The 3rd Universal Cup

Stage 2: Zielona Góra

June 15-16, 2024

This problem set should contain 13 problems on 19 numbered pages.

Based on

POTYCZKI ALGORYTMICZNE

Potyczki Algorytmiczne / Algorithmic Engagements

Problem A. Interesting Paths

Input file:	`standard input`
Output file:	`standard output`
Time limit:	4 seconds
Memory limit:	1024 megabytes

We are given a directed graph with n vertices and m edges. The vertices of the graph are numbered from 1 to n, and each edge leads from a vertex with a smaller number to a vertex with a larger number.

We call a sequence of paths interesting if:

- each path starts at vertex 1 and ends at vertex n,
- each path contains at least one edge that was not present in any of the previous paths.

What is the length of the longest interesting sequence of paths?

Input

The first line of the input contains two integers n and m ($2 \leq n \leq 10^6$, $0 \leq m \leq 10^6$), denoting the number of vertices and edges of the considered graph.

Each of the next m lines contains two integers a and b ($1 \leq a < b \leq n$) describing a directed edge leading from vertex a to vertex b. Each pair (a, b) appears in the input at most once.

Output

The output should contain a single integer - the maximum length of the interesting sequence of paths.

Examples

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

Note

For the first example, an interesting sequence of paths of length 4 may look as follows:

- $1 \to 3 \to 5$ (the edge $1 \to 3$ was used for the first time),
- $1 \to 2 \to 3 \to 5$ (the edge $1 \to 2$ was used for the first time),
- $1 \to 3 \to 4 \to 5$ (the edge $3 \to 4$ was used for the first time),
- $1 \to 2 \to 4 \to 5$ (the edge $2 \to 4$ was used for the first time).

In the second example, there is no path from vertex 1 to vertex 5.

Problem B. Roars III

Input file:	`standard input`
Output file:	`standard output`
Time limit:	6 seconds
Memory limit:	1024 megabytes

In the middle of the Hundred Bytes Wood grows an unusual tree inhabited by roaring snails. This tree consists of n vertices numbered from 1 to n connected in a connected graph with $n - 1$ edges. At the beginning, there is at most one male roaring snail in each vertex of the tree.

In a moment, a female will appear in a certain vertex of the tree and will start roaring. Each time the female roars, one male will move along an edge to an adjacent vertex of the tree closer to the female. However, the male cannot move if there is already another male in the target vertex, or if it is already in the same vertex as the female. The female stops roaring if after the next roar no male could make a move.

You are given a description of the tree inhabited by roaring snails as input. For each vertex, you also receive information about whether a male snail is present in it. For each vertex, determine the maximum number of times the female could roar if she were to be in that vertex. We assume that the males move in a way that maximizes the number of roars.

Input

The first line of the input contains a single integer n ($1 \le n \le 2 \cdot 10^5$) representing the number of vertices in the tree under consideration.

The second line of the input contains a word consisting of n characters `0` and `1`. If the i-th character of this word is `1`, then there is a male snail in the i-th vertex of the tree. If the i-th character of this word is `0`, then there is no male snail in the i-th vertex of the tree.

The next $n - 1$ lines contain two integers a_i and b_i ($1 \le a_i, b_i \le n; a_i \ne b_i$) indicating that the vertices a_i and b_i of the tree are connected by an edge.

Output

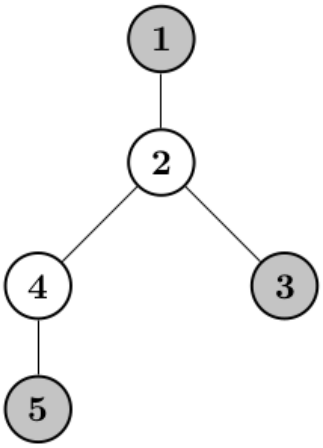
The output should contain n integers in a single line; the i-th integer should indicate the maximum number of roars the female can make if she were to be in the i-th vertex of the tree.

