Codeforces Round #779 (Div. 2)

A. Marin and Photoshoot

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Today, Marin is at a cosplay exhibition and is preparing for a group photoshoot!

For the group picture, the cosplayers form a horizontal line. A group picture is considered *beautiful* if for every contiguous segment of at least 2 cosplayers, the number of males does not exceed the number of females (obviously).

Currently, the line has n cosplayers which can be described by a binary string s. The i-th cosplayer is male if $s_i = 0$ and female if $s_i = 1$. To ensure that the line is *beautiful*, you can invite some additional cosplayers (possibly zero) to join the line at any position. You can't remove any cosplayer from the line.

Marin wants to know the minimum number of cosplayers you need to invite so that the group picture of all the cosplayers is *beautiful*. She can't do this on her own, so she's asking you for help. Can you help her?

Input

The first line contains a single integer t ($1 \leq t \leq 10^3$) — the number of test cases.

The first line of each test case contains a positive integer n ($1 \leq n \leq 100$) — the number of cosplayers in the initial line.

The second line of each test case contains a binary string s of length n — describing the cosplayers already in line. Each character of the string is either `0` describing a male, or `1` describing a female.

Note that there is no limit on the sum of n.

Output

For each test case, print the minimum number of cosplayers you need to invite so that the group picture of all the cosplayers is *beautiful*.

Example

input
9 3 000 3 001 3 010 3 011 3 100 3 101 3 110 3 111 19 1010110000100000101
output
4 2 1 0 2 0 0 0 17

Note

In the first test case, for each pair of adjacent cosplayers, you can invite two female cosplayers to stand in between them. Then, $000 \rightarrow 0110110$.

In the third test case, you can invite one female cosplayer to stand next to the second cosplayer. Then, $010 \rightarrow 0110$.

B. Marin and Anti-coprime Permutation

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Marin wants you to count number of permutations that are *beautiful*. A *beautiful* permutation of length n is a permutation that has the following property:

$$\gcd(1 \cdot p_1,\ 2 \cdot p_2,\ \ldots,\ n \cdot p_n) > 1,$$

where $\gcd$ is the greatest common divisor.

A permutation is an array consisting of n distinct integers from 1 to n in arbitrary order. For example, $[2, 3, 1, 5, 4]$ is a permutation, but $[1, 2, 2]$ is not a permutation (2 appears twice in the array) and $[1, 3, 4]$ is also not a permutation ($n = 3$ but there is 4 in the array).

Input
The first line contains one integer t ($1 \le t \le 10^3$) — the number of test cases.

Each test case consists of one line containing one integer n ($1 \le n \le 10^3$).

Output
For each test case, print one integer — number of *beautiful* permutations. Because the answer can be very big, please print the answer modulo $998\,244\,353$.

Example

input
7 1 2 3 4 5 6 1000
output
0 1 0 4 0 36 665702330

Note
In first test case, we only have one permutation which is $[1]$ but it is not beautiful because $\gcd(1 \cdot 1) = 1$.

In second test case, we only have one beautiful permutation which is $[2, 1]$ because $\gcd(1 \cdot 2, 2 \cdot 1) = 2$.

C. Shinju and the Lost Permutation

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Shinju loves permutations very much! Today, she has borrowed a permutation p from Juju to play with.

The i-th cyclic shift of a permutation p is a transformation on the permutation such that $p = [p_1, p_2, \ldots, p_n]$ will now become $p = [p_{n-i+1}, \ldots, p_n, p_1, p_2, \ldots, p_{n-i}]$.

Let's define the *power* of permutation p as the number of distinct elements in the prefix maximums array b of the permutation. The prefix maximums array b is the array of length n such that $b_i = \max(p_1, p_2, \ldots, p_i)$. For example, the power of $[1, 2, 5, 4, 6, 3]$ is 4 since $b = [1, 2, 5, 5, 6, 6]$ and there are 4 distinct elements in b.

Unfortunately, Shinju has lost the permutation p! The only information she remembers is an array c, where c_i is the *power* of the $(i-1)$-th cyclic shift of the permutation p. She's also not confident that she remembers it correctly, so she wants to know if her memory is good enough.

Given the array c, determine if there exists a permutation p that is consistent with c. You do **not** have to construct the permutation p.

A permutation is an array consisting of n distinct integers from 1 to n in arbitrary order. For example, $[2, 3, 1, 5, 4]$ is a permutation, but $[1, 2, 2]$ is not a permutation (2 appears twice in the array) and $[1, 3, 4]$ is also not a permutation ($n = 3$ but there is 4 in the array).

Input

The input consists of multiple test cases. The first line contains a single integer t ($1 \le t \le 5 \cdot 10^3$) — the number of test cases.

The first line of each test case contains an integer n ($1 \le n \le 10^5$).

The second line of each test case contains n integers $c_1, c_2, \ldots, c_n$ ($1 \le c_i \le n$).

