Hunan University 2024 the 20th Programming Contest

Contest Section

Hunan University

May 19, 2024

湖南大学 HUNAN UNIVERSITY

Problem Set

A	**……**	**The Long March of Ants**
B	**……**	**MAX MEX SUM**
C	**……**	**Clone String**
D	**……**	**L-craft (easy version)**
E	**……**	**L-craft (hard version)**
F	**……**	**Fireworks**
G	**……**	**K-shortest path**
H	**……**	**Celery Sticks**
I	**……**	**Strange clocks**
J	**……**	**Cantor Function**
K	**……**	**Knapsack for All Segments**
L	**……**	**Latest Winning Round**

The 2024 20th Hunan University Programming Contest

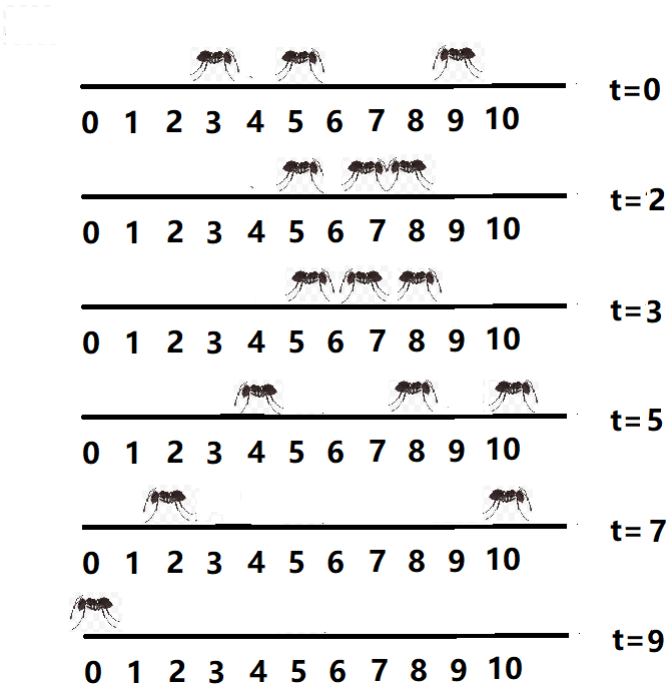
Problem A. The Long March of Ants

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

N ants move along the number axis with a length of L. Starting from their initial position, some of them move to the right (positive direction) and some move to the left (negative direction). They move at the same speed, with a length of 1 second per unit. If they meet halfway and change their direction (instantly), turn around and walk. The position markers on the number axis are from 0 to L, and ants will not stop their Long March until they reach position 0 or position L.

For example:

The initial position of three ants is 3, 5, and 9, with the first two facing positive and the last one facing negative,where L is 10. Two seconds later, the second ant and the third ant met at position 7. The second ant changed direction to negative and the third ant changed direction to positive. After another second, the first ant and the second ant met at position 6. The first ant changed direction to negative and the second ant changed direction to positive. The third ant ends its Long March at 5 seconds (from the beginning), the second ant ends its Long March at 7 seconds (from the beginning), and the first ant ends its Long March at 9 seconds (from the beginning).



Now the initial position and direction of each ant are given. Ask the time when the last ant to complete the Long March.

Input

The first line contains a integer T indicating the number of testcases.

For each test case, there are two lines with the first line containing two positive integers N and L, representing N ants and the length of the number axis. ($N \leq 10,000, L \leq 1,000,000,000, N < L$).

The next line contains N integers $p_i (1 \leq i \leq N, 0 < |p_i| < L)$, representing the initial position and direction of the i-th ant.

If p_i is a positive integer, it indicates that the initial position p_i and initial direction of the i-th ant are positive.

If p_i is a negative integer, it indicates that the initial position $-p_i$ and initial direction of the i-th ant are negative.

No two ants have the same initial position. There is no initial position for ants at the end position.

