Problem A. Magic Computer

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

Bob has a magic computer which has infinite USB ports. But there is a problem with this computer——it can only read and write the two most recently inserted USB disks.

Initially, Bob has n USB disks and every USB disk has one unique file. He wants to insert all the n USB disks into his computer. Since his computer is old, he must insert USB disk in the following way:

1. Select 2 unplugged USB disks, insert them into computer.
2. The two USB disks inserted in the previous step will share the files with each other, which means after that both two USB disks will have all files either in the first one or in the second one.
3. If all the n USB disks are in the computer, the task is finished. Otherwise, choose one USB disk in the computer, unplug it and go to step 1.

Bob wants to know the minimum number of n such that there is a sequence of operations that ends up with at least k different files in each USB disk.

Input

Only one number,k ($2 \leq k \leq 100000$),meaning that finally each USB disk owns at least k different files.

Output

You need to output one number n, which means the minimum number of USB disks in module 998244353.

Examples

standard input	standard output
3	4
5	16
1000	510735315

Problem B. Chevonne's Game

Input file:	`standard input`
Output file:	`standard output`
Time limit:	5 seconds
Memory limit:	512 megabytes

Would you like to play a game with Chevonne?

Chevonne is a kind witch. She has many magic pearls, the color of which is either white or black. The pearls are in a line and numbered from 1 to n. Chevonne may change the color of some of the pearls while giving you a number of queries. Anyone who can answer all the queries correctly will be blessed by her and get an "Accepted".

Specifically, Chevonne will perform q actions in the following two forms:

- `M L R` Chevonne uses the magic to change the color(white to black and black to white) of the pearls with the number from L to R.
- `Q L R` Chevonne gives you a query about the minimum number of times needed to take away all the pearls with the number from L to R. You can only take away the pearls in the following way: Select several continuous pearls(the concept of which is similar to sub-string), and every two adjacent pearls of them have different colors (selecting one pearl is always OK). Take away these pearls and merge the remaining pearls(if any) to form a whole.

Input

The first line contains two integers n ($1 \le n \le 10^6$) and q ($1 \le q \le 10^6$).

The second line contains a binary string s of length n, the ith character represents the color of the pearl with the number i(0 represents white and 1 represents black).

Following comes q lines. Each line contains one character and two integers L, R ($1 \le L \le R \le n$), representing an action Chevonne will perform.

Output

For each query, please print one line of a single number representing the answer of the corresponding query.

Example

standard input	standard output
8 5 10001101 Q 1 3 Q 1 8 M 3 5 Q 1 3 Q 1 8	2 3 1 2

Note

For the first query, we need to figure out the minimal number of times to take away the first three pearls"100".

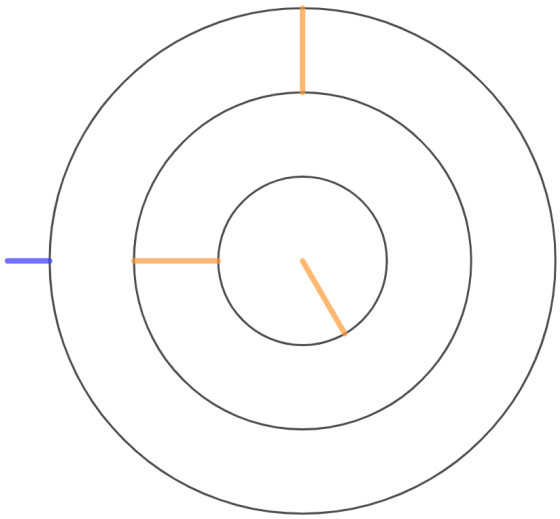
The answer is 2. A feasible solution is:

1. Take away the first two pearls"10"; 2. Take away the remaining one pearl"0".

Problem C. Compass

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

Honkai: Star Rail is a game where players may encounter a puzzle called the *Navigation Compass* when arriving at *Xianzhou Luofu*. This puzzle consists of three layers, each with an orange line.



