

Codeforces Round #692 (Div. 2, based on Technocup 2021 Elimination Round 3)

A. In-game Chat

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

You have been assigned to develop a filter for bad messages in the in-game chat. A message is a string S of length n , consisting of lowercase English letters and characters ')'. The message is *bad* if the number of characters ')' at the end of the string strictly greater than the number of remaining characters. For example, the string ")bc)))" has three parentheses at the end, three remaining characters, and is not considered bad.

Input

The first line contains the number of test cases t ($1 \leq t \leq 100$). Description of the t test cases follows.

The first line of each test case contains an integer n ($1 \leq n \leq 100$). The second line of each test case contains a string S of length n , consisting of lowercase English letters and characters ')'.
)

Output

For each of t test cases, print "Yes" if the string is bad. Otherwise, print "No".

You can print each letter in any case (upper or lower).

Example

input
5 2)) 12 g!))h f)))) 9 gege)))) 14)aa))b)))))) 1)
output
Yes No Yes Yes Yes

B. Fair Numbers

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

We call a positive integer number *fair* if it is divisible by each of its nonzero digits. For example, 102 is fair (because it is divisible by 1 and 2), but 282 is not, because it isn't divisible by 8. Given a positive integer n . Find the minimum integer x , such that $n \leq x$ and x is fair.

Input

The first line contains number of test cases t ($1 \leq t \leq 10^3$). Each of the next t lines contains an integer n ($1 \leq n \leq 10^{18}$).

Output

For each of t test cases print a single integer — the least *fair* number, which is not less than n .

Example

input
4 1 282 1234567890 1000000000000000000
output
1

288 1234568040 10000000000000000000

Note

Explanations for some test cases:

- In the first test case number 1 is fair itself.
- In the second test case number 288 is fair (it's divisible by both 2 and 8). None of the numbers from [282, 287] is fair, because, for example, none of them is divisible by 8.

C. Peaceful Rooks

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

You are given a $n \times n$ chessboard. Rows and columns of the board are numbered from 1 to n . Cell (x, y) lies on the intersection of column number x and row number y .

Rook is a chess piece, that can in one turn move any number of cells vertically or horizontally. There are m rooks ($m < n$) placed on the chessboard in such a way that no pair of rooks attack each other. I.e. there are no pair of rooks that share a row or a column.

In one turn you can move one of the rooks any number of cells vertically or horizontally. Additionally, it shouldn't be attacked by any other rook after movement. What is the minimum number of moves required to place all the rooks on the main diagonal?

The main diagonal of the chessboard is all the cells (i, i) , where $1 \leq i \leq n$.

Input

The first line contains the number of test cases t ($1 \leq t \leq 10^3$). Description of the t test cases follows.

The first line of each test case contains two integers n and m — size of the chessboard and the number of rooks ($2 \leq n \leq 10^5$, $1 \leq m < n$). Each of the next m lines contains two integers x_i and y_i — positions of rooks, i -th rook is placed in the cell (x_i, y_i) ($1 \leq x_i, y_i \leq n$). It's guaranteed that no two rooks attack each other in the initial placement.

The sum of n over all test cases does not exceed 10^5 .

Output

For each of t test cases print a single integer — the minimum number of moves required to place all the rooks on the main diagonal.

It can be proved that this is always possible.

Example

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

Note

Possible moves for the first three test cases:

1. $(2, 3) \rightarrow (2, 2)$
2. $(2, 1) \rightarrow (2, 3)$, $(1, 2) \rightarrow (1, 1)$, $(2, 3) \rightarrow (2, 2)$
3. $(2, 3) \rightarrow (2, 4)$, $(2, 4) \rightarrow (4, 4)$, $(3, 1) \rightarrow (3, 3)$, $(1, 2) \rightarrow (1, 1)$

D. Grime Zoo

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

output: standard output

Currently, XXOC's rap is a string consisting of zeroes, ones, and question marks. Unfortunately, haters gonna hate. They will write x angry comments for every occurrence of **subsequence** 01 and y angry comments for every occurrence of **subsequence** 10. You should replace all the question marks with 0 or 1 in such a way that the number of angry comments would be as small as possible.

String b is a subsequence of string a , if it can be obtained by removing some characters from a . Two occurrences of a subsequence are considered distinct if sets of positions of remaining characters are distinct.

Input

The first line contains string s — XXOC's rap ($1 \leq |s| \leq 10^5$). The second line contains two integers x and y — the number of angry comments XXOC will receive for every occurrence of 01 and 10 accordingly ($0 \leq x, y \leq 10^6$).

Output

Output a single integer — the minimum number of angry comments.

Examples

input
0?1 2 3
output
4
input
???? 13 37
output
0
input
?10? 239 7
output
28
input
01101001 5 7
output
96

Note

In the first example one of the optimum ways to replace is 001. Then there will be 2 subsequences 01 and 0 subsequences 10. Total number of angry comments will be equal to $2 \cdot 2 + 0 \cdot 3 = 4$.