Examples

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

Problem B. MAX MEX SUM

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

For a given integer n as the length of an arbitrary permutation p indexed from 1 to n, find the maximum value of $\sum_{1\le l\le r\le n} \operatorname{mex}(a[l,r])$.

A permutation of length n is an array of length n that each elements of this array is not less than 1 and not greater than n, and each number appears exactly once. $a[l,r]$ denotes the subarray of a from l to r.

mex is a function that for an input array a, $\operatorname{mex}(a)$ returns the minimum positive integer that does not appear in array a.

Input

The input contains T testcases, the first line is an integer T ($1 \le T \le 10^5$), denoting the number of testcases.

For each test case, the input contains only one integer n ($1 \le n \le 10^9$) in one line, denoting n as the length of the permutation p.

Output

For each testcase, output one integer in one line, denoting the answer. Since this number maybe very large, output the maximum value modulo 998244353.

Examples

standard input	standard output
5 1 2 3 4 5	2 6 13 23 37
3 16707 62135 1000000	537374136 226433365 99413837

Note

For the second test case, given the inputs $n = \{16707, 62135, 1000000\}$, the answers are $\{388854427453, 19994062579602, 83334208334250000\}$.

Therefore, the outputs are $\{537374136, 226433365, 99413837\}$ after applying modulo operation with 998244353.

Problem C. Clone String

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

displaylzy recently like to study the copying and querying of strings, he found that after the string is copied, the change of position will have a certain pattern now he would like to ask you to solve the queries about the position of the string after copying.

Specifically, a string of length n with m operations will be given.

There are two kinds of operations.

$(1, x, len, y)$: Starting from the x-th position of the string, take len length of the string and then clone this substring y times.

By cloning, each letter will be repeated y times.

For example,

for abaab with $n = 5$, the operation $(1, 1, 2, 3)$ is applied,

i.e. $(ab)aab$ becomes $(aaabbb)aab$

$(2, x)$: Query the x-th character of the current string.

Input

The input contains $2 + m$ lines.

The first line contains two integers, n and m, $(n, m \leq 10^5)$, denoting the length of the initial string and the number of operations.

The second line contains a string s of length n, denoting the initial string.

The next m lines, each line denotes an operation.

It is guaranteed that all operations are legal. The length of the string after at each time will not exceed 10^{18}.

Output

For all queries like $(2, x)$, output the character in the corresponding position at that time.

Example

standard input	standard output
5 3 abaab 2 2 1 1 2 3 2 2	b a

Problem D. L-craft (easy version)

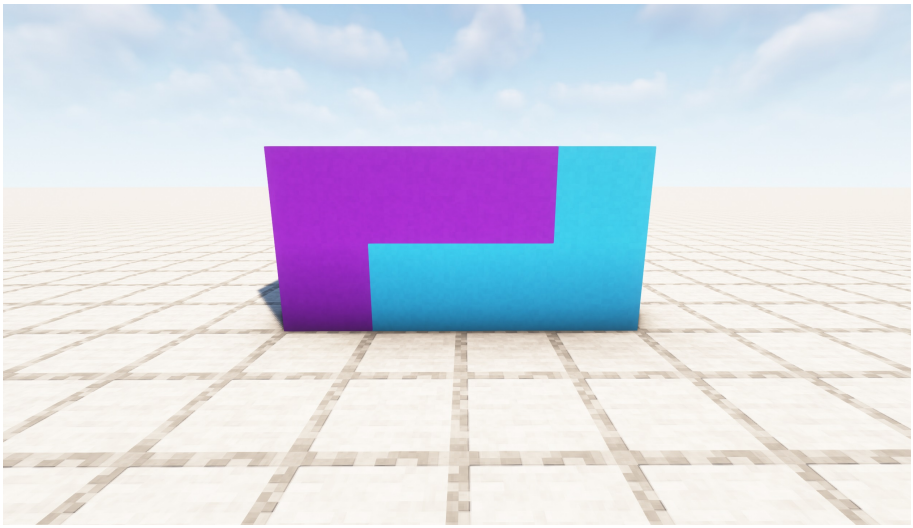
Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

This is the easy version of the problem. The differences between two versions are the constraint of n and m.