This puzzle aims to align all the orange lines with the blue line by rotating the layers **clockwise**. As the layers are rotated, the orange lines will shift position, but the blue line remains fixed. Due to astronomical reasons, the degree to which each layer can be rotated in a single step is predetermined. For instance, if layer 1 can rotate by 60° per step, it will take 6 steps to complete a full rotation.

Mr. Wor is stuck on the final level of this puzzle, where the player is only allowed to rotate any two layers simultaneously in each step. Given the initial positions of the orange lines and the number of steps required for each layer to complete a full rotation, please find the minimum number of steps required to align all the lines.

In other words, you are given 6 integers $x_0, x_1, x_2, y_0, y_1, y_2$, please find three nonnegative integers t_0, t_1, t_2, such that $x_0 + t_1 + t_2 \equiv 0 (\mathrm{mod} y_0), x_1 + t_0 + t_2 \equiv 0 (\mathrm{mod} y_1), x_2 + t_0 + t_1 \equiv 0 (\mathrm{mod} y_2)$, and minimize $t_0 + t_1 + t_2$.

Input

The first line contains one integer $T (1 \leq T \leq 10)$, denoting the number of test cases.

Then in the next T lines, each line contains 6 integers $x_0, x_1, x_2, y_0, y_1, y_2 (\forall i, 0 \leq x_i < y_i \leq 2000)$.

Output

For each test case, print the minimum value $t_0 + t_1 + t_2$. If there is no solution, print -1.

Example

standard input	standard output
3 1 1 1 2 3 4 1 1 0 2 3 4 1 1 1 4 7 5	3 5 9

Problem D. Pandemic

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

Kanade's university was recently forced to close due to a pandemic. The students were locked in the dormitory building and were not allowed to go out. But fortunately, the university has prepared abundant free meals for them. While enjoying the considerate service, the students also volunteer enthusiastically, contributing their modest efforts to the university's work. Kanade is no exception.

Kanade's job is to distribute boxed lunches to the students in n rooms numbered from 1 to n, which are continuously arranged from left to right on a corridor, with 4 students in each room. The school provides m different kinds of boxed meals, each of which is **sufficiently provided** (meaning they are infinite).

Kanade decided to distribute the boxed lunches to the rooms from left to right. Specifically, the i-th distribution is to randomly take out 4 boxed lunches from the boxed lunch chest and send it to the i-th room. A few days later however, Kanade found it exhausting to rush back and forth in the building and knock the door from room to room, so he came up with a more relaxing way.

The order is still left-to-right, but the i-th distribution is to randomly take $4k_i$ boxed lunches from the chest, and knock the doors of k_i **continuous** room(s) starting from the first room from left to right that hasn't been distributed. He will then put the $4k_i$ boxed lunches on the floor for the students from the k_i room(s) to choose. The process will continue until all the n rooms are distributed. Each k_i is a positive integer that Kanade can decide, but he doesn't want k_i to be too large to cause unnecessary confusion, so he stipulated that $k_i \leq K$.

Kanade wonders how many different plans there are to distribute the boxed lunches in such way. Two plans are considered different if and only if the times of distribution are different, k_i for the i-th distribution are different, or the number of any kind of boxed lunch taken out in i-th distribution are different.

Since this number may be too large, output it modulo 998244353.

Input

There is only one line, which contains three positive integer numbers n, m, K $(1 \leq K \leq n \leq 10^5, 1 \leq m \leq 10^5)$ — the number of rooms, the number of the types of boxed meals, and the upper bound of rooms that Kanade can distribute in each distribution.

Output

Output a single number — the number of different distributing plans modulo 998244353.

Examples

standard input	standard output
3 1 2	3
3 2 1	125
10 4 3	559774277

Note

Note that what Kanade wonders is the number of different **distributing plans**, meaning that he doesn't care how the students divide the $4k_i$ boxed lunches in the i-th distribution.