Example

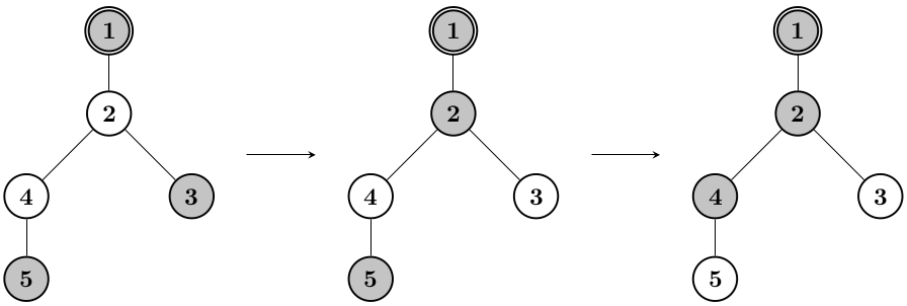
standard input	standard output
5 10101 1 2 2 3 2 4 4 5	2 2 2 3 3

Note

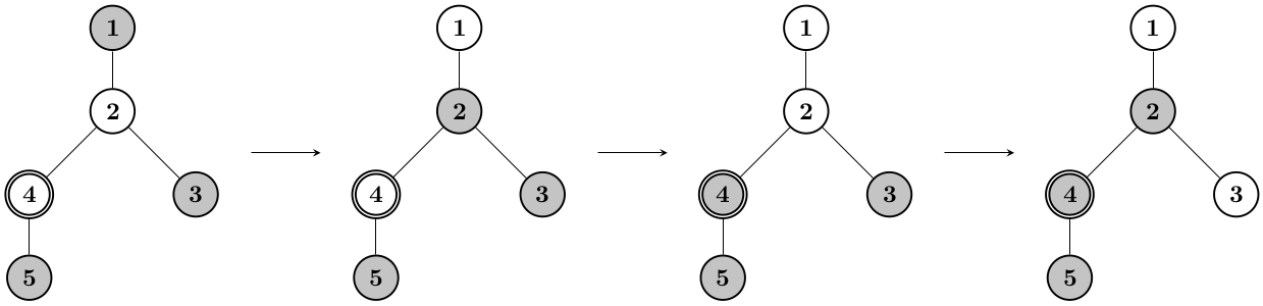
The tree in the sample test is shown below. The males are located in the vertices shaded in gray.



If the female is in the first vertex, she can roar at most twice, for example by first attracting the male from the third vertex to the second, and then the male from the fifth vertex to the fourth:



If the female is in the fourth vertex, as long as the male from the fifth vertex does not move, she can roar up to three times:



Problem C. Radars

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	1024 megabytes

We are given a board of size $n \times n$, where n is an odd number. For each cell of this board, the cost of building a radar in it is given. The radar covers a square area (with sides parallel to the sides of the board) with a side length of n and its center in the cell where the radar is located. Your task is to minimize the total cost of building a certain number of radars so that each field of the board is covered at least once. Additionally, you must solve multiple test cases.

Input

The first line of the standard input contains a single integer t ($t \geq 1$), indicating the number of test cases.

Then, in t blocks, the descriptions of subsequent test cases follow.

The first line of the test case description contains a single integer n ($1 \leq n \leq 499; n$ **is odd**), indicating the dimensions of the board.

In the next n lines, the board description is given.

In the i-th line, there are n integers $a_{i,1}$, $a_{i,2}$, ..., $a_{i,n}$ ($1 \leq a_{i,j} \leq 10^9$), where $a_{i,j}$ denotes the cost of building a radar in the cell located in the i-th row and j-th column of the board.

It is guaranteed that the sum of the values of n^2 in one file will not exceed 500 000.

Output

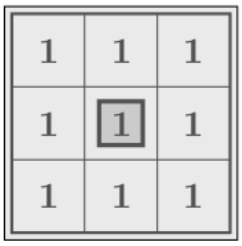
The output should contain t lines. The i-th line should contain a single integer - the minimum cost of building radars in the i-th test case.