Vistar loves to play Minecraft.

Minecraft takes place in a three-dimensional grid world, each occupied by a specific type of cube. It focuses on allowing players to explore, interact with, and change the infinite world. And one of Vistar's favorite gameplay styles is creating buildings.

Recently, he was working on a renovation of his castle. There is an $n \times m$ sized wall in the castle, and to make it artistic, he wants to fill this wall with L-shaped concrete cubes. For example, a 2×4 example is shown below:



This wall contains 2 L-shapes.

However, Vistar soon realizes that some walls cannot be completely tessellated(i.e., tightly paved) with an L-shape. He wishes to know that, given n and m, what is the maximum number of L-shapes that the wall can be filled with?

Since Vistar likes 8 and its multiples, he wants at least one of n and m to be a multiple of 8.

Input

The first line contains a single integer t ($1 \le t \le 10^4$) — the number of test cases.

The first and only line of each test case contains two integers n and m ($1 \le n, m \le 2 \cdot 10^5$, at least one of n and m is a multiple of 8).

It is guaranteed that the sum of $n \times m$ over all test cases does not exceed $2 \cdot 10^5$.

Output

For each test case:

Output an $n \times m$ rectangle of non-negative integers to represent your solution. For each positive integer value, it should **exactly form an L-shape**. For example, the following output:

1	2	2	2	1	0	3	3
1	1	1	2	1	1	1	3

is invalid, because the number 1 appears in two L-shapes and the number 3 doesn't form an L-shape.

Example

standard input	standard output
1 2 8	1 2 2 2 3 4 4 4 1 1 1 2 3 3 3 4

Problem E. L-craft (hard version)

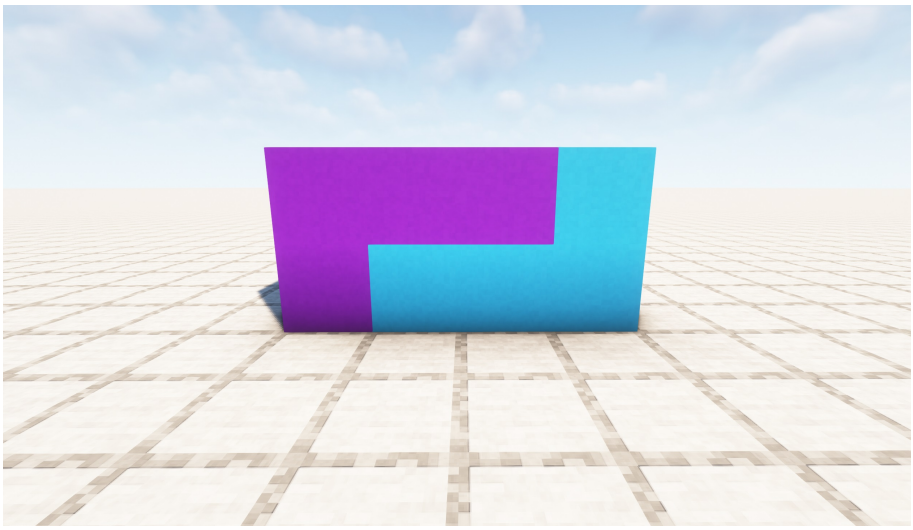
Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

This is the hard version of the problem. The differences between two versions are the constraint of n and m.

Vistar loves to play Minecraft.

Minecraft takes place in a three-dimensional grid world, each occupied by a specific type of cube. It focuses on allowing players to explore, interact with, and change the infinite world. And one of Vistar's favorite gameplay styles is creating buildings.

Recently, he was working on a renovation of his castle. There is an $n \times m$ sized wall in the castle, and to make it artistic, he wants to fill this wall with L-shaped concrete cubes. For example, a 2×4 example is shown below:



This wall contains 2 L-shapes.

