Problem A. Almost Bobo Number

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

A positive integer is called a *bobo number* if its decimal representation can be obtained by concatenating two copies of the same integer. For example, 12341234 and 3232 are *bobo numbers*, while 1234321 and 1322 are not.

A positive integer is called an *almost bobo number* if, after merging all the consecutive equal digits, the resulting number is a *bobo number*. For example, 1112223112233 becomes 123123 after merging all the consecutive equal digits, and thus is an *almost bobo number*.

Bobo has a very large number n, and he would like to know the largest *almomst bobo number* less than n.

Input

The input contains zero or more test cases, and is terminated by end-of-file. For each test case:

The first line contains an integer n without leading zeros ($1 \le n \le 10^{5\,000\,000}$).

It is guaranteed that the total length of the decimal representations of all n in the input does not exceed $5\,000\,000$.

Output

For each test case, output an integer without leading zeros denoting the largest *almomst bobo number* strictly less than n. If there is no such integer, output -1 instead.

Example

standard input	standard output
12345 67890 11111 1000 26782641	12212 67767 11010 -1 26777267

Problem B. Connected Spanning Subgraph

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

Bobo has a connected undirected graph G with n vertices and m edges where vertices are conveniently labeled with $1, 2, \ldots, n$.

Bobo chooses a non-empty subset of edges such that the graph with the chosen edges is still connected. He would like to know the number of such subsets modulo 2.

Note that a graph is connected if, for any two vertices a and b, there exists a path which connects a and b.

Input

The input contains zero or more test cases, and is terminated by end-of-file. For each test case:

The first line contains two integers n and m ($2 \le n \le 2 \cdot 10^5$, $1 \le m \le 2 \cdot 10^5$).

The i-th of the following m lines contains two integers a_i and b_i which denote an edge between vertices a_i and b_i.

It is guaranteed that the sum of all m does not exceed $2 \cdot 10^5$, and all the given graphs are connected.

Output

For each test case, output an integer which denotes the remainder modulo 2.

Example

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

Problem C. Power of Power Partition Function

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

Let m be a fixed integer such that $m \geq 2$. For a positive integer n, let $b_m(n)$ denote the number of ways of writing n as a sum of powers of m using non-negative exponents with repetitions allowed and the order of the summands not being taken into account. We also set $b_m(0) = 1$ (there is one empty sum).

For example, the first 10 terms of $\{b_2(n)\}$ are $\{1, 1, 2, 2, 4, 4, 6, 6, 10, 10\}$, and the first 10 terms of $\{b_3(n)\}$ are $\{1, 1, 1, 2, 2, 2, 3, 3, 3, 5\}$.

Let $c_m^k(n)$ be the k-th convolution power of $b_m(n)$, which is defined as follows:

$$c_m^k(n) = \begin{cases} b_m(n), & k = 1 \\ \sum_{i=0}^{n} b_m(i) \cdot c_m^{k-1}(n-i), & k \geq 2 \end{cases}$$

Given n, m and k, Bobo would like to find the value of

$$f(n) = \left(\sum_{i=0}^{n} c_m^k(i) \right) \bmod (10^9 + 7).$$

Input

The first line contains three integers n, m and k ($0 \leq n \leq 10^{18}$, $2 \leq m \leq 10^{18}$, $1 \leq k \leq 10$).

Output

Output an integer denoting the value of $f(n)$.

Examples

standard input	standard output
0 2 1	1
10 2 3	2700
100 2 10	490796617

Problem D. Line Counting

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

Bobo has a set P of $\frac{n(n+1)}{2}$ points: $\{(x, y) : 1 \le x \le y \le n,\ x, y \in \mathbb{Z}\}$. He would like to know the number of distinct lines passing through at least two points in P, taken modulo $(10^9 + 7)$.

Input

The input contains zero or more test cases, and is terminated by end-of-file.

Each test case is a single line containing an integer n $(2 \le n \le 2 \cdot 10^9)$.

It is guaranteed that the number of test cases does not exceed 10^5, and the sum of all n does not exceed $2 \cdot 10^9$.

Output

For each test case, output an integer which denotes the number of distinct lines.

Example

standard input	standard output
2 3 5	3 9 51

Problem E. Maximum Flow

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

Bobo has an undirected graph with $(2n + 2)$ vertices conveniently labeled with the following pairs of integers: $(0, 0), (0, 1), \ldots, (0, n), (1, 0), (1, 1), \ldots, (1, n)$. The graph has three classes of edges.