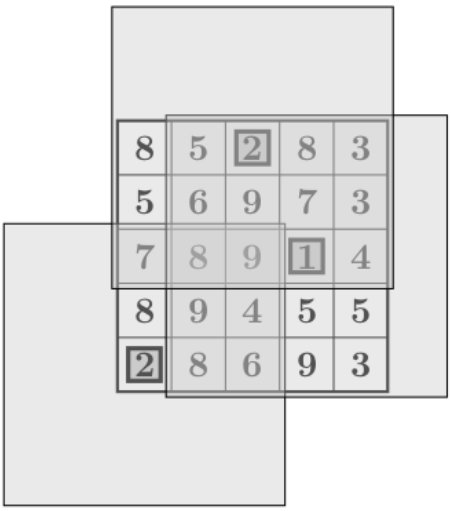
Example

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

Note

In the first sample test, it is worth building only one radar. In the second, it is worth building three. The optimal positions of the radars along with the areas covered by them are shown in the figures below:


1 1 1
1 1 1
1 1 1


8 5 2 8 3
5 6 9 7 3
7 8 9 1 4
8 9 4 5 5
2 8 6 9 3

Problem D. Xor Partitions

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	1024 megabytes

A sequence of integers $a_1, a_2, \ldots, a_n$ is given. A partition of the sequence $a_1, a_2, \ldots, a_n$ is defined as a set of its contiguous intervals, such that each element belongs to exactly one interval.

The xor of a contiguous interval of this sequence is defined as the bitwise `xor` of the numbers in this interval. The value of a partition of the sequence is defined as the product of xors of the intervals. Calculate the sum of the values of all possible partitions of the sequence $a_1, a_2, \ldots, a_n$. As this number can be very large, it is sufficient to provide its remainder when divided by $10^9 + 7$.

Input

The first line of the input contains a single integer n ($1 \le n \le 3 \cdot 10^5$) representing the length of the sequence.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($0 \le a_i \le 10^{18}$) representing the sequence.

Output

The output should contain a single integer, which is the remainder when the sum of the values of all possible partitions of the given sequence is divided by $10^9 + 7$.

Example

standard input	standard output
4 7 3 1 2	170

Note

Possible partitions of the sequence are:

- $[7, 3, 1, 2]$ – with a value of 7,
- $[7, 3], [1, 2]$ – with a value of $4 \cdot 3 = 12$,
- $[7], [3, 1, 2]$ – with a value of $7 \cdot 0 = 0$,
- $[7, 3, 1], [2]$ – with a value of $5 \cdot 2 = 10$,
- $[7, 3], [1], [2]$ – with a value of $4 \cdot 1 \cdot 2 = 8$,
- $[7], [3], [1, 2]$ – with a value of $7 \cdot 3 \cdot 3 = 63$,
- $[7], [3, 1], [2]$ – with a value of $7 \cdot 2 \cdot 2 = 28$,
- $[7], [3], [1], [2]$ – with a value of $7 \cdot 3 \cdot 1 \cdot 2 = 42$.

The sum of the values of the partitions is $7 + 12 + 0 + 10 + 8 + 63 + 28 + 42 = 170$.

Problem E. Pattern Search II

Input file:	`standard input`
Output file:	`standard output`
Time limit:	4 seconds
Memory limit:	1024 megabytes

Let's define an infinite sequence of *Fibonacci words* $S_0, S_1, S_2, S_3 \ldots$ as follows:

- $S_0 =$ `b`
- $S_1 =$ `a`
- $S_i = S_{i-1}S_{i-2}$ for $i \geq 2$

The first few words of this sequence look as follows:

- $S_0 =$ `b`
- $S_1 =$ `a`
- $S_2 =$ `ab`
- $S_3 =$ `aba`
- $S_4 =$ `abaab`
- $S_5 =$ `abaababa`
- $S_6 =$ `abaababaabaab`

It is easy to notice that each word (except for S_0) is a prefix of the next one. Therefore, we can also define an infinite Fibonacci word S, where the i-th character is the i-th character of these words in the infinite sequence of Fibonacci words (except for S_0) that have at least i characters.

You are given a word t consisting only of the characters '`a`' and '`b`'. Your task is to find the shortest possible word s that is a **substring** of S and contains t as a **subsequence**, and output its length.

Input

The standard input contains a single line with one word t, consisting only of the letters '`a`' and '`b`'. t will contain at least one and at most $150,000$ characters.

Output

The output should contain a single integer, indicating the minimum possible length of the described word s.

Example

standard input	standard output
aabbaab	8

Note

S does not contain a substring of length 7 or less that contains t as a subsequence. However, it does contain the substring `aababaab`, from which we can remove the fourth character, thus obtaining t. Therefore, the sought length of s is 8.