However, Vistar soon realizes that some walls cannot be completely tessellated(i.e., tightly paved) with an L-shape. He wishes to know that, given n and m, what is the maximum number of L-shapes that the wall can be filled with?

Since Vistar doesn't want his castle to be too odd, he prefers at least one of n and m to be an even number.

Input

The first line contains a single integer t ($1 \le t \le 10^4$) — the number of test cases.

The first and only line of each test case contains two integers n and m ($1 \le n, m \le 2 \cdot 10^5$, at least one of n and m is even).

It is guaranteed that the sum of $n \times m$ over all test cases does not exceed $2 \cdot 10^5$.

Output

For each test case:

Output an $n \times m$ rectangle of non-negative integers to represent your solution. For each positive integer value, it should **exactly form an L-shape**. For example, the following output:

1	2	2	2	1	0	3	3
1	1	1	2	1	1	1	3

is invalid, because the number 1 appears in two L-shapes and the number 3 doesn't form an L-shape.

Example

standard input	standard output
2 2 4 3 6	1 2 2 2 1 1 1 2 1 2 2 3 4 4 1 0 2 3 0 4 1 1 2 3 3 4

Problem F. Fireworks

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

It's family beaches that I desire
Sacred nights, where we watch the fireworks...

— Animal Collective, *Fireworks*

At a family gathering, you want to go to the fireworks. Unfortunately, flashy and splendid sparklers would frighten the babies.

There are n locations where fireworks can be placed. Let's define a_i as the viewing level of location i and b_i as the frightening level of location i, where a_i and b_i can only take values of 0 or 1. In order to enjoy the beauty of the fireworks while ensuring a pleasant atmosphere, you are going to pick a continuous section $[l, r]$ $(1 \le l \le r \le n)$ to place the fireworks, in such a way that the value of the following function is maximized:

$$f(l,r) = \sum_{i=l}^{r} a_i - \Big(\sum_{i=l}^{r} b_i\Big)^2$$

Input

The first line of input contains one integer n $(1 \le n \le 10^5)$ — the length of the array.

The second line contains an array a of length n $(a_i \in \{0, 1\})$.

The third line contains an array b of length n $(b_i \in \{0, 1\})$.

Output

Print a single integer, representing the maximum value of f.

Examples

standard input	standard output
5 1 0 0 0 1 0 1 0 0 0	1
11 1 0 1 1 1 1 1 1 1 1 1 0 0 0 1 0 0 0 0 1 1 0	6

Problem G. K-shortest path

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 megabytes

Given a directed weighted graph with the possibility of having parallel edges, self-loops, and negative weights. The graph consists of N nodes and M directed edges. For every pair of nodes, determine the minimum distance achievable between them by traversing **no more than** K directed edges, with the option of repeatedly passing through the same edge.

Formally speaking. Let edge i go from node u_i to v_i with weight w_i. A path from u to v traversing k edges is defined as an array p of length $len_p = k$, where $1 \le p_i \le m, u_{p_1} = u, v_{p_k} = v, v_{p_i} = u_{p_{i+1}}$ for $1 \le i < k$, and $dis_p = \sum_{i=1}^{k} w_{p_i}$ represents the distance of the path. For each u, $v(1 \le u, v \le N)$, you should calculate $\min_{p, len_p \le K} dis_p$.

Input

The first line contains three integers: N, M, and $K(2 \le N \le 100, 1 \le M \le 10^4, 0 \le K \le 10^9)$. The next M lines each contain three integers: u_i, v_i, and w_i, indicating a directed edge from node u_i to node v_i with weight w_i $(1 \le u_i, v_i \le N, -10^9 \le w_i \le 10^9)$.

Output

Output an $N \times N$ matrix, where the element at row i and column j represents the minimum distance from node i to node j while traversing through at most K edges. If it is not possible to reach node j from node i, output "NO"(without quotes).