In the second example one of the optimum ways to replace is 11111. Then there will be 0 subsequences 01 and 0 subsequences 10. Total number of angry comments will be equal to $0 \cdot 13 + 0 \cdot 37 = 0$.

In the third example one of the optimum ways to replace is 1100. Then there will be 0 subsequences 01 and 4 subsequences 10. Total number of angry comments will be equal to $0 \cdot 239 + 4 \cdot 7 = 28$.

In the fourth example one of the optimum ways to replace is 01101001. Then there will be 8 subsequences 01 and 8 subsequences 10. Total number of angry comments will be equal to $8 \cdot 5 + 8 \cdot 7 = 96$.

E. Poman Numbers

time limit per test: 1 second

memory limit per test: 256 megabytes

input: standard input

output: standard output

You've got a string S consisting of n lowercase English letters from your friend. It turned out that this is a number written in *poman* numerals. The poman numeral system is long forgotten. All that's left is the algorithm to transform number from poman numerals to the numeral system familiar to us. Characters of S are numbered from 1 to n from left to right. Let's denote the value of S as $f(S)$, it is defined as follows:

- If $|S| > 1$, an arbitrary integer m ($1 \leq m < |S|$) is chosen, and it is defined that $f(S) = -f(S[1, m]) + f(S[m + 1, |S|])$, where $S[l, r]$ denotes the substring of S from the l -th to the r -th position, inclusively.
- Otherwise $S = c$, where c is some English letter. Then $f(S) = 2^{\text{pos}(c)}$, where $\text{pos}(c)$ is the position of letter c in the alphabet ($\text{pos}(a) = 0, \text{pos}(z) = 25$).

Note that m is chosen independently on each step.

Your friend thinks it is possible to get $f(S) = T$ by choosing the right m on every step. Is he right?

Input

The first line contains two integers n and T ($2 \leq n \leq 10^5, -10^{15} \leq T \leq 10^{15}$).

The second line contains a string S consisting of n lowercase English letters.

Output

Print "Yes" if it is possible to get the desired value. Otherwise, print "No".

You can print each letter in any case (upper or lower).

Examples

input
2 -1 ba
output
Yes

input
3 -7 abc
output
No

input
7 -475391 qohshra
output
Yes

Note

In the second example, you cannot get -7 . But you can get 1 , for example, as follows:

- 1. First choose $m = 1$, then $f(abc) = -f(a) + f(bc)$
- 2. $f(a) = 2^0 = 1$
- 3. $f(bc) = -f(b) + f(c) = -2^1 + 2^2 = 2$
- 4. In the end $f(abc) = -1 + 2 = 1$

F. The Thorny Path

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

According to a legend the Hanoi Temple holds a permutation of integers from 1 to n . There are n stones of distinct colors lying in one line in front of the temple. Monks can perform the following operation on stones: choose a position i ($1 \leq i \leq n$) and cyclically shift stones at positions $i, p[i], p[p[i]], \dots$. That is, a stone from position i will move to position $p[i]$, a stone from position $p[i]$ will move to position $p[p[i]]$, and so on, a stone from position j , such that $p[j] = i$, will move to position i .

Each day the monks must obtain a new arrangement of stones using an arbitrary number of these operations. When all possible arrangements will have been obtained, the world will end. You are wondering, what if some elements of the permutation could be swapped just before the beginning? How many days would the world last?

You want to get a permutation that will allow the world to last as long as possible, using the minimum number of exchanges of two elements of the permutation.

Two arrangements of stones are considered different if there exists a position i such that the colors of the stones on that position are different in these arrangements.

Input

Each test consists of multiple test cases. The first line contains the number of test cases t ($1 \leq t \leq 10^3$). Description of the test cases follows.

The first line of each test case contains n ($3 \leq n \leq 10^6$). The next line contains n integers p_1, \dots, p_n ($1 \leq p_i \leq n$). It is guaranteed that p is a permutation.

It is guaranteed that the sum of n over all test cases does not exceed 10^6 .

Output

For each of the t test cases, print two integers on a new line: the largest possible number of days the world can last, modulo $10^9 + 7$, and the minimum number of exchanges required for that.

Examples

input
3 3 2 3 1 3 2 1 3 3 1 2 3
output
3 0 3 1 3 2

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

Note

Let's label the colors of the stones with letters. Explanations for the first two test cases of the first example:

1. Using the permutation $[2, 3, 1]$, we can additionally obtain the arrangements CAB and BCA from ABC. This is already the maximum possible result.
2. Using the permutation $[2, 1, 3]$, the only BAC can be obtained from ABC. As we saw in the previous case, two arrangements are not the maximum possible number of distinct arrangements for $n = 3$. To get an optimal permutation, for example, we can swap 1 and 3, so we will get the permutation $[2, 3, 1]$.