Problem F. Waterfall Matrix

Input file:	`standard input`
Output file:	`standard output`
Time limit:	8 seconds
Memory limit:	1024 megabytes

Radewoosh has recently discovered a special type of matrix. We say that a square matrix containing integers is a *Waterfall Matrix* if the number in each cell is greater than or equal to the numbers directly below and to the right, if they exist. In other words, a Waterfall Matrix M of size $n \times n$ is a matrix in which for all pairs (i, j) satisfying $1 \leq i \leq n$ and $1 \leq j \leq n-1$, we have $M_{i,j} \geq M_{i,j+1}$ and $M_{j,i} \geq M_{j+1,i}$.

Radewoosh would like to create a Waterfall Matrix of size $n \times n$. For n of its cells, he has come up with the values he would like to enter. Unfortunately, it may not be possible to enter exactly the value he wants in each chosen cell. Therefore, he decided to create a matrix that minimizes the sum of the absolute differences between what he wanted to enter and what he actually entered in the respective cells.

Formally, Radewoosh has a list of triplets of numbers (a_i, b_i, c_i) and wants to choose a Waterfall Matrix M to minimize the value of $\sum_i |M_{a_i,b_i} - c_i|$. Help him and output the minimum value that the mentioned sum can achieve if Radewoosh optimally chooses his matrix.

Input

The first line of the standard input contains a single integer n ($1 \leq n \leq 200\,000$), indicating the size of the matrix Radewoosh wants to draw **and** the number of fields he has chosen.

In each of the next n lines, there are three integers a_i, b_i, and c_i ($1 \leq a_i, b_i \leq n; 1 \leq c_i \leq 10^9$), as described in the task.

It is guaranteed that for $i \neq j$, $(a_i, b_i) \neq (a_j, b_j)$.

Output

The output should contain a single integer as described in the task.

Example

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

Note

One of the optimal matrices that Radewoosh can choose looks as follows:

9	7	**5**	5	5
6	6	5	5	3
5	**3**	**3**	3	**3**
3	2	2	**1**	1
2	1	1	1	1

For the above matrix, we can calculate the result as follows:

$|M_{1,3}-5|+|M_{3,2}-1|+|M_{3,3}-3|+|M_{4,4}-1|+|M_{3,5}-4| = |5-5|+|3-1|+|3-3|+|1-1|+|3-4| = 0+2+0+0+1 = 3$

Problem G. Puzzle II

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	1024 megabytes

Radewoosh received a new puzzle from his parents. This time it contains two cyclic sequences of balls. Each sequence contains n balls, each of which is either white or black. In total, both sequences contain exactly n white balls and n black balls. To describe the puzzle, we will use strings a and b - the consecutive characters of string a denote the colors of the consecutive balls in the first sequence, and the consecutive characters of string b denote the colors of the consecutive balls in the second sequence. The positions in both sequences are numbered from 1 to n.

The puzzle also involves an important number k. In one move, Radewoosh can select a cyclic interval of exactly k balls from the first sequence and a cyclic interval of exactly k balls from the second sequence, and swap them. The goal is to make both sequences monochromatic, meaning all the balls in the first sequence are of the same color (all black or all white) and all the balls in the second sequence are of the same color.

Help Radewoosh solve his puzzle in at most n moves. It can be proven that this is always possible.

Input

The first line of the standard input contains two integers n and k ($2 \leq n \leq 300\,000; 1 \leq k \leq n-1$), as described in the task.

The next line contains a string a, consisting of exactly n characters. If the i-th character is '`B`', then the i-th ball in the first sequence is white (as Polish for "white"is "biay"). Otherwise, if the character is '`C`', the ball is black (as Polish for "black"is "czarny").

The following line contains a string b, consisting of exactly n characters, which similarly describes the second sequence of balls.

In total, the strings a and b contain exactly n '`B`' characters and n '`C`' characters.

Output

The first line of the output should contain a single number r ($0 \leq r \leq n$), indicating the number of moves you want to make.

Next, the output should contain r lines. Each of them should contain two integers. The numbers in the i-th of these lines, c_i and d_i ($1 \leq c_i, d_i \leq n$), should indicate that you are selecting a cyclic interval from the first sequence starting at position c_i and a cyclic interval from the second sequence starting at position d_i.