- The edges of the first class connect vertices $(0, i-1)$ and $(0, i)$ with capacity a_i for $i \in \{1, 2, \ldots, n\}$.
- The edges of the second class connect vertices $(1, i-1)$ and $(1, i)$ with capacity b_i for $i \in \{1, 2, \ldots, n\}$.
- The edges of the third class connect vertices $(0, \lfloor \frac{i-1}{2} \rfloor)$ and $(1, \lfloor \frac{i}{2} \rfloor)$ with capacity c_i for $i \in \{1, 2, \ldots, 2n + 1\}$.

Bobo would like to find the maximum flow from vertex $(0, 0)$ to vertex $(1, n)$.

Input

The input contains zero or more test cases, and is terminated by end-of-file. For each test case:

The first line contains an integer n ($1 \le n \le 5 \cdot 10^5$).

The second line contains n integers $a_1, a_2, \ldots, a_n$.

The third line contains n integers $b_1, b_2, \ldots, b_n$.

The fourth line contains $(2n + 1)$ integers $c_1, c_2, \ldots, c_{2n+1}$.

The constraints are: $1 \le a_i, b_i, c_i \le 10^9$.

It is guaranteed that the number of test cases does not exceed 10^5, and the sum of all n does not exceed $5 \cdot 10^5$.

Output

For each test case, output an integer which denotes the maximum flow.

Example

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

Problem F. Rectangles Inside Rectangle

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

Bobo has a large rectangle with lower left and upper right corners at $(0, 0)$ and $(w, 10^6)$. He also has n small axis-parallel rectangles inside the large rectangle. The weight of the i-th rectangle is v_i. For each rectangle, either its left border or its right border (but not both) coincides with the left or right side of the large rectangle.

Bobo would like to choose a subset of small rectangles in such a manner that the rectangles may touch each other, but they do not overlap (that is, there are no points that belong to the interior of more than one rectangle). Among all the possibilities, he wants the one with the maximum possible sum of weights.

Input

The input contains zero or more test cases, and is terminated by end-of-file. For each test case:

The first line contains two integers n and w ($1 \le n \le 2000$, $2 \le w \le 10^6$) denoting the number of small rectangles and the width of the large rectangle.

The i-th of the following n lines contains five integers $type_i$, l_i, a_i, b_i and v_i ($type_i \in \{0, 1\}$, $0 \le a_i < b_i \le 10^6$, $1 \le l_i < w$, $0 \le v_i \le 10^6$) where v_i is the weight of the i-th rectangle. Here, $type_i = 0$ means the lower left and upper right corner of the i-th rectangle are $(0, a_i)$ and (l_i, b_i), while $type_i = 1$ means the lower left and upper right corner of the i-th rectangle are $(w - l_i, a_i)$ and (w, b_i).

It is guaranteed that for all $1 \le i < j \le n$, $a_i \ne a_j$, $a_i \ne b_j$, $b_i \ne a_j$ and $b_i \ne b_j$. Additionally, the sum of all n does not exceed 2000.

Output

For each test case, output an integer which denotes the maximum sum of weights.

Example

standard input	standard output
3 10 0 3 1 6 12 0 3 3 4 100 1 9 2 5 11 3 10 0 3 1 6 12 0 1 3 4 5 1 9 2 5 11 6 5 1 1 17 32 4 0 3 1 18 7 1 3 4 8 12 1 2 15 20 14 1 1 30 33 16 1 4 2 16 13	100 16 42

Note

For the third test, Bobo can choose the 3-rd, 4-th and 5-th rectangles.

Problem G. Cute Panda

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

There are n pandas numbered from 1 to n, i-th of them has a_i donuts. There are also n bins numbered from 1 to n, i-th of them can hold b_i donuts. For any i from 1 to n, i-th panda can distribute his donuts to i-th and $(i \bmod n + 1)$-th bin.

Can you find a way to maximize the number of distributed donuts?

Input

The input contains zero or more test cases, and is terminated by end-of-file. For each test case:

The first line contains an integer n ($3 \le n \le 10^6$).

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($0 \le a_i \le 10^9$).

The third line contains n integers $b_1, b_2, \ldots, b_n$ ($0 \le b_i \le 10^9$).

It is guaranteed that the sum of all n does not exceed 10^6.

Output

For each test case, output an integer which denotes the maximum number of distributed donuts.

Example

standard input	standard output
5 8 4 8 3 10 1 0 4 5 1 5 9 4 10 0 4 3 5 2 2 1	11 13

Problem H. Order-Preserving Partition

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

Bobo has two permutations: $P = \{p_1, p_2, \ldots, p_n\}$ and $Q = \{q_1, q_2, q_3, q_4\}$. He would like to partition P into four non-empty and contiguous parts in such a manner that:

- The numbers in each part can be rearranged to form an *interval* of values: an increasing sequence where each element is greater than the previous by exactly one.
- For all $1 \le i < j \le 4$, $(s_i - s_j) \cdot (q_i - q_j) > 0$ where s_i is the minimum value in the i-th part.

