Educational Codeforces Round 94 (Rated for Div. 2)

A. String Similarity

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

A binary string is a string where each character is either 0 or 1. Two binary strings a and b of equal length are *similar*, if they have the same character in some position (there exists an integer i such that $a_i = b_i$). For example:

- 10010 and 01111 are *similar* (they have the same character in position 4);
- 10010 and 11111 are *similar*;
- 111 and 111 are *similar*;
- 0110 and 1001 are not *similar*.

You are given an integer n and a binary string s consisting of $2n - 1$ characters. Let's denote $s[l..r]$ as the contiguous substring of s starting with l-th character and ending with r-th character (in other words, $s[l..r] = s_l s_{l+1} s_{l+2} \dots s_r$).

You have to construct a binary string w of length n which is *similar* to **all of the following strings**: $s[1..n]$, $s[2..n+1]$, $s[3..n+2]$, ..., $s[n..2n-1]$.

Input

The first line contains a single integer t ($1 \le t \le 1000$) — the number of test cases.

The first line of each test case contains a single integer n ($1 \le n \le 50$).

The second line of each test case contains the binary string s of length $2n - 1$. Each character s_i is either 0 or 1.

Output

For each test case, print the corresponding binary string w of length n. If there are multiple such strings — print any of them. It can be shown that at least one string w meeting the constraints always exists.

Example

input
4 1 1 3 00000 4 1110000 2 101
output
1 000 1010 00

Note

The explanation of the sample case (equal characters in equal positions are bold):

The first test case:

- $\mathbf{1}$ is similar to $s[1..1] = \mathbf{1}$.

The second test case:

- $\mathbf{000}$ is similar to $s[1..3] = \mathbf{000}$;
- $\mathbf{000}$ is similar to $s[2..4] = \mathbf{000}$;
- $\mathbf{000}$ is similar to $s[3..5] = \mathbf{000}$.

The third test case:

- $\mathbf{1010}$ is similar to $s[1..4] = \mathbf{1110}$;
- $\mathbf{1010}$ is similar to $s[2..5] = \mathbf{1100}$;
- $\mathbf{1010}$ is similar to $s[3..6] = \mathbf{1000}$;
- $\mathbf{1010}$ is similar to $s[4..7] = \mathbf{0000}$.

The fourth test case:

- $\mathbf{0}0$ is similar to $s[1..2] = \mathbf{10}$;
- $\mathbf{0}0$ is similar to $s[2..3] = \mathbf{01}$.

B. RPG Protagonist

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

You are playing one RPG from the 2010s. You are planning to raise your smithing skill, so you need as many resources as possible. So how to get resources? By stealing, of course.

You decided to rob a town's blacksmith and you take a follower with you. You can carry at most p units and your follower — at most f units.

In the blacksmith shop, you found cnt_s swords and cnt_w war axes. Each sword weights s units and each war axe — w units. You don't care what to take, since each of them will melt into one steel ingot.

What is the maximum number of weapons (both swords and war axes) you and your follower can carry out from the shop?

Input

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

The first line of each test case contains two integers p and f ($1 \le p, f \le 10^9$) — yours and your follower's capacities.

The second line of each test case contains two integers cnt_s and cnt_w ($1 \le cnt_s, cnt_w \le 2 \cdot 10^5$) — the number of swords and war axes in the shop.

The third line of each test case contains two integers s and w ($1 \le s, w \le 10^9$) — the weights of each sword and each war axe.

It's guaranteed that the total number of swords and the total number of war axes in all test cases don't exceed $2 \cdot 10^5$.

Output

For each test case, print the maximum number of weapons (both swords and war axes) you and your follower can carry.

Example

input
3 33 27 6 10 5 6 100 200 10 10 5 5 1 19 1 3 19 5
output
11 20 3

Note

In the first test case:

- you should take 3 swords and 3 war axes: $3 \cdot 5 + 3 \cdot 6 = 33 \le 33$
- and your follower — 3 swords and 2 war axes: $3 \cdot 5 + 2 \cdot 6 = 27 \le 27$.

$3 + 3 + 3 + 2 = 11$ weapons in total.

In the second test case, you can take all available weapons even without your follower's help, since $5 \cdot 10 + 5 \cdot 10 \le 100$.

In the third test case, you can't take anything, but your follower can take 3 war axes: $3 \cdot 5 \le 19$.

C. Binary String Reconstruction

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Consider the following process. You have a binary string (a string where each character is either 0 or 1) w of length n and an integer x. You build a new binary string s consisting of n characters. The i-th character of s is chosen as follows:

- if the character w_{i-x} exists and is equal to 1, then s_i is 1 (formally, if $i > x$ and $w_{i-x} = 1$, then $s_i = 1$);
- if the character w_{i+x} exists and is equal to 1, then s_i is 1 (formally, if $i + x \le n$ and $w_{i+x} = 1$, then $s_i = 1$);
- if both of the aforementioned conditions are false, then s_i is 0.

You are given the integer x and the resulting string s. Reconstruct the original string w.

Input
The first line contains one integer t ($1 \le t \le 1000$) — the number of test cases.

Each test case consists of two lines. The first line contains the resulting string s ($2 \le |s| \le 10^5$, each character of s is either `0` or `1`). The second line contains one integer x ($1 \le x \le |s| - 1$).

The total length of all strings s in the input does not exceed 10^5.

Output
For each test case, print the answer on a separate line as follows:

- if no string w can produce the string s at the end of the process, print -1;
- otherwise, print the binary string w consisting of $|s|$ characters. If there are multiple answers, print any of them.