If $c_i + k - 1 \leq n$, then you are indicating positions $c_i, c_i+1, \ldots, c_i+k-2, c_i+k-1$ in the first sequence. If $c_i+k-1 \geq n+1$, then you are indicating positions $c_i, c_i+1, \ldots, n-1, n, 1, 2, \ldots, c_i+k-2-n, c_i+k-1-n$. The value of d_i has a similar meaning.

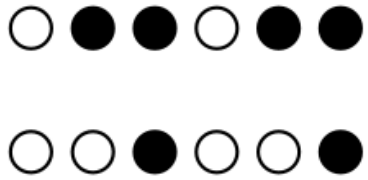
After performing all the moves described by you, all the balls in the first sequence should have the same color, and all the balls in the second sequence should have the same color. Note that you do not have to minimize the number of moves - it is enough to make at most n moves.

Example

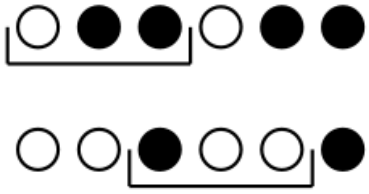
standard input	standard output
6 3 BCCBCC BBCBBC	2 1 3 5 1

Note

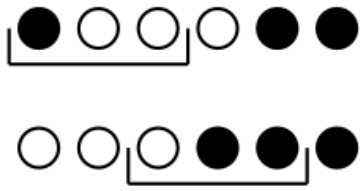
The sequences of balls initially look as follows (the first sequence is drawn above, and the second below):



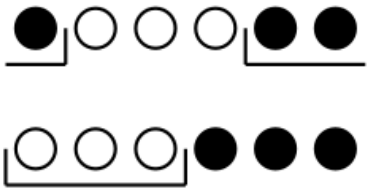
The first move selects these two intervals:



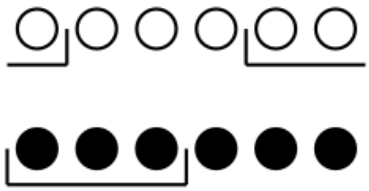
Then swaps them:



The second move selects these two intervals:



Then swaps them:



Note that it would also be correct to move all the black balls to the first sequence and all the white balls to the second sequence.

Problem H. Weather Forecast

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	1024 megabytes

Radewoosh is a meteorologist. Thanks to his research station, he has learned the predicted temperature for the next n days. The temperature for each day, expressed in Bytenheit degrees, is always a positive integer.

Radewoosh now needs to present the weather forecast on television. To make people happy, tired of the cool climate, he decided to create the impression that it will be quite warm. He has decided to divide the next n days into k non-empty intervals so that each day belongs to exactly one interval. Then, in the weather forecast, Radewoosh will only provide k numbers - the arithmetic averages of the temperatures from the days belonging to the respective intervals.

Radewoosh knows that viewers watching the forecast are very afraid of low temperatures. Therefore, he would like to make a division that maximizes the average temperature in the coldest of the periods he provides (we call a period of days the coldest if none of the remaining periods has a strictly lower average temperature). Help him make such a division!

Input

The first line of the input contains two integers n and k ($1 \le n \le 2 \cdot 10^5$, $1 \le k \le n$) representing the number of days for which Radewoosh has the temperature forecast and the number of periods into which he wants to divide it.

The second line of the input contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 1000$) representing the temperature on consecutive days, expressed in Bytenheit degrees.

Output

The output should contain a single real number, representing the maximum average temperature in the coldest of the periods. The answer will be accepted if its **absolute** error does not exceed 10^{-4}.

Example

standard input	standard output
7 3 1 3 1 2 2 2 1	1.666666666667

Note

The days can be divided into periods as follows: $[1, 3, 1], [2], [2, 2, 1]$. The average temperature in the consecutive intervals will be $\frac{5}{3}$, 2, and $\frac{5}{3}$, respectively, resulting in a minimum temperature of $\frac{5}{3}$. This division maximizes the minimum average temperature in a single period.

Problem I. Mercenaries

Input file:	`standard input`
Output file:	`standard output`
Time limit:	5 seconds
Memory limit:	1024 megabytes

In the magical land of Byteland, there are n cities arranged in a line, numbered from 1 to n from left to right, and $n-1$ roads between them. Each road connects two neighboring cities. Due to the hilly terrain, each road can only be traveled from a city with a smaller number to a city with a larger number. In other words, these passages are one-way.