It is guaranteed that the sum of n over all test cases does not exceed 10^5.

Output

For each test case, print "YES" if there is a permutation p exists that satisfies the array c, and "NO" otherwise.

You can output "YES" and "NO" in any case (for example, strings "yEs", "yes", "Yes" and "YES" will be recognized as a positive response).

Example

input
6 1 1 2 1 2 2 2 2 6 1 2 4 6 3 5 6 2 3 1 2 3 4 3 3 2 1
output
YES YES NO NO YES NO

Note

In the first test case, the permutation $[1]$ satisfies the array c.

In the second test case, the permutation $[2, 1]$ satisfies the array c.

In the fifth test case, the permutation $[5, 1, 2, 4, 6, 3]$ satisfies the array c. Let's see why this is true.

- The zeroth cyclic shift of p is $[5, 1, 2, 4, 6, 3]$. Its power is 2 since $b = [5, 5, 5, 5, 6, 6]$ and there are 2 distinct elements — 5 and 6.
- The first cyclic shift of p is $[3, 5, 1, 2, 4, 6]$. Its power is 3 since $b = [3, 5, 5, 5, 5, 6]$.
- The second cyclic shift of p is $[6, 3, 5, 1, 2, 4]$. Its power is 1 since $b = [6, 6, 6, 6, 6, 6]$.
- The third cyclic shift of p is $[4, 6, 3, 5, 1, 2]$. Its power is 2 since $b = [4, 6, 6, 6, 6, 6]$.
- The fourth cyclic shift of p is $[2, 4, 6, 3, 5, 1]$. Its power is 3 since $b = [2, 4, 6, 6, 6, 6]$.
- The fifth cyclic shift of p is $[1, 2, 4, 6, 3, 5]$. Its power is 4 since $b = [1, 2, 4, 6, 6, 6]$.

Therefore, $c = [2, 3, 1, 2, 3, 4]$.

In the third, fourth, and sixth testcases, we can show that there is no permutation that satisfies array c.

D1. 388535 (Easy Version)

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

This is the easy version of the problem. The difference in the constraints between both versions is colored below in red. You can make hacks only if all versions of the problem are solved.

Marin and Gojou are playing hide-and-seek with an array.

Gojou initially performs the following steps:

- First, Gojou chooses 2 integers l and r such that $l \le r$.
- Then, Gojou makes an array a of length $r - l + 1$ which is a permutation of the array $[l, l + 1, \ldots, r]$.
- Finally, Gojou chooses a secret integer x and sets a_i to $a_i \oplus x$ for all i (where $\oplus$ denotes the bitwise XOR operation).

Marin is then given the values of l, r and the final array a. She needs to find the secret integer x to win. Can you help her?

Note that there may be multiple possible x that Gojou could have chosen. Marin can find any possible x that could have resulted in the final value of a.

Input
The first line contains a single integer t ($1 \leq t \leq 10^5$) — the number of test cases.

In the first line of each test case contains two integers l and r ($\mathbf{0 = l} \leq r < 2^{17}$).

The second line contains $r - l + 1$ integers of $a_1, a_2, \ldots, a_{r-l+1}$ ($0 \leq a_i < 2^{17}$). It is guaranteed that a can be generated using the steps performed by Gojou.

It is guaranteed that the sum of $r - l + 1$ over all test cases does not exceed 2^{17}.

Output
For each test case print an integer x. If there are multiple answers, print any.

Example

input
3 0 3 3 2 1 0 0 3 4 7 6 5 0 2 1 2 3
output
0 4 3

Note
In the first test case, the original array is $[3, 2, 1, 0]$.

In the second test case, the original array is $[0, 3, 2, 1]$.

In the third test case, the original array is $[2, 1, 0]$.

D2. 388535 (Hard Version)

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

This is the hard version of the problem. The difference in the constraints between both versions are colored below in red. You can make hacks only if all versions of the problem are solved.

Marin and Gojou are playing hide-and-seek with an array.

Gojou initially perform the following steps:

- First, Gojou chooses 2 integers l and r such that $l \leq r$.
- Then, Gojou will make an array a of length $r - l + 1$ which is a permutation of the array $[l, l + 1, \ldots, r]$.
- Finally, Gojou chooses a secret integer x and sets a_i to $a_i \oplus x$ for all i (where $\oplus$ denotes the bitwise XOR operation).

Marin is then given the values of l, r and the final array a. She needs to find the secret integer x to win. Can you help her?

Note that there may be multiple possible x that Gojou could have chosen. Marin can find any possible x that could have resulted in the final value of a.

Input
The first line contains a single integer t ($1 \leq t \leq 10^5$) — the number of test cases.

In the first line of each test case contains two integers l and r ($\mathbf{0 \leq l} \leq r < 2^{17}$).

The second line contains $r - l + 1$ space-seperated integers of $a_1, a_2, \ldots, a_{r-l+1}$ ($0 \leq a_i < 2^{17}$). It is guaranteed that array a is valid.

