned} (1) \quad \tilde{S}_2 &= \texttt{abcabcabca}\dots \\ (2) \quad \tilde{S}_5 &= \texttt{abcabcabca}\dots \\ (3) \quad \tilde{S}_3 &= \texttt{bcabcabcab}\dots \\ (4) \quad \tilde{S}_6 &= \texttt{bcabcabcab}\dots \\ (5) \quad \tilde{S}_1 &= \texttt{cabcabcabc}\dots \\ (6) \quad \tilde{S}_4 &= \texttt{cabcabcabc}\dots \end{aligned}$$

Here, the position of $\tilde{S} = \tilde{S}_1$ is 5.

Given a string S of length n, find the number of $\tilde{S}$ in the specified ordering.

Input

On the first line of the input file, a string S ($1 \le |S| \le 1\,000\,000$) is given. The input consists of lowercase Latin letters only.

Output

Output the only number on a line by itself — the number of $\tilde{S}$ in the specified ordering of its first $|S|$ suffixes.

Examples

standard input	standard output
abracadabra	3
cabcab	5

Problem C. False RSA

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

Roman, Serge and Andrew has decided to upgrade the famous RSA encryption algorithm. They think that the limitation that modulo n used in RSA must be a product of two distinct primes is redundant. Instead they plan to use n which is the product of k-th powers of two distinct primes: $n = p^k q^k$.

However, Nick pointed out that besides all other mathematical problems, the scheme may be easier to crack. That is — it is difficult to factorize n which is a product of two distinct primes, because it has exactly one non-trivial factorization. On the other hand, in case $n = p^k q^k$ there can be more different non-trivial factorizations. For example, $100 = 2^2 5^2$ has eight non-trivial factorizations: $100 = 2 \cdot 50$, $100 = 2 \cdot 2 \cdot 25$, $100 = 2 \cdot 2 \cdot 5 \cdot 5$, $100 = 2 \cdot 5 \cdot 10$, $100 = 4 \cdot 25$, $100 = 4 \cdot 5 \cdot 5$, $100 = 5 \cdot 20$ and $100 = 10 \cdot 10$.

Now Roman, Serge and Andrew wonder — given $n = p^k q^k$, how many different non-trivial factorizations of n are there?

Input

The input file contains one integer number n ($6 \le n \le 10^{18}$, it is guaranteed that $n = p^k q^k$ for different primes p and q and integer $k > 0$).

Output

Output one integer number — the number of different non-trivial factorizations of n.

Example

standard input	standard output
6	1
100	8

Problem D. A Coloring Game

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

Two players play a graph coloring game. They make moves in turn, first player moves first. Initially they take some undirected graph. At each move, a player can color an uncolored vertex with either white or black color (each player can use any color, possibly different at different turns). It's not allowed to color two adjacent vertices with the same color. A player that can't move loses.

After playing this game for some time, they decided to study it. For a start, they've decided to study very simple kind of graph — a chain. A chain consists of N vertices, v_1, v_2, ..., v_N, and $N - 1$ edges, connecting v_1 with v_2, v_2 with v_3, ..., v_{N-1} with v_N.

Given a position in this game, and assuming both players play optimally, who will win?

Input

The first line of the input file contains the integer N, $1 \le N \le 100\,000$.

The second line of the input file describes the current position. It contains N digits without spaces. i^{th} digit describes the color of vertex v_i: 0 — uncolored, 1 — black, 2 — white. No two vertices of the same color are adjacent.

Output

On the only line of the output file, print "`FIRST`" (without quotes) if the player moving first in that position wins the game, and "`SECOND`" (without quotes) otherwise.

Example

standard input	standard output
5 00100	SECOND
4 1020	FIRST

Problem E. Hippopotamus

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

You think that your roof looks unpretty. So you opt for a new one, consisting of n consecutive long narrow boards. You have two types of boards: wooden ones and iron ones, giving you an amazing total of 2^n possible roofs.

But the safety should not be left aside. Having considered the weight and the cruising speed of a falling hippopotamus, you decide to have at least k iron boards among every m consecutive boards.

How many possibilities do you have?

Input

The input file contains three integers, n, m and k, separated by spaces and/or line breaks. $1 \leq n \leq 60$, $1 \leq m \leq 15$, $0 \leq k \leq m \leq n$.