Problem E. Ethernet

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

Mr. Wor's router at home is broken again. After some repairs, he needs to reconnect the Ethernet cables to the router.

It is known that Mr. Wor's router has n ports, and there are n Ethernet cables. Then, Mr. Wor will sequentially insert the i-th Ethernet cable to the i-th port, for all i from 1 to n.

However, due to Mr. Wor's mysterious actions, the first m cables were not inserted into their designated ports, but were inserted into m ports uniformly at random. For the remaining cables, when cable i is being inserted, if port i is not occupied, then cable i will be inserted into port i. Otherwise, cable i will be inserted into a randomly chosen unoccupied port.

Mr. Wor wants to know the probability that the last cable, cable n, is plugged into port n.

Input

Only one line contains two integers $n, m (1 \le n \le 10, 0 \le m \le n)$, denoting the number of cables and the number of cables which was inserted randomly.

Output

Print the probability in one line. Your answer is considered correct if the relative or absolute error is less than or equal to 10^{-6}.

Example

standard input	standard output
3 1	0.5000000000

Problem F. Folder

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

There are n folders in a computer, indexed from 1 to n. The folder indexed 1 is the root folder, and each folder may contain several subfolders, forming a tree structure. You can perform several "cut" operations, each cutting a source folder into a target folder. Note that the target folder cannot be the same as the source folder, nor can it be the source folder's subfolder, subfolder's subfolder, and so on. Obviously, the root folder cannot be cut, and it will always be the root folder. Find out that at least how many "cut" operations should be performed so that each folder contains at most one subfolder.

Input

The first line contains a single integer n ($1 \le n \le 10^5$) — the number of folders. The second line of each test case contains $n-1$ space separated integers $p_2, p_3, \cdots, p_n$ ($1 \le p_i \le n$) — p_i represents the index of the parent folder of folder i.

Output

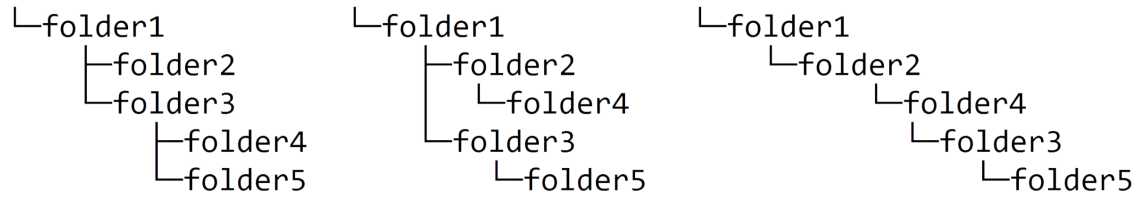
Output the minimum number of "cut" operations performed. Note that there may not be a need to perform "cut" operation. Please output 0 in this case.

Example

standard input	standard output
5 1 1 3 3	2

Note

Here is the explanation for the first sample.



(a) Initial state (b) Cut folder4 into folder2 (c) Cut folder3 into folder4

Problem G. Gravity

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

Joseph's task today is to collect asteroids to build new space stations.

The universe where Joseph lives is a 2D plane. The asteroids are located at n points denoted by $P_1, P_2, \cdots, P_n$. The mass of them can be considered equal. On the first day, he will collect some asteroids and bring them to their center of gravity. On the second day, he will do the same with the remaining asteroids. He will then build two new stations at the two gravity centers, denoted by points A and B.

Noted that building a space station requires at least one asteroid, so Joseph must collect **at least** one asteroid each day.

There is one original space station on point $O(0, 0)$. The coverage area of the three space stations is defined by the triangle ABO. Joseph wants to maximize this area as much as possible.

Please help him find an optimal collecting plan to get the maximum coverage area.

Input

The first line contains one integer n ($2 \le n \le 10^5$), indicating the number of asteroids.