Bobo wants to know the number of such partitions. As the number may be very large, you just need to print the answer modulo $(10^9 + 7)$.

Input

The input contains zero or more test cases, and is terminated by end-of-file. For each test case:

The first line contains an integer n, the length of the first permutation ($4 \le n \le 10^6$).

The second line contains n integers $p_1, p_2, \ldots, p_n$.

The third line contains four integers q_1, q_2, q_3, q_4.

It is guaranteed that the sum of all n does not exceed 10^6.

Output

For each test case, output an integer denoting the answer.

Example

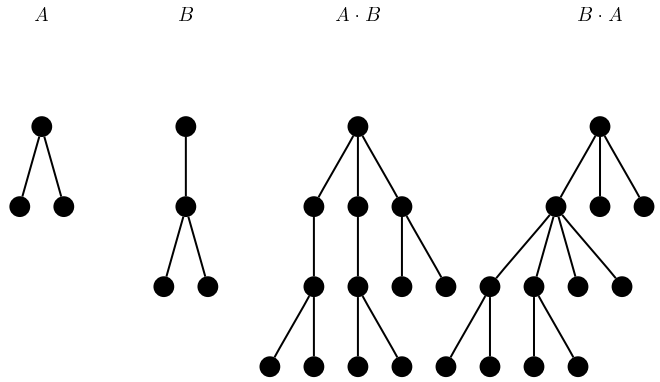
standard input	standard output
10 2 1 4 3 10 9 8 7 5 6 2 4 1 3 10 1 2 3 4 5 6 7 8 9 10 1 2 3 4	0 84

Problem I. Prime Tree

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

Bobo proposes a multiplication operation on rooted trees.

Let A and B be two arbitrary rooted trees. Then $T = A \cdot B$ is built by making a copy of B for each vertex $x \in A$ and merging the root of this copy with x (see the following figure for more details). We then call A and B *factors* of T.



Apparently, we have $T \cdot \mathbf{1} = \mathbf{1} \cdot T = T$, where $\mathbf{1}$ is the rooted tree with only one vertex. So, $\mathbf{1}$ is a factor of every rooted tree, and every rooted tree is a factor of itself. And if a rooted tree T only has T and $\mathbf{1}$ as his factors, we call T a *prime* tree.

Bobo has a rooted tree T with n nodes which are conveniently labeled with $1, 2, \ldots, n$. He wants to factor T into multiplication of as many prime trees as possible (that is, find an equation $T = T_1 \cdot T_2 \cdots T_m$ where T_i $(1 \le i \le m)$ are prime trees and m is maximum).

Note that $\mathbf{1}$ is not a prime tree.

Input

The input contains zero or more test cases, and is terminated by end-of-file. For each test case:

The first line contains an integer n, the number of nodes $(2 \le n \le 10^6)$.

The second line contains $(n - 1)$ integers $p_2, p_3, \ldots, p_n$, where p_i is the parent of the i-th node $(1 \le p_i \le i - 1)$.

It is guaranteed that the sum of all n does not exceed 10^6.

Output

For each test case, output an integer denoting the maximum number of prime factors.

Example

standard input	standard output
12 1 1 1 1 2 2 4 5 5 6 10 3 1 1 6 1 1 1 2 3 13 1 1 1 2 2 3 3 4 5 6 7 8	3 1 2 1

Problem J. Hamiltonian k-vertex-connected Graph

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

A graph (other than a complete graph) has connectivity k if k is the size of the smallest subset of vertices such that the graph becomes disconnected if you delete them.

A connected undirected graph G is called Hamiltonian if it has a Hamiltonian cycle: a cycle that visits each vertex exactly once (except for the vertex that is both the start and the end, which is visited twice).

Bobo would like to construct a Hamiltonian graph with n vertices which has connectivity k. Also, the number of edges in the graph should be minimum possible.

Input

The first line contains two integers n and k where n is the number of vertices in the graph ($3 \le n \le 100$, $1 \le k \le n - 2$).

Output

If there is no such graph, output -1 on a single line. Otherwise, output an integer m denoting the minimum number of edges. Then in each of the next m lines, output two integers x and y ($1 \le x, y \le n$, $x \ne y$) denoting an edge in the graph. In the following line, output a permutation of integers $1, 2, \ldots, n$ denoting a Hamiltonian cycle in the graph.

Example

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