In each city, there lives a mercenary. These mercenaries can be described by two numbers representing their strength and their knowledge of magic. The mercenary living in the i-th city is described by the values s_i and m_i.

There is a shop at each road. Each shop offers a certain list of items, and each item is described by its bonus to strength and bonus to knowledge of magic. These bonuses indicate how much should be added to the respective hero's statistics. More precisely, for the i-th shop (i.e., the one on the road connecting the i-th and $(i+1)$-st city), we know the list of pairs $(s'_{i,1}, m'_{i,1}), (s'_{i,2}, m'_{i,2}), \ldots, (s'_{i,r_i}, m'_{i,r_i})$, representing the bonuses of the items offered in that shop. A mercenary traveling along a road can buy at most one item at each shop he passes. There is no limit to how many items a mercenary can use at the same time—all bonuses from his items are added to his own statistics and accumulate.

Cities can be attacked by monsters. In the i-th attack scenario, the monster is described by the number of the city it attacks (denoted as v_i) and the values a_i, b_i, and c_i. They indicate that the mercenary from the j-th city will defeat the monster if $a_i \cdot S_j + b_i \cdot M_j \geq c_i$, where the values S_j and M_j are the strength and knowledge of magic of the mercenary, taking into account the bonuses he obtains by buying items on the way from his city to the city attacked by the monster. In particular, this means that $j \leq v_i$ must hold. Here, we allow $j = v_i$—then the mercenary will not have the opportunity to acquire any items. Otherwise, he will buy at most one item at each shop he passes on the way from the j-th to the v_i-th city.

Your task, as an advisor to the king of Byteland, is to prepare plans of action for all monster attack scenarios. More precisely, for each attack option i, you must find the largest city number j such that $j \leq v_i$ and the mercenary from the j-th city would be able to choose the purchased items in a way that allows him to defeat the monster. Note that the attack scenarios considered are only theoretical, and the mercenaries do not move or acquire any items.

Input

The first line of the standard input contains a single integer n ($2 \leq n \leq 200\,000$), indicating the number of cities. The next $2n-1$ lines describe the cities and shops on the roads between them.

For each i satisfying $1 \leq i \leq n$, the $(2i-1)$-th of these lines contains two integers s_i and m_i ($0 \leq s_i, m_i \leq 10^9$), indicating the strength and knowledge of magic of the mercenary living in the i-th city.

For each i satisfying $1 \leq i \leq n-1$, the $(2i)$-th of these lines starts with a single integer r_i ($1 \leq r_i \leq 500\,000$), indicating the number of items offered in the shop located on the road connecting the i-th and $(i+1)$-st city. Then, in the same line, there is a sequence of $2r_i$ integers $s'_{i,1}, m'_{i,1}, s'_{i,2}, m'_{i,2}, \ldots, s'_{i,r_i}, m'_{i,r_i}$ ($0 \leq s'_{i,j}, m'_{i,j} \leq 5\,000$), indicating the bonuses to strength and knowledge of magic of the successive items offered in that shop. The sum of all values of r_i does not exceed $500\,000$.

The next line contains a single integer q ($1 \leq q \leq 200\,000$), indicating the number of monster attack scenarios.

Each of the subsequent q lines contains a description of an attack scenario. The i-th of these lines contains four integers v_i, a_i, b_i, and c_i ($1 \leq v_i \leq n$; $0 \leq a_i, b_i \leq 10^9$; $a_i + b_i \geq 1$; $1 \leq c_i \leq 10^{18}$), as described in the problem statement.

Output

The output should consist of q lines: the i-th of them should contain a single integer, indicating the largest city number from which the mercenary would be able to defeat the monster in the i-th attack scenario, or -1 if no mercenary would be able to do so.

Example

standard input	standard output
3 1 1 2 1 2 1 2 3 2 5 1 5 4 3 3 4 5 1 1 2 4 5 12 1 1 1 1 2 1 1 1 3 1 1 1 3 1 1 9 3 2 2 20 3 1 2 18 3 1 2 19 3 1 2 20 3 0 1 8 2 1 0 4 2 1 0 3 2 1 0 2	1 2 3 3 2 2 1 -1 1 -1 2 2

Note

In the first shop, two items are sold, described by the same pairs $(1, 2)$ and $(1, 2)$. In the second shop, five items are sold, described by the pairs $(1, 5)$, $(4, 3)$, $(3, 4)$, $(5, 1)$, and $(1, 2)$.