In the next n lines, the i-th line contains two integers x_i, y_i ($-10^4 \le x_i, y_i \le 10^4$), denoting that there is one asteroid at point $P_i = (x_i, y_i)$

Output

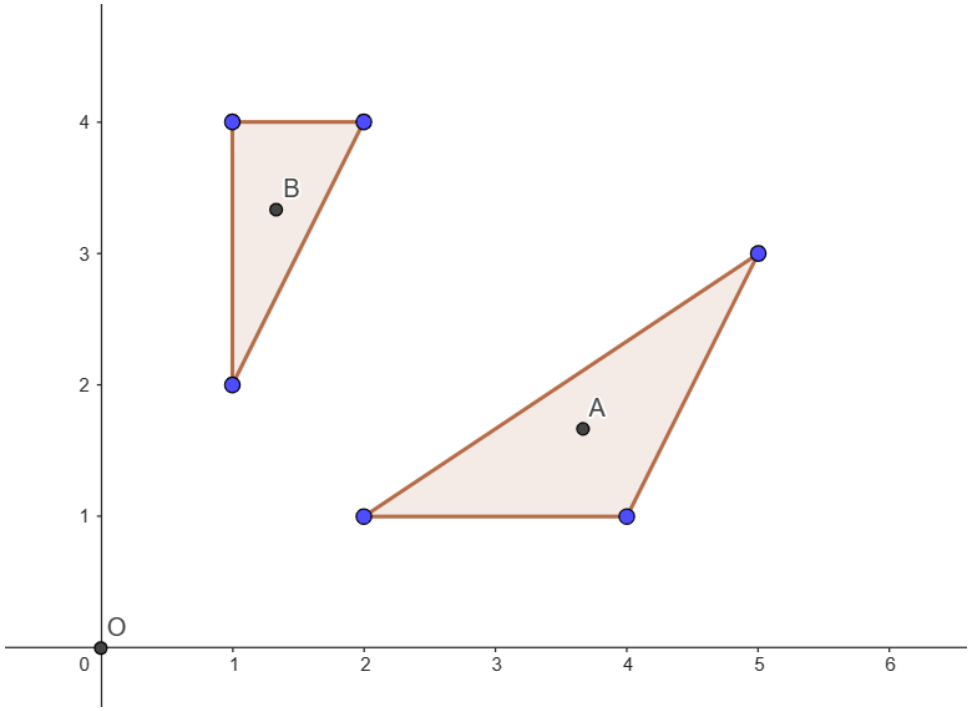
In one line print the maximum coverage area. Your answer is considered correct if the relative or absolute error is less than or equal to 10^{-9}.

Examples

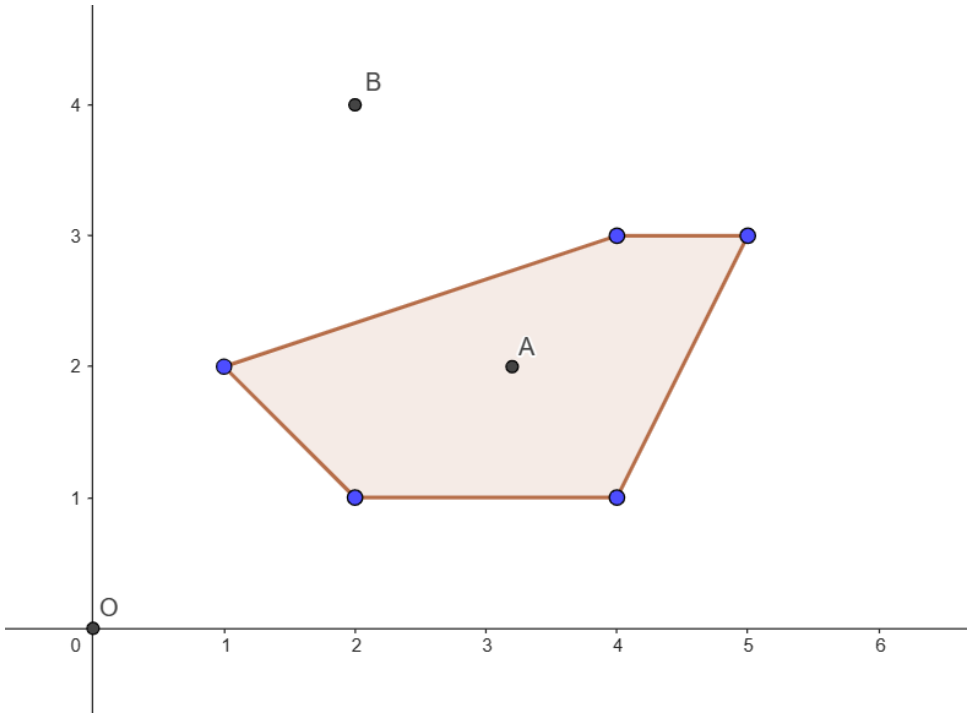
standard input	standard output
6 2 1 1 2 4 1 1 4 5 3 2 4	5.0000000000
6 2 1 1 2 4 1 4 3 5 3 2 4	4.4000000000