Output

Output the number of possibilities.

Example

standard input	standard output
10 2 1	144
5 5 2	26
3 2 2	1

Problem F. Perspective

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

Breaking news! A Russian billionaire has bought a yet undisclosed NBA team. He's planning to invest huge effort and money into making that team the best. And in fact he's been very specific about the expected result: the first place.

Being his advisor, you need to determine whether it's possible for your team to finish first in its division or not.

More formally, the NBA regular season is organized as follows: all teams play some games, in each game one team wins and one team loses. Teams are grouped into divisions, some games are between the teams in the same division, and some are between the teams in different divisions.

Given the current score and the total number of remaining games for each team of your division, and the number of remaining games between each pair of teams in your division, determine if it's possible for your team to score at least as much wins as any other team in your division.

Input

The first line of the input file contains N ($2 \le N \le 20$) — the number of teams in your division. They are numbered from 1 to N, your team has number 1.

The second line of the input file contains N integers w_1, w_2, ..., w_N, where w_i is the total number of games that i^{th} team has won to the moment.

The third line of the input file contains N integers r_1, r_2, ..., r_N, where r_i is the total number of remaining games for the i^{th} team (including the games inside the division).

The next N lines contain N integers each. The j^{th} integer in the i^{th} line of those contains a_{ij} — the number of games remaining between teams i and j. It is always true that $a_{ij} = a_{ji}$ and $a_{ii} = 0$, for all i $\sum_j a_{ij} \le r_i$.

All the numbers in the input file are non-negative and don't exceed 10 000.

Output

On the only line of output, print "`YES`" (without quotes) if it's possible for the team 1 to score at least as much wins as any other team of its division, and "`NO`" (without quotes) otherwise.

Example

standard input	standard output
3 1 2 2 1 1 1 0 0 0 0 0 0 0 0 0	YES
3 1 2 2 1 1 1 0 0 0 0 0 1 0 1 0	NO

Problem G. Circular Railway

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

There are L stations along a circular railway, numbered 1 through L. Trains travel in both directions, and take 1 minute to get from a station to the neighbouring one (i.e., between 1st and 2nd, between 2nd and 3rd, ..., between $(L-1)$-th and L-th and between L-th and 1-st).

There are n employee's houses along the railway, and n offices, each house or office located near a railway station. You are to establish a one-to-one correspondence between houses and offices in such a way that total travel time (sum of travel times of each employee) is minimized.

Input

The first line of the input file contains two integer numbers, n and L ($1 \le n \le 50000$, $2 \le L \le 10^9$). The second line contains n locations of the employee's houses, and the third line contains n locations of the offices. Each location is an integer number between 1 and L. Some houses or offices or both can be located at the same railway station.

Output

Output the minimal total travel time followed by the description of the one-to-one correspondence. The description should be represented by n numbers (one for each employee, ordered as in the input), denoting the 1-based index of the office assigned to the corresponding employee.

Example

standard input	standard output
3 15 1 2 10 11 12 13	9 2 3 1
4 12 2 5 8 11 6 9 12 3	4 4 1 2 3

Problem H. SETI

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

Amateur astronomers Tom and Bob try to find radio broadcasts of extraterrestrial civilizations in the air. Recently they received some strange signal and represented it as a word consisting of small letters of the English alphabet. Now they wish to decode the signal. But they do not know what to start with.

They think that the extraterrestrial message consists of words, but they cannot identify them. Tom and Bob call a subword of the message a *potential word* if it has at least two non-overlapping occurrences in the message.

For example, if the message is "`abacabacaba`", "`abac`" is a potential word, but "`acaba`" is not because two of its occurrences overlap.

Given a message m help Tom and Bob to find the number of potential words in it.

Input

Input file contains one string that consists of small letters of the English alphabet. The length of the message doesn't exceed 10 000.

Output

Output one integer number — the number of potential words in a message.

Example

standard input	standard output
abacabacaba	15

Problem I. Small Graph

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

Consider a permutation $p_1, p_2, \ldots, p_n$ of integers between 1 and n. A directed graph G with at least n vertices is called an *inversion graph* of that permutation when for any i, j such that $1 \le i, j \le n$, $i \ne j$ the j-th vertex of G is reachable via some path from the i-th vertex of G *if and only if* $i < j$ and $p_i > p_j$ (such pair is called an *inversion* of the permutation).