Examples

standard input	standard output
5 6 3 1 2 3 1 3 1 3 2 1 1 4 1 4 5 1 5 2 1	0 2 1 1 2 NO 0 NO NO NO NO 1 0 NO NO NO 2 NO 0 1 NO 1 NO NO 0
5 10 16 1 2 7 2 1 9 2 2 -1 2 1 -4 3 5 4 3 2 2 3 1 -3 4 5 2 5 5 0 5 3 -3	-11 -8 NO NO NO -19 -16 NO NO NO -16 -13 0 NO 4 -15 -12 -1 0 2 -18 -15 -3 NO 0

Problem H. Celery Sticks

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

As a celery sticks manufacturer, you have a celery stick of length n.

There is a standard length for the celery sticks, denoted as l. Deviations from this standard length will result in a lower selling price. Specifically, if the length of a celery stick is x, the selling price would be $p - |x - l|$, where p is a given value. Note that the selling price can be negative, as you should pay for garbage disposal.

You can cut the celery sticks in any way you wish, as long as the lengths of the pieces are still integers, to maximize your profit after selling them. Please determine the total maximum profit you can achieve.

Input

The input contains T test cases, the first line is an integer T ($1 \le T \le 10^5$), denoting the number of test cases.

For each test case, the input contains three integers n, p, and l, ($1 \le n, p, l \le 10^9$), denotes the length of initial celery stick, the highest selling price of a celery stick, and the standard length of the celery sticks.

Output

For each test case, output the maximum profit you can get.

Example

standard input	standard output
6 10 5 2 10 1 3 11 5 2 11 1 3 12 8 7 10 1 20	40 2 44 3 24 -9

Note

In the 1st test case, cut the celery stick into 10 celery sticks of length 1, each sells \$4, the total profit is \$40.

In the 2rd test case, cut the celery stick into the lengths of {3,3,4}, the selling price would be {\$1,\$1,\$0}, therefore, the total profit is \$2.

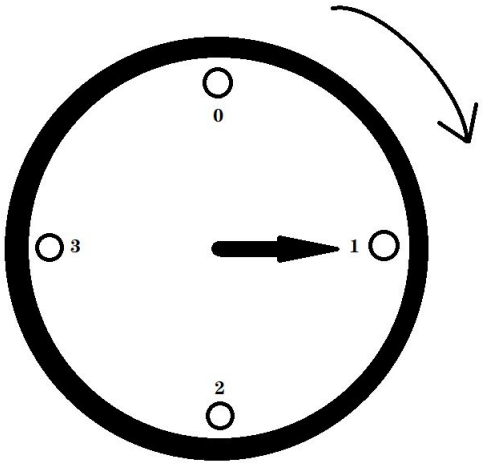
In the 6th test case, do nothing to get the selling price \$-9, which is the best solution. Any cut will result in much money loss.

Problem I. Strange clocks

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Welcome to the Time Management Bureau !

We need to calibrate some clocks that are affected by time floods. There is a row of n clocks to be calibrated here, with the scale on each clock starting from 0 and placed clockwise at $1, 2$..... Each clock provides two numbers x and y to indicate that it needs to rotate x times to complete one revolution and has already completed y rotations from the 0 scale.

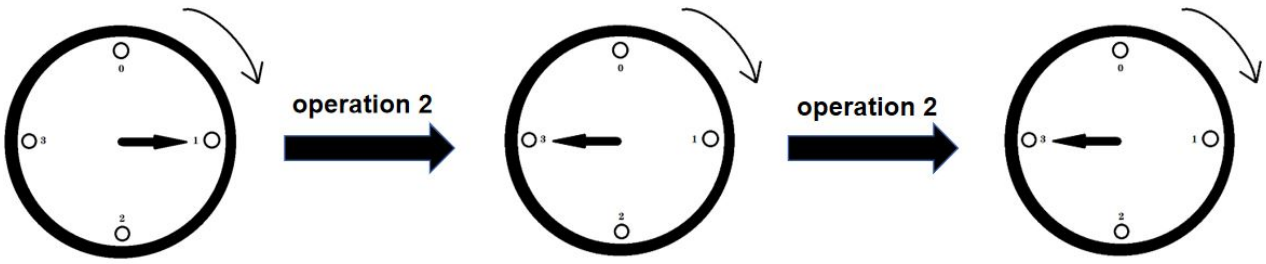


The clock with $x = 4$ $y = 1$.

