Codeforces Round #735 (Div. 2)

A. Cherry

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

You are given n integers $a_1, a_2, \ldots, a_n$. Find the maximum value of $max(a_l, a_{l+1}, \ldots, a_r) \cdot min(a_l, a_{l+1}, \ldots, a_r)$ over all pairs (l, r) of integers for which $1 \le l < r \le n$.

Input

The first line contains a single integer t ($1 \le t \le 10\,000$) — the number of test cases.

The first line of each test case contains a single integer n ($2 \le n \le 10^5$).

The second line of each test case contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^6$).

It is guaranteed that the sum of n over all test cases doesn't exceed $3 \cdot 10^5$.

Output

For each test case, print a single integer — the maximum possible value of the product from the statement.

Example

input
4 3 2 4 3 4 3 2 3 1 2 69 69 6 719313 273225 402638 473783 804745 323328
output
12 6 4761 381274500335

Note

Let $f(l, r) = max(a_l, a_{l+1}, \ldots, a_r) \cdot min(a_l, a_{l+1}, \ldots, a_r)$.

In the first test case,

- $f(1, 2) = max(a_1, a_2) \cdot min(a_1, a_2) = max(2, 4) \cdot min(2, 4) = 4 \cdot 2 = 8$.
- $f(1, 3) = max(a_1, a_2, a_3) \cdot min(a_1, a_2, a_3) = max(2, 4, 3) \cdot min(2, 4, 3) = 4 \cdot 2 = 8$.
- $f(2, 3) = max(a_2, a_3) \cdot min(a_2, a_3) = max(4, 3) \cdot min(4, 3) = 4 \cdot 3 = 12$.

So the maximum is $f(2, 3) = 12$.

In the second test case, the maximum is $f(1, 2) = f(1, 3) = f(2, 3) = 6$.

B. Cobb

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

You are given n integers $a_1, a_2, \ldots, a_n$ and an integer k. Find the maximum value of $i \cdot j - k \cdot (a_i | a_j)$ over all pairs (i, j) of integers with $1 \le i < j \le n$. Here, $|$ is the bitwise OR operator.

Input

The first line contains a single integer t ($1 \le t \le 10\,000$) — the number of test cases.

The first line of each test case contains two integers n ($2 \le n \le 10^5$) and k ($1 \le k \le \min(n, 100)$).

The second line of each test case contains n integers $a_1, a_2, \ldots, a_n$ ($0 \le a_i \le n$).

It is guaranteed that the sum of n over all test cases doesn't exceed $3 \cdot 10^5$.

Output
For each test case, print a single integer — the maximum possible value of $i \cdot j - k \cdot (a_i|a_j)$.

Example

input
4 3 3 1 1 3 2 2 1 2 4 3 0 1 2 3 6 6 3 2 0 0 5 6
output
-1 -4 3 12

Note
Let $f(i, j) = i \cdot j - k \cdot (a_i|a_j)$.

In the first test case,

- $f(1, 2) = 1 \cdot 2 - k \cdot (a_1|a_2) = 2 - 3 \cdot (1|1) = -1$.
- $f(1, 3) = 1 \cdot 3 - k \cdot (a_1|a_3) = 3 - 3 \cdot (1|3) = -6$.
- $f(2, 3) = 2 \cdot 3 - k \cdot (a_2|a_3) = 6 - 3 \cdot (1|3) = -3$.

So the maximum is $f(1, 2) = -1$.

In the fourth test case, the maximum is $f(3, 4) = 12$.

C. Mikasa

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

You are given two integers n and m. Find the MEX of the sequence $n \oplus 0, n \oplus 1, \ldots, n \oplus m$. Here, $\oplus$ is the bitwise XOR operator.

MEX of the sequence of non-negative integers is the smallest non-negative integer that doesn't appear in this sequence. For example, $\text{MEX}(0, 1, 2, 4) = 3$, and $\text{MEX}(1, 2021) = 0$.

Input
The first line contains a single integer t ($1 \le t \le 30\,000$) — the number of test cases.

The first and only line of each test case contains two integers n and m ($0 \le n, m \le 10^9$).

Output
For each test case, print a single integer — the answer to the problem.

Example

input
5 3 5 4 6 3 2 69 696 123456 654321
output
4 3 0 640 530866

Note
In the first test case, the sequence is $3 \oplus 0, 3 \oplus 1, 3 \oplus 2, 3 \oplus 3, 3 \oplus 4, 3 \oplus 5$, or $3, 2, 1, 0, 7, 6$. The smallest non-negative integer which isn't present in the sequence i. e. the MEX of the sequence is 4.