Example

input
3 101110 2 01 1 110 1
output
111011 10 -1

D. Zigzags

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

You are given an array $a_1, a_2 \dots a_n$. Calculate the number of tuples (i, j, k, l) such that:

- $1 \le i < j < k < l \le n$;
- $a_i = a_k$ and $a_j = a_l$;

Input
The first line contains a single integer t ($1 \le t \le 100$) — the number of test cases.

The first line of each test case contains a single integer n ($4 \le n \le 3000$) — the size of the array a.

The second line of each test case contains n integers $a_1, a_2, \dots, a_n$ ($1 \le a_i \le n$) — the array a.

It's guaranteed that the sum of n in one test doesn't exceed 3000.

Output
For each test case, print the number of described tuples.

Example

input
2 5 2 2 2 2 2 6 1 3 3 1 2 3
output
5 2

Note
In the first test case, for any four indices $i < j < k < l$ are valid, so the answer is the number of tuples.

In the second test case, there are 2 valid tuples:

- $(1, 2, 4, 6)$: $a_1 = a_4$ and $a_2 = a_6$;
- $(1, 3, 4, 6)$: $a_1 = a_4$ and $a_3 = a_6$.

E. Clear the Multiset

time limit per test: 2 seconds

memory limit per test: 256 megabytes
input: standard input
output: standard output

You have a multiset containing several integers. Initially, it contains a_1 elements equal to 1, a_2 elements equal to 2, ..., a_n elements equal to n.

You may apply two types of operations:

- choose two integers l and r ($l \le r$), then remove one occurrence of l, one occurrence of $l + 1$, ..., one occurrence of r from the multiset. This operation can be applied only if each number from l to r occurs at least once in the multiset;
- choose two integers i and x ($x \ge 1$), then remove x occurrences of i from the multiset. This operation can be applied only if the multiset contains at least x occurrences of i.

What is the minimum number of operations required to delete all elements from the multiset?

Input
The first line contains one integer n ($1 \le n \le 5000$).

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

Output
Print one integer — the minimum number of operations required to delete all elements from the multiset.

Examples

input
4 1 4 1 1
output
2

input
5 1 0 1 0 1
output
3

F. x-prime Substrings

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

You are given an integer value x and a string s consisting of digits from 1 to 9 inclusive.

A substring of a string is a contiguous subsequence of that string.

Let $f(l, r)$ be the sum of digits of a substring $s[l..r]$.

Let's call substring $s[l_1..r_1]$ x-*prime* if

- $f(l_1, r_1) = x$;
- there are no values l_2, r_2 such that
  - $l_1 \le l_2 \le r_2 \le r_1$;
  - $f(l_2, r_2) \ne x$;
  - x is divisible by $f(l_2, r_2)$.

You are allowed to erase some characters from the string. If you erase a character, the two resulting parts of the string are concatenated without changing their order.

What is the minimum number of characters you should erase from the string so that there are no x-*prime* substrings in it? If there are no x-*prime* substrings in the given string s, then print 0.

Input
The first line contains a string s ($1 \le |s| \le 1000$). s contains only digits from 1 to 9 inclusive.

The second line contains an integer x ($1 \le x \le 20$).

Output
Print a single integer — the minimum number of characters you should erase from the string so that there are no x-*prime* substrings in it. If there are no x-*prime* substrings in the given string s, then print 0.

Examples

input
116285317 8
output
2

input
314159265359 1
output
2

input
13 13
output
0

input
3434343434 7
output
5

Note

In the first example there are two 8-prime substrings "8" and "53". You can erase these characters to get rid of both: "116285317". The resulting string "1162317" contains no 8-prime substrings. Removing these characters is also a valid answer: "116285317".

In the second example you just have to erase both ones.

In the third example there are no 13-prime substrings. There are no substrings with the sum of digits equal to 13 at all.

In the fourth example you can have neither "34", nor "43" in a string. Thus, you have to erase either all threes or all fours. There are 5 of each of them, so it doesn't matter which.

G. Mercenaries

time limit per test: 7 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

Polycarp plays a (yet another!) strategic computer game. In this game, he leads an army of mercenaries.

Polycarp wants to gather his army for a quest. There are n mercenaries for hire, and the army should consist of some subset of them.

The i-th mercenary can be chosen if the **resulting** number of chosen mercenaries is not less than l_i (otherwise he deems the quest to be doomed) and not greater than r_i (he doesn't want to share the trophies with too many other mercenaries). Furthermore, m pairs of mercenaries hate each other and cannot be chosen for the same quest.

How many **non-empty** subsets does Polycarp need to consider? In other words, calculate the number of non-empty subsets of mercenaries such that the size of this subset belongs to $[l_i, r_i]$ for each chosen mercenary, and there are no two mercenaries in the subset that hate each other.

The answer may be large, so calculate it modulo 998244353.

Input

The first line contains two integers n and m ($1 \le n \le 3 \cdot 10^5$, $0 \le m \le \min(20, \frac{n(n-1)}{2})$) — the number of mercenaries and the number of pairs of mercenaries that hate each other.

Then n lines follow, the i-th of them contains two integers l_i and r_i ($1 \le l_i \le r_i \le n$).

Then m lines follow, the i-th of them contains two integers a_i and b_i ($1 \le a_i < b_i \le n$) denoting that the mercenaries a_i and b_i hate each other. There are no two equal pairs in this list.

Output

Print one integer — the number of non-empty subsets meeting the constraints, taken modulo 998244353.

Examples

input
3 0 1 1 2 3 1 3
output
3

input
3 1 1 1 2 3 1 3 2 3
output
2