Note

The figure below describes the first sample. One optimal collecting plan is to collect the $1, 3, 5$-th asteroids on the first day and collect the $2, 4, 6$-th asteroid on the second day. The gravity centers are A and B respectively. The area of the triangle ABO is 5.0, which is maximized.



The figure below describes the second sample.

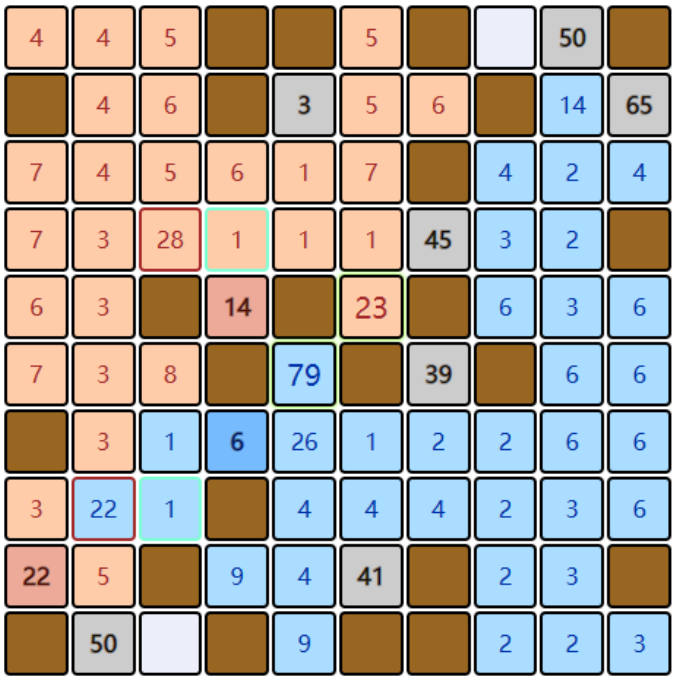


Problem H. KingZ

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

Mr. Wor is a senior player of the RTS games such as 'StarCraft'. However, in recently, Blizzard announced that they will exit the China market. As an enthusiast of Blizzard Entertainment, Wor felt frustrated deeply and had to find some substitution. In this situation, the game 'KingZ' attracted Wor's sight.

KingZ is a Table-Real-Time-Strategy game, which means two player play against and make decisions at the same time. Now we are giving you some **fundamental knowledge** about this game.



Basic Game Rules

As the picture illustrate, only two players (red and blue) play this game on a 10×10 size board. It is a turn base game, which means player do decision in each round. Each player occupied some fields where some garrisons on duty have, in other words, some troops on the occupied field are there. The garrison means the number of troops on the field. In each round of game, some reinforcements will reinforce the occupied field. Depending on various types of the fields, different number of reinforcements will be added into garrison. In this game, both players make decision and make strategy at the same time during each round of the game. Finally, who defeats the opposite core first will be the winner.