It is guaranteed that the sum of $r - l + 1$ over all test cases does not exceed 2^{17}.

Output
For each test case print an integer x. If there are multiple answers, print any.

Example

input
3 4 7 3 2 1 0 4 7 4 7 6 5

1 3 0 2 1
output
4 0 3

Note

In the first test case, the original array is $[7, 6, 5, 4]$.

In the second test case, the original array is $[4, 7, 6, 5]$.

In the third test case, the original array is $[3, 1, 2]$.

E. Gojou and Matrix Game

time limit per test: 4 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Marin feels exhausted after a long day of cosplay, so Gojou invites her to play a game!

Marin and Gojou take turns to place one of their tokens on an $n \times n$ grid with Marin starting first. There are some restrictions and allowances on where to place tokens:

- Apart from the first move, the token placed by a player must be more than Manhattan distance k away from the previous token placed on the matrix. In other words, if a player places a token at (x_1, y_1), then the token placed **by the other player** in the next move must be in a cell (x_2, y_2) satisfying $|x_2 - x_1| + |y_2 - y_1| > k$.
- Apart from the previous restriction, a token can be placed anywhere on the matrix, **including cells where tokens were previously placed by any player**.

Whenever a player places a token on cell (x, y), that player gets $v_{x, y}$ points. All values of v on the grid are **distinct**. You still get points from a cell even if tokens were already placed onto the cell. The game finishes when each player makes 10^{100} moves.

Marin and Gojou will play n^2 games. For each cell of the grid, there will be exactly one game where Marin places a token on that cell on her first move. Please answer for each game, if Marin and Gojou play optimally (after Marin's first move), who will have more points at the end? Or will the game end in a draw (both players have the same points at the end)?

Input

The first line contains two integers n, k ($3 \le n \le 2000$, $1 \le k \le n - 2$). Note that under these constraints it is always possible to make a move.

The following n lines contains n integers each. The j-th integer in the i-th line is $v_{i,j}$ ($1 \le v_{i,j} \le n^2$). All elements in v are distinct.

Output

You should print n lines. In the i-th line, print n characters, where the j-th character is the result of the game in which Marin places her first token in the cell (i, j). Print 'M' if Marin wins, 'G' if Gojou wins, and 'D' if the game ends in a draw. Do not print spaces between the characters in one line.

Example

3 1 1 2 4 6 8 3 9 5 7
output
GGG MGG MGG

F. Juju and Binary String

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

The *cuteness* of a binary string is the number of 1s divided by the length of the string. For example, the *cuteness* of 01101 is $\frac{3}{5}$.

Juju has a binary string s of length n. She wants to choose some non-intersecting subsegments of s such that their concatenation has length m and it has the same *cuteness* as the string s.

More specifically, she wants to find two arrays l and r of equal length k such that $1 \le l_1 \le r_1 < l_2 \le r_2 < \ldots < l_k \le r_k \le n$, and

also:

- $\sum_{i=1}^{k}(r_i - l_i + 1) = m$;
- The *cuteness* of $s[l_1, r_1] + s[l_2, r_2] + \ldots + s[l_k, r_k]$ is equal to the *cuteness* of s, where $s[x, y]$ denotes the subsegment $s_x s_{x+1} \ldots s_y$, and $+$ denotes string concatenation.

Juju does not like splitting the string into many parts, so she also wants to **minimize** the value of k. Find the minimum value of k such that there exist l and r that satisfy the constraints above or determine that it is impossible to find such l and r for any k.

Input

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

The first line of each test case contains two integers n and m ($1 \leq m \leq n \leq 2 \cdot 10^5$).

The second line of each test case contains a binary string s of length n.

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

Output

For each test case, if there is no valid pair of l and r, print -1.

Otherwise, print $k + 1$ lines.

In the first line, print a number k ($1 \leq k \leq m$) — the minimum number of subsegments required.

Then print k lines, the i-th should contain l_i and r_i ($1 \leq l_i \leq r_i \leq n$) — the range of the i-th subsegment. Note that you should output the subsegments such that the inequality $l_1 \leq r_1 < l_2 \leq r_2 < \ldots < l_k \leq r_k$ is true.

Example

input
4 4 2 0011 8 6 11000011 4 3 0101 5 5 11111
output
1 2 3 2 2 3 5 8 -1 1 1 5

Note

In the first example, the *cuteness* of 0011 is the same as the *cuteness* of 01.

In the second example, the *cuteness* of 11000011 is $\frac{1}{2}$ and there is no subsegment of size 6 with the same cuteness. So we must use 2 disjoint subsegments 10 and 0011.

In the third example, there are 8 ways to split the string such that $\sum_{i=1}^{k}(r_i - l_i + 1) = 3$ but none of them has the same *cuteness* as 0101.

In the last example, we don't have to split the string.