Codeforces Round #582 (Div. 3)

A. Chips Moving

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

You are given n chips on a number line. The i-th chip is placed at the integer coordinate x_i. Some chips **can have equal coordinates**.

You can perform each of the two following types of moves any (possibly, zero) number of times on any chip:

- Move the chip i by 2 to the left or 2 to the right **for free** (i.e. replace the current coordinate x_i with $x_i - 2$ or with $x_i + 2$);
- move the chip i by 1 to the left or 1 to the right and pay **one coin** for this move (i.e. replace the current coordinate x_i with $x_i - 1$ or with $x_i + 1$).

Note that it's allowed to move chips to any integer coordinate, including negative and zero.

Your task is to find the minimum total number of coins required to move all n chips to the same coordinate (i.e. all x_i should be equal after some sequence of moves).

Input

The first line of the input contains one integer n ($1 \le n \le 100$) — the number of chips.

The second line of the input contains n integers $x_1, x_2, \ldots, x_n$ ($1 \le x_i \le 10^9$), where x_i is the coordinate of the i-th chip.

Output

Print one integer — the minimum total number of coins required to move all n chips to the same coordinate.

Examples

input
3 1 2 3
output
1

input
5 2 2 2 3 3
output
2

Note

In the first example you need to move the first chip by 2 to the right and the second chip by 1 to the right or move the third chip by 2 to the left and the second chip by 1 to the left so the answer is 1.

In the second example you need to move two chips with coordinate 3 by 1 to the left so the answer is 2.

B. Bad Prices

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

Polycarp analyzes the prices of the new berPhone. At his disposal are the prices for n last days: $a_1, a_2, \ldots, a_n$, where a_i is the price of berPhone on the day i.

Polycarp considers the price on the day i to be bad if later (that is, a day with a greater number) berPhone was sold at a lower price. For example, if $n = 6$ and $a = [3, 9, 4, 6, 7, 5]$, then the number of days with a bad price is 3 — these are days 2 ($a_2 = 9$), 4 ($a_4 = 6$) and 5 ($a_5 = 7$).

Print the number of days with a bad price.

You have to answer t independent data sets.

Input

The first line contains an integer t ($1 \le t \le 10000$) — the number of sets of input data in the test. Input data sets must be processed independently, one after another.

Each input data set consists of two lines. The first line contains an integer n ($1 \le n \le 150000$) — the number of days. The second line contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^6$), where a_i is the price on the i-th day.

It is guaranteed that the sum of n over all data sets in the test does not exceed 150000.

Output
Print t integers, the j-th of which should be equal to the number of days with a bad price in the j-th input data set.

Example

input
5 6 3 9 4 6 7 5 1 1000000 2 2 1 10 31 41 59 26 53 58 97 93 23 84 7 3 2 1 2 3 4 5
output
3 0 1 8 2

C. Book Reading

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

Polycarp is reading a book consisting of n pages numbered from 1 to n. Every time he finishes the page with the number divisible by m, he writes down the last digit of this page number. For example, if $n = 15$ and $m = 5$, pages divisible by m are $5, 10, 15$. Their last digits are $5, 0, 5$ correspondingly, their sum is 10.

Your task is to calculate the sum of all digits Polycarp has written down.

You have to answer q independent queries.

Input
The first line of the input contains one integer q ($1 \le q \le 1000$) — the number of queries.

The following q lines contain queries, one per line. Each query is given as two integers n and m ($1 \le n, m \le 10^{16}$) — the number of pages in the book and required divisor, respectively.

Output
For each query print the answer for it — the sum of digits written down by Polycarp.

Example

input
7 1 1 10 1 100 3 1024 14 998244353 1337 123 144 1234312817382646 13
output
1 45 153 294 3359835 0 427262129093995

D1. Equalizing by Division (easy version)

time limit per test: 2 seconds
memory limit per test: 256 megabytes

input: standard input
output: standard output

The only difference between easy and hard versions is the number of elements in the array.

You are given an array a consisting of n integers. In one move you can choose any a_i and divide it by 2 rounding down (in other words, in one move you can set $a_i := \lfloor \frac{a_i}{2} \rfloor$).

You can perform such an operation **any** (possibly, zero) number of times with **any** a_i.

Your task is to calculate the minimum possible number of operations required to obtain at least k equal numbers in the array.

Don't forget that it is possible to have $a_i = 0$ after some operations, thus the answer always exists.

Input
The first line of the input contains two integers n and k ($1 \le k \le n \le 50$) — the number of elements in the array and the number of equal numbers required.

The second line of the input contains n integers $a_1, a_2, \dots, a_n$ ($1 \le a_i \le 2 \cdot 10^5$), where a_i is the i-th element of a.

Output
Print one integer — the minimum possible number of operations required to obtain at least k equal numbers in the array.

Examples

input
5 3 1 2 2 4 5
output
1

input
5 3 1 2 3 4 5
output
2

input
5 3 1 2 3 3 3
output
0

D2. Equalizing by Division (hard version)

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

The only difference between easy and hard versions is the number of elements in the array.

You are given an array a consisting of n integers. In one move you can choose any a_i and divide it by 2 rounding down (in other words, in one move you can set $a_i := \lfloor \frac{a_i}{2} \rfloor$).

You can perform such an operation **any** (possibly, zero) number of times with **any** a_i.