Map and Items

The map is composed of 10×10 blocks which we call 'field'. There are four types of each field: Core, Keep, Lawn and Wall. Each field has it own coordinate where the left-top corner is $(0, 0)$ and the right-bottom corner is $(10, 10)$. Noticing that each player has only one core.

Garrison and Occupation

Dictionary: Affiliation means belong to.

Every field can be occupied by garrison except the wall. initially, lawn has no garrison and is occupied by neither of players, Keep has neutral garrison, each player occupies only core with 10 garrison. Please be noted that this is just an initial situation that will be changed as the game continue.

Once some field be occupied by someone's garrison, the right of control of this field belongs to this player. A puzzling condition shows that sometimes a field has no garrison in a round, the affiliations of this field

will remain, which means the affiliation stays the same as the player who occupied this field in last round.

Battle

After the decision done, at the end of each round, if some field contains garrisons of different affiliations, they will battle.

The battle will follow the steps:

first, if there are garrisons affiliate both Blue and Red, they will battle first by doing subtraction of their number of garrisons. Then the stronger one who still has garrisons left will continue his battle.

Second, the remaining garrisons will battle with the neutral troops also by doing subtraction of their number of garrisons.

Finally, there are definitely at most one affiliation of garrison survives and occupy the field.

Formally speaking, we define the field has x garrisons affiliating Blue, y garrisons affiliating Red, z garrisons are neutral, now do following steps:

```
m=min(x,y)
x=x-m
y=y-m
if (x>0) then
begin
        m=min(x,z)
        x=x-m
        z=z-m
end
else if (y>0) then
begin
        m=min(y,z)
        y=y-m
        z=z-m
end
```

Reinforcement

After the end of each round, the number of garrisons in the core will increase by 2.

After the end of each round, if the keep is occupied by any player, the number of garrisons in the keep will increase by 1. Please be noted that those keep which are still neutral would not be reinforced.

After each $8k$ end of the round, the number of garrisons on the lawn where occupied by any players will increase by 1.

Decision(Operation)

Each decision, or we call it operation, can be translated into quintuple (u_x, u_y, x, v_x, v_y)**, which means the** x **troops of garrison on** (u_x, u_y) **will move directly to** (v_x, v_y) **without any cost.** You have to ensure that $x \leq$ **rounds number** $+ dis(u, v)$ **,** $dis(u, v) = |u_x - v_x| + |u_y - v_y|$**,** $0 \leq x \leq garrison_{u_x,u_y}$.

Victory or Defeat

The player who defeats the opposite core first will win.

Extra Limitation

PLEASE BE NOTED THAT THESE EXTRA LIMITATIONS IN THIS PROBLEM ARE SIGNIFICANT.

In this problem, unlimited times of operation can be done in a single round. And you have to ensure that after all operation done, the number of each garrisons in occupied field are greater than zero.

Although unlimited times of operation can be done, the quadruple (u_x, u_y, v_x, v_y) **will exist only once in operation list.**

Main Problem

Now, telling you the current situation, and no operations except reinforcement before you start operation will be done, which means before both red player and blue player start operating, both of two player do nothing, and we call this as waiting rounds. Your mission is to calculate the minimum waiting rounds, which by waiting these rounds then you can occupy all field except the wall in a single round by doing unlimited times of operation. To simplify the problem, we can easily consider that opposite does nothing throughout.

Please be noted that if the number of rounds exceed 300, then we judge the game as draw and output '-1'.

Input

Contain two 10×10 integer matrices $a_{i,j}, c_{i,j}$, first input is the integer matrix $a_{i,j}(-1 \leq a_{i,j} \leq 256)$ and second input is the integer matrix $c_{i,j}(c_{i,j} \in \{0, 1, 2, 3, 4, 5, 6, 7\})$.