We have the three calibration operations:

1. For each $i \in [1, n]$, let $y_i = [y_i \& (y_i + 1)] \mod x_i$.
2. For each $i \in [1, n]$, let $y_i = [y_i \oplus (y_i + 1)] \mod x_i$.
3. For each $i \in [1, n]$, let $y_i = (y_i + 1) \mod x_i$.

Here & denotes the bitwise AND operation, and $\oplus$ denotes the bitwise XOR operation.



An example of operation two

The Time Management Bureau needs to calibrate these n clocks to the correct scale through several operations. But to save time, please calculate the minimum number of operations to calibrate these n clocks to the correct scale.

Output -1 when there is no solution.

Input

The first line contains one positive integer $n(1 \leq n \leq 40)$.

The second line contains n positive integers separated by spaces to represent the correct scale sequence, ensuring that the sequence is legal.

The next n lines, each contains two positive integers $x(2 \leq x \leq 4)$ and $y(0 \leq y \leq x-1)$ separated by spaces.

Output

The output contains an integer representing the minimum number of operations to calibrate all clocks.

Example

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

Note

In the first test case, let's first perform operation 1 and then operation 2:

Scale sequence after completing operation 1: $0, 0, 0, 2, 2$

Scale sequence after completing operation 2: $1, 1, 1, 1, 1$

Problem J. Cantor Function

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Little X learned about the Cantor function from their Advanced Mathematics class. And he thinks it is interesting, so he wants to calculate this function.

The Cantor function is a function $C : [0, 1] \to [0, 1]$ that is defined as follows:

1. Write the ternary expansion of $x \in [0, 1]$. That is, $x = \sum_{n=1}^{\infty} \frac{a_n}{3^n}$, where each a_n is either 0 or 2.
2. Define the sequence $\{y_n\}_{n=1}^{\infty}$ by deleting all of the 1s from the ternary expansion of x after the nth digit and replacing all of the 2s with 1s. That is, $y_n = \sum_{i=1}^{n} \frac{b_i}{2^i}$ where $b_i = 0$ if the ith digit in the ternary expansion of x after the first n digits is 0, and $b_i = 1$ if it is 2.
3. Define $C(x)$ to be the limit of the sequence $\{y_n\}_{n=1}^{\infty}$.

The Cantor function has several properties:

1. C is continuous, non-decreasing, and maps $[0, 1]$ onto $[0, 1]$.
2. C is constant on each interval $[a, b] \subset [0, 1]$ that is not in the middle-third Cantor set.
3. C is strictly increasing on each interval $(a, b) \subseteq [0, 1]$ that is entirely contained in the middle-third Cantor set.
4. C is differentiable almost everywhere, and its derivative is zero almost everywhere on $[0, 1]$.

These properties make the Cantor function an important example in mathematical analysis and fractal geometry.

To calculate it for a specific value of x, you can follow these steps:

1. Write the ternary expansion of x. For example, if $x = \frac{13}{18}$, the ternary expansion would be $0.20111..._{(3)}$
2. Delete all digit after the first occurrences of 1 from the ternary expansion. In the case of $x = \frac{13}{18}$, the resulting sequence would be $0.201_{(3)}$
3. Replace all digits 2 with 1. In the case of $x = \frac{13}{18}$, the resulting sequence would be $0.101_{(3)}$
4. Convert the resulting sequence back to decimal representation. The result is the value of the Cantor function at x. In the case of $x = \frac{13}{18}$, the resulting sequence would be $0.101_{(2)} = \frac{5}{8}$

Input

Input one line that cantains two integers a, b ($1 \le a < b \le 1e18$), representing the value of $x = \frac{a}{b}$ for which Little X want to calculate the Cantor function.