In the first three attack scenarios, the monster attacks the first, second, and third city, respectively. In all of these scenarios, the mercenary already present in that city would be able to defeat the monster.

Let's consider the sixth, seventh, and eighth attack scenarios. In each of them, the monster attacks the third city with the same parameters a_i and b_i. In the sixth scenario, when $c_i = 18$, the monster could be defeated by the mercenary from the second city if he chose the item described by the pair $(1, 5)$. Then his strength would be $3 + 1 = 4$, and his knowledge of magic would be $2 + 5 = 7$, resulting in $1 \cdot 4 + 2 \cdot 7 = 18 \geq 18$. In the seventh scenario, when $c_i = 19$, only the mercenary from the first city could defeat the monster. By purchasing the items described by the pairs $(1, 2)$ and $(1, 5)$, his strength would be $1 + 1 + 1 = 3$, and his knowledge of magic would be $1 + 2 + 5 = 8$, resulting in $1 \cdot 3 + 2 \cdot 8 = 19 \geq 19$. In the eighth scenario, when $c_i = 20$, no mercenary would be able to defeat the monster. Note that in the sixth scenario ($c_i = 18$), the first mercenary would also be able to defeat the monster, but we are interested in the largest city number, so the answer is 2.

Problem J. Polygon II

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	1024 megabytes

Kostka would like to build a large polygon for himself. He ordered n random segments. The length of the i-th segment will be a random **real** number drawn from a uniform distribution in the range $\mathbf{(0, 2^{a_i})}$. The lengths of the individual segments are independently drawn. Kostka is interested in the probability that it is possible to construct a non-degenerate n-sided polygon from all these segments. Help him calculate it.

It can be proven that for the constraints given below, the result can be represented as a rational number $\frac{\ell}{m}$ such that the denominator m is not divisible by $10^9 + 7$. Your program for the given parameters a_i should output the remainder of dividing this fraction by $10^9 + 7$, i.e., a number x such that $m \cdot x \equiv \ell$ modulo $10^9 + 7$.

Note: Remember that the lengths of the segments are not drawn from the interval $(0, a_i)$, but from $(0, 2^{a_i})$.

Input

The first line of the input contains a single positive integer n ($3 \leq n \leq 1000$), indicating the number of ordered segments. The second line contains a sequence of n integers $a_1, \ldots, a_n$ ($0 \leq a_i \leq 50$), indicating the parameters used to draw the lengths of the individual segments.

Output

The output should contain a single integer - the remainder of the probability divided by $10^9 + 7$ that it is possible to construct a non-degenerate n-sided polygon from the random segments ordered by Kostka.

Example

standard input	standard output
3 0 2 0	166666668

Note

The sought probability is exactly $\frac{1}{6}$.

Problem K. Power Divisions

Input file:	`standard input`
Output file:	`standard output`
Time limit:	5 seconds
Memory limit:	1024 megabytes

We are given a sequence of non-negative integers $a_1, a_2, \ldots, a_n$. Based on this sequence, we create a sequence of integers $b_1, b_2, \ldots, b_n$, where $b_i = 2^{a_i}$ for each i.

A division of the sequence $b_1, b_2, \ldots, b_n$ is called a set of its contiguous intervals, such that each element belongs to exactly one interval. We call a division good if the sum of numbers in each interval is a power of two (with an integer exponent).

Your task is to count the number of good divisions of the sequence $b_1, b_2, \ldots, b_n$. Since this number can be very large, it is sufficient to provide its remainder when divided by $10^9 + 7$.

Input

The first line of the input contains a single integer n ($1 \le n \le 3 \cdot 10^5$) representing the length of the sequence $a_1, a_2, \ldots, a_n$ (and thus also the length of the sequence $b_1, b_2, \ldots, b_n$).

The second line of the input contains n integers $a_1, a_2, \ldots, a_n$ ($0 \le a_i \le 10^6$).

Output

The output should contain a single integer, representing the remainder when divided by $10^9 + 7$ of the number of good divisions of the sequence $b_1, b_2, \ldots, b_n$.