Matrix $a_{i,j}$ meaning that:

$a_{i,j}$	**meaning**
-1	Wall
0	have no garrison
>0	be occupied and have $a_{i,j}$ troops of garrison

Matrix $c_{i,j}$ meaning that:

$c_{i,j}$	**meaning**
0	no affiliation
1	our core
2	our keep
3	our lawn
4	enemy core
5	enemy keep
6	enemy lawn
7	affiliate keep

Output

Only one integer *ans* represent that the minimum waiting rounds that by waiting these rounds then you can occupy all field expect the wall in a single round by doing infinity operation.

Noticing that if the round exceed 300, then we judge the game as draw and output '-1'.

Examples

standard input	standard output
7 5 -1 3 15 12 8 -1 47 0 4 2 -1 61 2 39 59 2 3 -1 2 11 3 1 -1 7 1 3 3 56 3 -1 1 74 -1 -1 2 3 3 8 0 -1 2 -1 7 6 2 6 60 7 2 60 2 4 5 6 -1 6 -1 6 2 9 2 4 -1 -1 74 6 -1 4 56 10 2 4 5 -1 3 16 0 2 -1 2 2 4 33 3 3 -1 0 2 0 47 -1 10 10 54 10 -1 0 3 3 6 0 3 2 3 3 0 7 0 3 3 0 6 3 1 3 3 3 0 3 3 5 6 0 3 3 3 3 7 3 0 6 7 0 0 3 3 3 3 0 0 6 0 3 3 3 3 7 3 6 7 6 6 3 3 0 3 0 3 6 6 6 6 0 0 7 3 0 3 7 6 6 6 6 0 6 5 0 3 0 6 6 6 4 6 6 0 0 3 0 7 0 6 6 7 6 0 0 3	-1
0 0 3 2 4 0 0 0 0 0 0 0 4 4 4 0 0 0 0 0 3 2 4 9 1 0 0 0 0 0 0 3 5 2 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 3 3 3 0 0 0 0 0 0 0 3 3 3 0 0 0 0 0 3 3 3 1 3 0 0 0 0 0 0 3 3 3 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	16

Problem I. Club

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	512 megabytes

In April, spring is in full swing, and the annual event — "Hundred Clubs Battle" will be held in JLU.

There are n clubs in JLU. During the day of the "Hundred Clubs Battle", each club will organize its own club game where participants can play to earn badges, and those who collect all types of badges will receive a mystery prize!

As a member of the JLU clubs association and organizer of the event, Paulliant designed m badges and assigned one of them to each club. Considering the participants do not know which type of badge each club has, they will randomly play in one club's game (which can be a previously selected club) at a time and receive its badge until all types of badges are collected. Paulliant wants to know how to assign the m badges to the n clubs in order to minimize the expected number of games played by participants. Output this expected value.

Note that the same badges can be assigned to different clubs, but each badge must be assigned to at least one club.

Input

There is only one line containing two positive integer numbers n, m ($1 \le n \le 10^5, 1 \le m \le 5000,\ m \le n$) — the number of clubs in JLU, and the number badges Paulliant designed, separated by a single space.

Output

Output a single number — the minimized expected number of games played by participants. Your answer will be considered correct if its absolute or relative error does not exceed 10^{-6}.

Examples

standard input	standard output
3 2	3.5000000000
111 7	18.1658637604

Problem J. XOR String

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

You are given a string S , consisting of n lowercase English letters $S_0, S_1, S_2, ..., S_{n-1}$.

Each character in the string S_i has a value V_i.

Define $U = \{x | x \in \mathbb{Z}, 0 \le x \le n-1\}$.

A string $t(|t| \le n)$ is said to be qualified if there exists a non-empty set P which meets the following requirements:

- $P \subseteq U$
- $\forall p \in P, \forall i \in [0, |t|-1], t_i = S_{(p+i) \mod n}$
- $\forall p \in \complement_U P, \exists i \in [0, |t|-1], t_i \neq S_{(p+i) \mod n}$
- $\bigoplus_{p \in P} V_p = 0$, which means the XOR sum of V_p equals to 0, for all elements p in P.