Condition1: Given a fraction $\frac{a}{b} = 0.eooo..._{(3)}$, where the digits in the repeating block are denoted by 'o'. For example, $\frac{13}{18} = 0.20111..._{(3)}$, where $e = 20$ and $o = 1$. It is guaranteed that $|e| + |o| \le 60$ for the input $\frac{a}{b}$.

Condition2: Given a fraction $\frac{a}{b}$, the number of digits before the first digit '1' is c. For example, for $\frac{13}{18} = 0.20111..._{(3)}$, the number of digits before the first digit '1' is 2. It is guaranteed that $c \le 60$ for the input $\frac{a}{b}$.

Note: For every input $\frac{x}{y}$, it is guaranteed to satisfy **at least one** of the prementioned conditions.

Output

The output should only contain two integers p and $q(1 \leq p \leq q, gcd(b, q) = 1)$, representing the fraction $\frac{p}{q}$ such that $C(\frac{a}{b}) = \frac{p}{q}$.

Examples

standard input	standard output
13 18	5 8
3 4	2 3
25 36	7 12

Note

For the 2nd example, $\frac{3}{4} = 0.202020...{}_{(3)} \xrightarrow{Delete} 0.202020...{}_{(3)} \xrightarrow{Replace} 0.101010...{}_{(3)} \xrightarrow{Review} 0.101010...{}_{(2)} = \frac{2}{3}$.

Problem K. Knapsack for All Segments

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Given are a sequence of N integers $A_1, A_2, \ldots, A_N$ and a positive integer S.

For a pair of integers (L, R) such that $1 \leq L \leq R \leq N$, let us define $f(L, R)$ as follows:

- $f(L, R)$ is the number of sequences of integers $(x_1, x_2, \ldots, x_k)$ such that $L \leq x_1 < x_2 < \cdots < x_k \leq R$ and $A_{x_1} + A_{x_2} + \cdots + A_{x_k} = S$.

Find the sum of $f(L, R)$ over all pairs of integers (L, R) such that $1 \leq L \leq R \leq N$. Since this sum can be enormous, print it modulo 998244353.

Input

The first line contains two integers N, S.

The second line contains N integers $A_1, A_2, \ldots, A_N$.

$1 \leq N, S, A_i \leq 3000$.

Output

Print the sum of $f(L, R)$, modulo 998244353.

Example

standard input	standard output
3 4 2 2 4	5

Note

For the first example:

The value of $f(L, R)$ for each pair is as follows, for a total of 5.

- $f(1, 1) = 0$
- $f(1, 2) = 1$ (for the sequence $(1, 2)$)
- $f(1, 3) = 2$ (for $(1, 2)$ and (3))
- $f(2, 2) = 0$
- $f(2, 3) = 1$ (for (3))
- $f(3, 3) = 1$ (for (3))

Problem L. Latest Winning Round

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

displaylzy has recently been working on a tournament called the "X Points to the Top"system.

Specifically:

There are n players, and at the end of each round, each player will receive a corresponding number of points. The player who ranks i-th place in the round will receive $n - i + 1$ points.

The rule is that when a player has a total score greater than or equal to X ($score \geq X$) at the end of a round, then that player **gets a chance** to reach the top.

When a player who **has been given a chance** to reach the top **ranks first** in a round, the game ends and that player wins.

Now, given a game situation of n players and a summit score X,

Can you determine what is the maximum number of rounds the game will continue after a specific scenario before a player reaches the top?

Input

The first line contains two integers n and X, $(n, X \leq 2000)$, denotes the number of players and the summit point.

The second line contains n integers, denotes the current points of each players.

It is guaranteed that the sum of the current points over all players will not exceed 10^5.

Output

Output the maximum number of game rounds to go through after the given scenario.

Examples

standard input	standard output
3 8 2 3 1	6
5 20 9 5 5 5 6	9

Note

A given game situation must be legal, i.e., derived from a number of rounds of the game.