In the second test case, the sequence is $4 \oplus 0, 4 \oplus 1, 4 \oplus 2, 4 \oplus 3, 4 \oplus 4, 4 \oplus 5, 4 \oplus 6$, or $4, 5, 6, 7, 0, 1, 2$. The smallest non-negative integer which isn't present in the sequence i. e. the MEX of the sequence is 3.

In the third test case, the sequence is $3 \oplus 0, 3 \oplus 1, 3 \oplus 2$, or $3, 2, 1$. The smallest non-negative integer which isn't present in the sequence i. e. the MEX of the sequence is 0.

D. Diane

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

You are given an integer n. Find any string s of length n consisting only of English lowercase letters such that each non-empty substring of s occurs in s an **odd** number of times. If there are multiple such strings, output any. It can be shown that such string always exists under the given constraints.

A string a is a substring of a string b if a can be obtained from b by deletion of several (possibly, zero or all) characters from the beginning and several (possibly, zero or all) characters from the end.

Input
The first line contains a single integer t ($1 \le t \le 500$) — the number of test cases.

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

It is guaranteed that the sum of n over all test cases doesn't exceed $3 \cdot 10^5$.

Output
For each test case, print a single line containing the string s. If there are multiple such strings, output any. It can be shown that such string always exists under the given constraints.

Example

input
4 3 5 9 19
output
abc diane bbcaabbba youarethecutestuwuu

Note
In the first test case, each substring of "abc" occurs exactly once.

In the third test case, each substring of "bbcaabbba" occurs an odd number of times. In particular, "b" occurs 5 times, "a" and "bb" occur 3 times each, and each of the remaining substrings occurs exactly once.

E. You

time limit per test: 5 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

You are given a tree with n nodes. As a reminder, a tree is a connected undirected graph without cycles.

Let $a_1, a_2, \ldots, a_n$ be a sequence of integers. Perform the following operation **exactly** n times:

- Select an **unerased** node u. Assign $a_u :=$ number of **unerased** nodes adjacent to u. Then, erase the node u along with all edges that have it as an endpoint.

For each integer k from 1 to n, find the number, modulo $998\,244\,353$, of different sequences $a_1, a_2, \ldots, a_n$ that satisfy the following conditions:

- it is possible to obtain a by performing the aforementioned operations **exactly** n times in some order.
- $\gcd(a_1, a_2, \ldots, a_n) = k$. Here, $\gcd$ means the greatest common divisor of the elements in a.

Input
The first line contains a single integer t ($1 \le t \le 10\,000$) — the number of test cases.

The first line of each test case contains a single integer n ($2 \le n \le 10^5$).

Each of the next $n - 1$ lines contains two integers u and v ($1 \le u, v \le n$) indicating there is an edge between vertices u and v. It is guaranteed that the given edges form a tree.

It is guaranteed that the sum of n over all test cases doesn't exceed $3 \cdot 10^5$.

Output

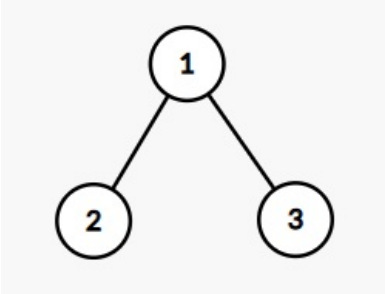
For each test case, print n integers in a single line, where for each k from 1 to n, the k-th integer denotes the answer when $\gcd$ equals to k.

Example

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

Note

In the first test case,



- If we delete the nodes in order $1 \to 2 \to 3$ or $1 \to 3 \to 2$, then the obtained sequence will be $a = [2, 0, 0]$ which has $\gcd$ equals to 2.
- If we delete the nodes in order $2 \to 1 \to 3$, then the obtained sequence will be $a = [1, 1, 0]$ which has $\gcd$ equals to 1.
- If we delete the nodes in order $3 \to 1 \to 2$, then the obtained sequence will be $a = [1, 0, 1]$ which has $\gcd$ equals to 1.
- If we delete the nodes in order $2 \to 3 \to 1$ or $3 \to 2 \to 1$, then the obtained sequence will be $a = [0, 1, 1]$ which has $\gcd$ equals to 1.

Note that here we are counting the number of different sequences, not the number of different orders of deleting nodes.