Please calculate the length of the longest qualified string t.

Input

The first line contains a string S of length n $(1 \le n \le 10^5)$, consisting of lowercase English letters. The second line contains n integers, $V_0, V_1, V_2, ..., V_{n-1}$ $(0 \le V_i < 2^{10})$.

Output

Output an integer, the length of the longest qualified string t. If there is no qualified string, output 0.

Example

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

Problem K. Turn-based Game

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

In a turn-based game, players control characters to defeat monsters. Now, you need to find an optimal strategy according to the rules.

In this game, the player holds a shield with A blood points. Each monster has B blood points, and a monster dies when the blood is less than or equal to 0. In each round, the player chooses a monster(any one of the monsters) and attacks it, then each monster alive will attack the player once. To simplify the problem, all attacks in the game will only drop the target's blood by 1 point.

Now the player will fight n battles. In the i-th battle, there are a_i monsters. Note that when all the monsters in a battle die, a new battle starts with a new round (which means the player attacks first in the new battle).

A character called Seele can help players kill monsters more efficiently. If the player kills a monster on an attack with Seele's help, he/she can immediately make another attack. For the i-th **attack** of each battle(for each battle, this count starts at 1), $b_i = 1$ means Seele will offer help, and $b_i = 0$ means she will not. It is guaranteed that when i is greater than m, $b_i = 0$.

If the remaining shields have less than or equal to 0 blood, the player loses. To avoid this condition, please find a strategy to maximize the blood of the remaining shield.

Input

The first line contains two integers A ($1 \le A \le 10^5$) and B ($1 \le B \le 20$), the blood points of the shield and monster.

The second line contains two integers n ($1 \le n \le 10^5$) and m ($1 \le m \le 2000$), the number of battles and the limit of Seele's help.

The third line contains n integers, and the i-th integer is a_i ($1 \le a_i \le 100$).

The fourth line contains m integers, and the i-th integer is b_i ($0 \le b_i \le 1, 0 \le \sum b_i \le 20$).

Output

If the player loses, output "LOSE". Otherwise, output an integer, the maximum blood of the remaining shield.

Examples

standard input	standard output
100 2 3 6 2 3 4 0 0 0 1 1 1	75
10 3 1 3 1 1 1 0	8

Problem L. Subxor

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

Given an array $a_1, a_2, \ldots, a_n$, you need to answer multiple queries raised by Mr. Wor, who is interested in understanding specific aspects of the array.

Each query consists of two integers l and r $(1 \le l \le r \le n)$. You need to find the longest contiguous subarray of $a_l, a_{l+1}, \ldots, a_r$, such that the XOR sum of its elements is less than or equal to a given value K.

Specifically, you need to find a pair u, v and maximize the value of $v - u + 1$, subject to the conditions as follows:

$$l \le u \le v \le r$$
$$a_u \oplus a_{u+1} \oplus \cdots \oplus a_v \le K$$

Note that the value of K is predetermined and remains constant for all queries. If there exists no valid pair u, v, then output 0.

Please write a program for Mr. Wor to answer all the queries correctly.

Input

The first line contains 2 integers n, K $(1 \le n \le 2 \times 10^4, 0 \le K < 2^{31})$.

The second line contains n integers $a_1, a_2, \ldots, a_n$ $(0 \le a_i < 2^{31})$, denoting the array.

The third line contains one integer q $(1 \le q \le 2 \times 10^4)$, denoting the number of queries.

In the next q lines, each line contains 2 integers l, r $(1 \le l \le r \le n)$, describing a query.

Output

For each query, print one integer in one line, denoting the maximum length of the subarray. Print 0 if such subarray doesn't exist.

Example

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