Naturally, there exist many inversion graphs for each permutation. You need to find a relatively small one: the graph must contain at most $30n$ vertices and at most $30n$ edges.

Input

The first line of the input file contains one integer n, $1 \le n \le 1000$, denoting the size of the permutation. The second line of the input file contains the permutation itself.

Output

In the first line of the output file, print two integers v and a — the number of vertices and arcs of the graph, respectively. The next a lines should contain the arcs. Each arc should be described by two vertex numbers (between 1 and v) — the source and destination of the arc.

The number v should be at least n and at most $30n$, a should be at least 0 and at most $30n$.

Examples

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

Problem J. 2-3 Trees

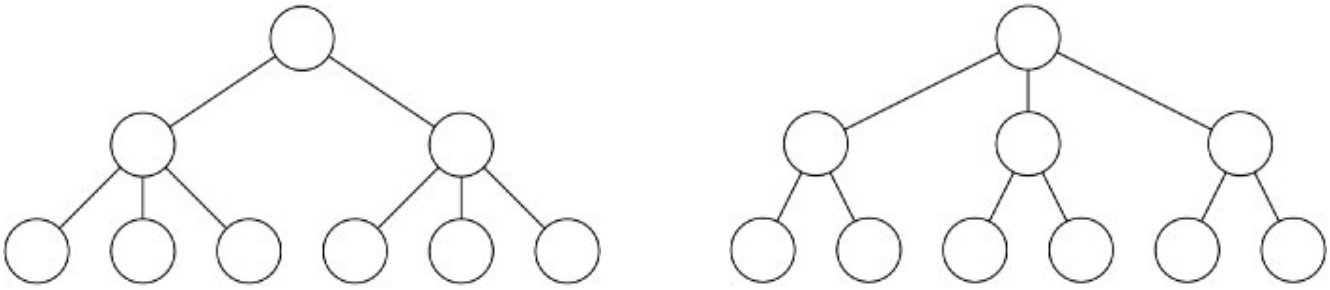
Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	256 Mebibytes

2-3 tree is an elegant data structure invented by John Hopcroft. It is designed to implement the same functionality as the binary search tree. 2-3 tree is an ordered rooted tree with the following properties:

- the root and each internal vertex have either 2 or 3 children;
- the distance from the root to any leaf of the tree is the same.

The only exception is the tree that contains exactly one vertex — in this case the root of the tree is the only vertex, and it is simultaneously a leaf, i.e. has no children. The main idea of the described properties is that the tree with l leaves has the height $O(\log l)$.

Given the number of leaves l there can be several valid 2-3 trees that have l leaves. For example, the picture below shows the two possible 2-3 trees with exactly 6 leaves.



Given l find the number of different 2-3 trees that have l leaves. Since this number can be quite large, output it modulo r.

Input

Input file contains two integer numbers: l and r ($1 \le l \le 5\,000$, $1 \le r \le 10^9$).

Output

Output one number — the number of different 2-3 trees with exactly l leaves modulo r.

Example

standard input	standard output
6 1000000000	2
7 1000000000	3

Problem K. Circular Shift

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

Vasya was at a meeting during his working day at Yandex. Suddenly he thought about a string s consisting of lowercase English letters.

Then he decided that a string $t = t_1 t_2 \dots t_m$ $(m > 0)$ is called a *good* string with respect to s if t is a substring of s and the left circular shift $t' = t_2 \dots t_m t_1$ of string t is also a substring of s.

Vasya was going to calculate the number of different good strings t with respect to the given string s... but suddenly a colleague asked him a question, so he had to return back to reality. Find that number for Vasya while he is busy with the meeting.

Input

The only input line contains a string s consisting of n $(1 \le n \le 300\,000)$ lowercase English letters.

Output

Output a single integer: the number of different good strings t with respect to the given string s.

Examples

standard input	standard output
abaac	7
aaa	3

Note

In the first sample case, the good strings are exactly the following strings: `a`, `b`, `c`, `aa`, `ab`, `ba`, `aba`.

In the second sample case, the good strings are exactly the following strings: `a`, `aa`, `aaa`.