Example

standard input	standard output
5 2 0 0 1 1	6

Note

The sequence $b_1, b_2, \ldots, b_n$ in the sample test is $4, 1, 1, 2, 2$. Its good divisions are:

- $[4], [1], [1], [2], [2]$,
- $[4], [1, 1], [2], [2]$,
- $[4], [1], [1], [2, 2]$,
- $[4], [1, 1], [2, 2]$,
- $[4], [1, 1, 2], [2]$,
- $[4, 1, 1, 2], [2]$.

Problem L. Chords

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	1024 megabytes

On the circle, $2n$ points are marked, arranged at equal intervals in a clockwise direction. These points have been **randomly** (more on this in the **Input** section) connected in n pairs to form n chords. Your task is to find the largest possible set of chords in which no two intersect, and output the size of this set.

Input

The first line of the standard input contains a single integer n ($1 \le n \le 100\,000$), indicating the number of chords.

In the next n lines, there are two integers a_i and b_i ($1 \le a_i < b_i \le 2n$) each, indicating that the chord connects the a_i-th and b_i-th points. All $2n$ values of a_i and b_i are pairwise distinct.

NOTE: ~~Out of laziness of the jury~~ To make the task more interesting, all tests for this task (except for the sample test) were randomly generated. For each test, a value of n and a seed for the random number generator were chosen. Then a random permutation of numbers from 1 to $2n$ was generated, which was divided into n pairs. Then, in each pair, the numbers could be swapped to ensure the condition $a_i < b_i$.

The sample test was created manually, but in order for the solution to be accepted, it must also be solved correctly.

Output

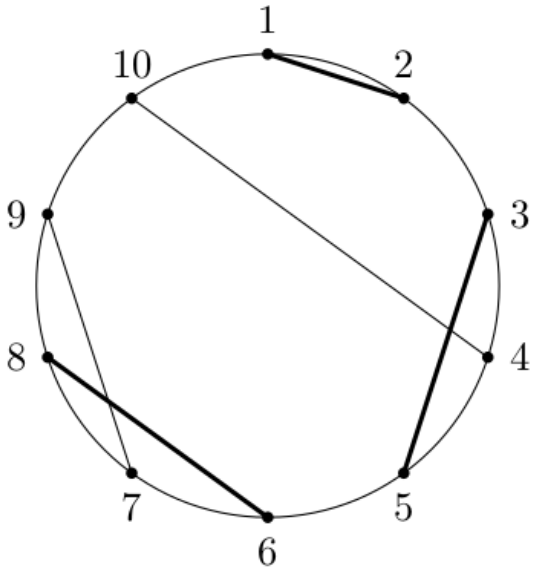
The output should contain a single integer, indicating the size of the largest set of non-intersecting chords.

Example

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

Note

The chords in the sample test look as follows:



One of the possible largest correct sets of chords contains the chords connecting points 1 and 2, 3 and 5, and 6 and 8.

Problem M. Balance of Permutation

Input file:	`standard input`
Output file:	`standard output`
Time limit:	8 seconds
Memory limit:	1024 megabytes

For a permutation p of numbers from 1 to n, its *balance* is defined as $\sum_{i=1}^{n} |p_i - i|$. Your task is to find the k-th smallest lexicographically permutation of length n with a balance equal to b.

Input

The standard input contains a single line with three integers n, b, and k ($1 \le n \le 30; 0 \le b; 1 \le k$), as described in the problem statement.

It is guaranteed that there are at least k permutations of length n with a balance equal to b.

Note that the value of the parameter k may not fit in a 64-bit data type. In `C++`, we recommend using the `__int128` type to store it.

Output

The output should contain n integers in a single line – the sought permutation.

Examples

standard input	standard output
6 6 6	1 2 6 3 4 5
30 300 303030303030303030303030	1 2 3 4 9 23 20 28 24 16 21 17 27 29 8 26 25 30 19 18 22 12 7 13 6 10 5 15 14 11

Note

Consider the first sample test. There are 46 permutations of length 6 with a balance equal to 6. Among them, the sixth smallest in lexicographic order is the permutation $[1, 2, 6, 3, 4, 5]$.

The line breaks in the second example were added to fit in the "output" section of the PDF statement. You do not need to add the line breaks in your actual output.