Your task is to calculate the minimum possible number of operations required to obtain at least k equal numbers in the array.

Don't forget that it is possible to have $a_i = 0$ after some operations, thus the answer always exists.

Input
The first line of the input contains two integers n and k ($1 \le k \le n \le 2 \cdot 10^5$) — the number of elements in the array and the number of equal numbers required.

The second line of the input contains n integers $a_1, a_2, \dots, a_n$ ($1 \le a_i \le 2 \cdot 10^5$), where a_i is the i-th element of a.

Output
Print one integer — the minimum possible number of operations required to obtain at least k equal numbers in the array.

Examples

input
5 3 1 2 2 4 5

output
1

input
5 3 1 2 3 4 5
output
2

input
5 3 1 2 3 3 3
output
0

E. Two Small Strings

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

You are given two strings s and t **both of length** 2 and both consisting only of characters 'a', 'b' and 'c'.

Possible examples of strings s and t: "ab", "ca", "bb".

You have to find a string res consisting of $3n$ characters, n characters should be 'a', n characters should be 'b' and n characters should be 'c' and s and t should not occur in res as substrings.

A substring of a string is a contiguous subsequence of that string. So, the strings "ab", "ac" and "cc" are substrings of the string "abacc", but the strings "bc", "aa" and "cb" are not substrings of the string "abacc".

If there are multiple answers, you can print any of them.

Input

The first line of the input contains one integer n ($1 \le n \le 10^5$) — the number of characters 'a', 'b' and 'c' in the resulting string.

The second line of the input contains one string s of length 2 consisting of characters 'a', 'b' and 'c'.

The third line of the input contains one string t of length 2 consisting of characters 'a', 'b' and 'c'.

Output

If it is impossible to find the suitable string, print "NO" on the first line.

Otherwise print "YES" on the first line and string res on the second line. res should consist of $3n$ characters, n characters should be 'a', n characters should be 'b' and n characters should be 'c' and s and t should not occur in res as substrings.

If there are multiple answers, you can print any of them.

Examples

input
2 ab bc
output
YES acbbac

input
3 aa bc
output
YES cacbacbab

input
1 cb ac
output

YES
abc

F. Unstable String Sort

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

Authors have come up with the string s consisting of n lowercase Latin letters.

You are given two permutations of its indices (not necessary equal) p and q (both of length n). Recall that the permutation is the array of length n which contains each integer from 1 to n exactly once.

For all i from 1 to $n-1$ the following properties hold: $s[p_i] \le s[p_{i+1}]$ and $s[q_i] \le s[q_{i+1}]$. It means that if you will write down all characters of s in order of permutation indices, the resulting string will be sorted in the non-decreasing order.

Your task is to restore **any** such string s of length n consisting of **at least** k **distinct lowercase Latin letters** which suits the given permutations.

If there are multiple answers, you can print any of them.

Input

The first line of the input contains two integers n and k ($1 \le n \le 2 \cdot 10^5, 1 \le k \le 26$) — the length of the string and the number of distinct characters required.

The second line of the input contains n integers $p_1, p_2, \ldots, p_n$ ($1 \le p_i \le n$, all p_i are distinct integers from 1 to n) — the permutation p.

The third line of the input contains n integers $q_1, q_2, \ldots, q_n$ ($1 \le q_i \le n$, all q_i are distinct integers from 1 to n) — the permutation q.

Output

If it is impossible to find the suitable string, print "NO" on the first line.

Otherwise print "YES" on the first line and string s on the second line. It should consist of n lowercase Latin letters, contain at least k distinct characters and suit the given permutations.

If there are multiple answers, you can print any of them.

Example

input
3 2 1 2 3 1 3 2
output
YES abb

G. Path Queries

time limit per test: 3 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

You are given a weighted tree consisting of n vertices. Recall that a tree is a connected graph without cycles. Vertices u_i and v_i are connected by an edge with weight w_i.

You are given m queries. The i-th query is given as an integer q_i. In this query you need to calculate the number of pairs of vertices (u, v) ($u < v$) such that the maximum weight of an edge on a simple path between u and v doesn't exceed q_i.

Input

The first line of the input contains two integers n and m ($1 \le n, m \le 2 \cdot 10^5$) — the number of vertices in the tree and the number of queries.

Each of the next $n-1$ lines describes an edge of the tree. Edge i is denoted by three integers u_i, v_i and w_i — the labels of vertices it connects ($1 \le u_i, v_i \le n$, $u_i \ne v_i$) and the weight of the edge ($1 \le w_i \le 2 \cdot 10^5$). It is guaranteed that the given edges form a tree.

The last line of the input contains m integers $q_1, q_2, \ldots, q_m$ ($1 \le q_i \le 2 \cdot 10^5$), where q_i is the maximum weight of an edge in the i-th query.

Output

Print m integers — the answers to the queries. The i-th value should be equal to the number of pairs of vertices (u, v) ($u < v$) such

that the maximum weight of an edge on a simple path between u and v doesn't exceed q_i.

Queries are numbered from 1 to m in the order of the input.

Examples

input
7 5 1 2 1 3 2 3 2 4 1 4 5 2 5 7 4 3 6 2 5 2 3 4 1
output
21 7 15 21 3

input
1 2 1 2
output
0 0

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

Note

The picture shows the tree from the first example: