Codeforces Round #630 (Div. 2)

A. Exercising Walk

time limit per test: 2 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

Alice has a cute cat. To keep her cat fit, Alice wants to design an exercising walk for her cat!

Initially, Alice's cat is located in a cell (x, y) of an infinite grid. According to Alice's theory, cat needs to move:

- exactly a steps left: from (u, v) to $(u - 1, v)$;
- exactly b steps right: from (u, v) to $(u + 1, v)$;
- exactly c steps down: from (u, v) to $(u, v - 1)$;
- exactly d steps up: from (u, v) to $(u, v + 1)$.

Note that the moves can be performed in an **arbitrary order**. For example, if the cat has to move 1 step left, 3 steps right and 2 steps down, then the walk right, down, left, right, right, down is valid.

Alice, however, is worrying that her cat might get lost if it moves far away from her. So she hopes that her cat is **always** in the area $[x_1, x_2] \times [y_1, y_2]$, i.e. for every cat's position (u, v) of a walk $x_1 \le u \le x_2$ and $y_1 \le v \le y_2$ holds.

Also, note that the cat can visit the same cell multiple times.

Can you help Alice find out if there exists a walk satisfying her wishes?

Formally, the walk should contain exactly $a + b + c + d$ unit moves (a to the left, b to the right, c to the down, d to the up). Alice can do the moves in **any** order. Her current position (u, v) should **always** satisfy the constraints: $x_1 \le u \le x_2$, $y_1 \le v \le y_2$. The staring point is (x, y).

You are required to answer t test cases **independently**.

Input

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

The first line of each test case contains four integers a, b, c, d ($0 \le a, b, c, d \le 10^8$, $a + b + c + d \ge 1$).

The second line of the test case contains six integers x, y, x_1, y_1, x_2, y_2 ($-10^8 \le x_1 \le x \le x_2 \le 10^8$, $-10^8 \le y_1 \le y \le y_2 \le 10^8$).

Output

For each test case, output "YES" in a separate line, if there exists a walk satisfying her wishes. Otherwise, output "NO" in a separate line.

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

Example

input
6 3 2 2 2 0 0 -2 -2 2 2 3 1 4 1 0 0 -1 -1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 1 0 0 0 0 0 1 5 1 1 1 0 0 -100 -100 0 100 1 1 5 1 0 0 -100 -100 100 0

output
Yes No No Yes Yes Yes

Note

In the first test case, one valid exercising walk is

$$(0, 0) \to (-1, 0) \to (-2, 0) \to (-2, 1) \to (-2, 2) \to (-1, 2) \to (0, 2) \to (0, 1) \to (0, 0) \to (-1, 0)$$

B. Composite Coloring

time limit per test: 2 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

A positive integer is called *composite* if it can be represented as a product of two positive integers, both greater than 1. For example, the following numbers are composite: 6, 4, 120, 27. The following numbers aren't: 1, 2, 3, 17, 97.

Alice is given a sequence of n composite numbers $a_1, a_2, \ldots, a_n$.

She wants to choose an integer $m \le 11$ and color each element one of m colors from 1 to m so that:

- for each color from 1 to m there is at least one element of this color;
- each element is colored and colored exactly one color;
- the greatest common divisor of any two elements that are colored the same color is greater than 1, i.e. $\gcd(a_i, a_j) > 1$ for each pair i, j if these elements are colored the same color.

Note that equal elements can be colored different colors — you just have to choose one of m colors for each of the indices from 1 to n.

Alice showed already that if all $a_i \le 1000$ then she can always solve the task by choosing some $m \le 11$.

Help Alice to find the required coloring. Note that you don't have to minimize or maximize the number of colors, you just have to find the solution with some m from 1 to 11.

Input

The first line contains a single integer t ($1 \le t \le 1000$) — the number of test cases. Then the descriptions of the test cases follow.

The first line of the test case contains a single integer n ($1 \le n \le 1000$) — the amount of numbers in a sequence a.

The second line of the test case contains n composite integers $a_1, a_2, \ldots, a_n$ ($4 \le a_i \le 1000$).

It is guaranteed that the sum of n over all test cases doesn't exceed 10^4.

Output
For each test case print 2 lines. The first line should contain a single integer m ($1 \le m \le 11$) — the number of used colors. Consider colors to be numbered from 1 to m. The second line should contain any coloring that satisfies the above conditions. Print n integers $c_1, c_2, \ldots, c_n$ ($1 \le c_i \le m$), where c_i is the color of the i-th element. If there are multiple solutions then you can print any of them. Note that you don't have to minimize or maximize the number of colors, you just have to find the solution with some m from 1 to 11.

Remember that each color from 1 to m should be used at least once. Any two elements of the same color should not be coprime (i.e. their GCD should be greater than 1).

Example

input
3 3 6 10 15 2 4 9 23 437 519 865 808 909 391 194 291 237 395 323 365 511 497 781 737 871 559 731 697 779 841 961

output
1 1 1 1 2 2 1 11 4 7 8 10 7 3 10 7 7 8 3 1 1 5 5 9 2 2 3 3 4 11 6

Note
In the first test case, $\gcd(6, 10) = 2$, $\gcd(6, 15) = 3$ and $\gcd(10, 15) = 5$. Therefore, it's valid to color all elements the same color. Note that there are other colorings which satisfy Alice's requirement in this test case.

In the second test case there is only one element of each color, so the coloring definitely satisfies Alice's requirement.

C. K-Complete Word

time limit per test: 2 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

Word s of length n is called k-complete if

- s is a palindrome, i.e. $s_i = s_{n+1-i}$ for all $1 \le i \le n$;
- s has a period of k, i.e. $s_i = s_{k+i}$ for all $1 \le i \le n - k$.

For example, "abaaba" is a 3-complete word, while "abccba" is not.

Bob is given a word s of length n consisting of only lowercase Latin letters and an integer k, such that n is divisible by k. He wants to convert s to any k-complete word.

To do this Bob can choose some i ($1 \le i \le n$) and replace the letter at position i with some other lowercase Latin letter.

So now Bob wants to know the minimum number of letters he has to replace to convert s to any k-complete word.

Note that Bob can do zero changes if the word s is already k-complete.

You are required to answer t test cases **independently**.

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

The first line of each test case contains two integers n and k ($1 \le k < n \le 2 \cdot 10^5$, n is divisible by k).

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

It is guaranteed that word s only contains lowercase Latin letters. And it is guaranteed that the sum of n over all test cases will not exceed $2 \cdot 10^5$.

Output
For each test case, output one integer, representing the minimum number of characters he has to replace to convert s to any k-complete word.

Example

input
4 6 2 abaaba 6 3 abaaba 36 9 hippopotomonstrosesquippedaliophobia 21 7 wudixiaoxingxingheclp

output
2 0 23 16

Note
In the first test case, one optimal solution is a**a**aa**a**a.

In the second test case, the given word itself is k-complete.

D. Walk on Matrix

time limit per test: 2 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

Bob is playing a game named "Walk on Matrix".

In this game, player is given an $n \times m$ matrix $A = (a_{i,j})$, i.e. the element in the i-th row in the j-th column is $a_{i,j}$. Initially, player is located at position $(1, 1)$ with score $a_{1,1}$.

To reach the goal, position (n, m), player can move right or down, i.e. move from (x, y) to $(x, y + 1)$ or $(x + 1, y)$, as long as player is still on the matrix.

However, each move changes player's score to the bitwise AND of the current score and the value at the position he moves to.

Bob can't wait to find out the maximum score he can get using the tool he recently learnt — dynamic programming. Here is his algorithm for this problem.

Algorithm 1 Bob's Dynamic Programming Algorithm

Input: matrix $A = (a_{i,j})_{n \times m}$;
Output: maximum score S after reaching the goal;

```
initialize dp_{i,j} ← 0, for all 0 ≤ i ≤ n, 0 ≤ j ≤ m except dp_{0,1} ← a_{1,1}
for i = 1 → n do
    for j = 1 → m do
        dp_{i,j} ← max(dp_{i-1,j}&a_{i,j}, dp_{i,j-1}&a_{i,j})
    end for
end for
S ← dp_{n,m}
```

However, he suddenly realize that the algorithm above fails to output the maximum score for some matrix A. Thus, for any given non-negative integer k, he wants to find out an $n \times m$ matrix $A = (a_{i,j})$ such that

- $1 \le n, m \le 500$ (as Bob hates large matrix);
- $0 \le a_{i,j} \le 3 \cdot 10^5$ for all $1 \le i \le n, 1 \le j \le m$ (as Bob hates large numbers);
- the difference between the maximum score he can get and the output of his algorithm is **exactly** k.

It can be shown that for any given integer k such that $0 \le k \le 10^5$, there exists a matrix satisfying the above constraints.

Please help him with it!

Input
The only line of the input contains one single integer k ($0 \le k \le 10^5$).

Output
Output two integers n, m ($1 \le n, m \le 500$) in the first line, representing the size of the matrix.

Then output n lines with m integers in each line, $a_{i,j}$ in the $(i+1)$-th row, j-th column.

Examples

input
0
output
1 1 300000

input
1
output
3 4 7 3 3 1 4 8 3 6 7 7 7 3

Note
In the first example, the maximum score Bob can achieve is 300000, while the output of his algorithm is 300000.

In the second example, the maximum score Bob can achieve is $7\&3\&3\&3\&7\&3 = 3$, while the output of his algorithm is 2.

E. Height All the Same

time limit per test: 2 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

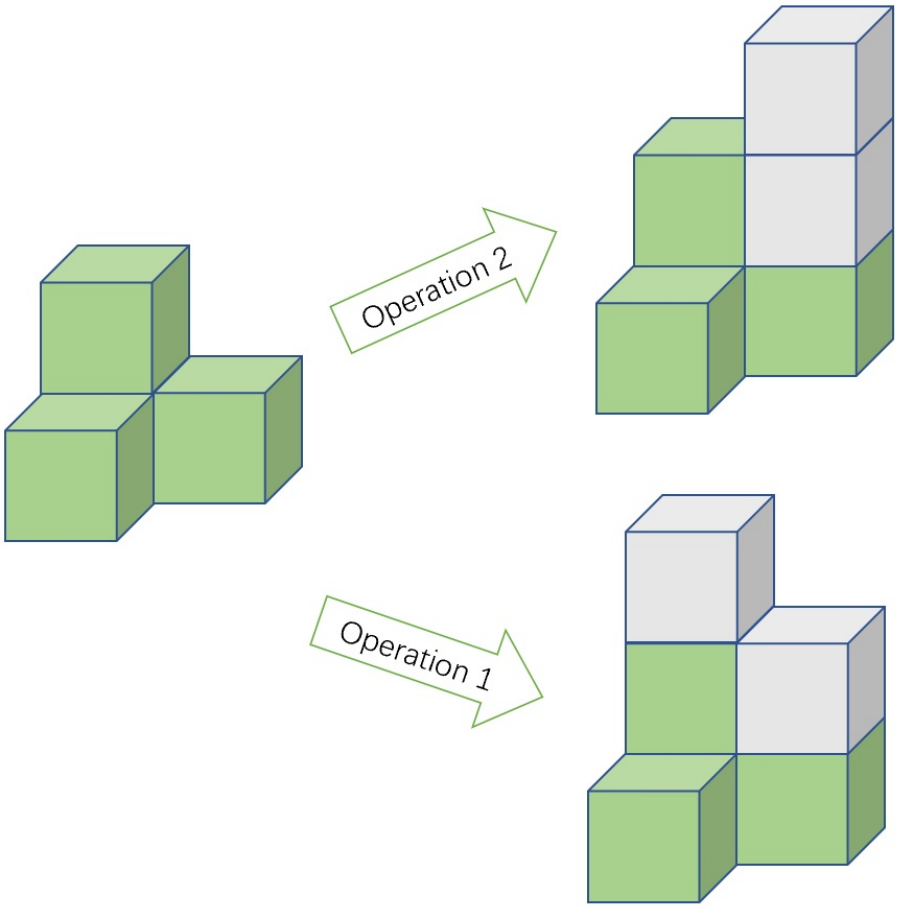
Alice has got addicted to a game called Sirtet recently.

In Sirtet, player is given an $n \times m$ grid. Initially $a_{i,j}$ cubes are stacked up in the cell (i, j). Two cells are called adjacent if they share a side. Player can perform the following operations:

- stack up one cube in two **adjacent** cells;
- stack up two cubes in one cell.

Cubes mentioned above are identical in height.

Here is an illustration of the game. States on the right are obtained by performing one of the above operations on the state on the left, and grey cubes are added due to the operation.



Player's goal is to **make the height of all cells the same** (i.e. so that each cell has the same number of cubes in it) using above operations.

Alice, however, has found out that on some starting grids she may never reach the goal no matter what strategy she uses. Thus, she is wondering the number of initial grids such that

- $L \le a_{i,j} \le R$ for all $1 \le i \le n$, $1 \le j \le m$;
- player can reach the goal using above operations.

Please help Alice with it. Notice that the answer might be large, please output the desired value modulo $998,244,353$.

Input

The only line contains four integers n, m, L and R ($1 \le n, m, L, R \le 10^9$, $L \le R$, $n \cdot m \ge 2$).

Output

Output one integer, representing the desired answer modulo $998,244,353$.

Examples

input
2 2 1 1
output
1

input
1 2 1 2
output
2

Note

In the first sample, the only initial grid that satisfies the requirements is $a_{1,1} = a_{2,1} = a_{1,2} = a_{2,2} = 1$. Thus the answer should be 1.

In the second sample, initial grids that satisfy the requirements are $a_{1,1} = a_{1,2} = 1$ and $a_{1,1} = a_{1,2} = 2$. Thus the answer should be 2.

F. Independent Set

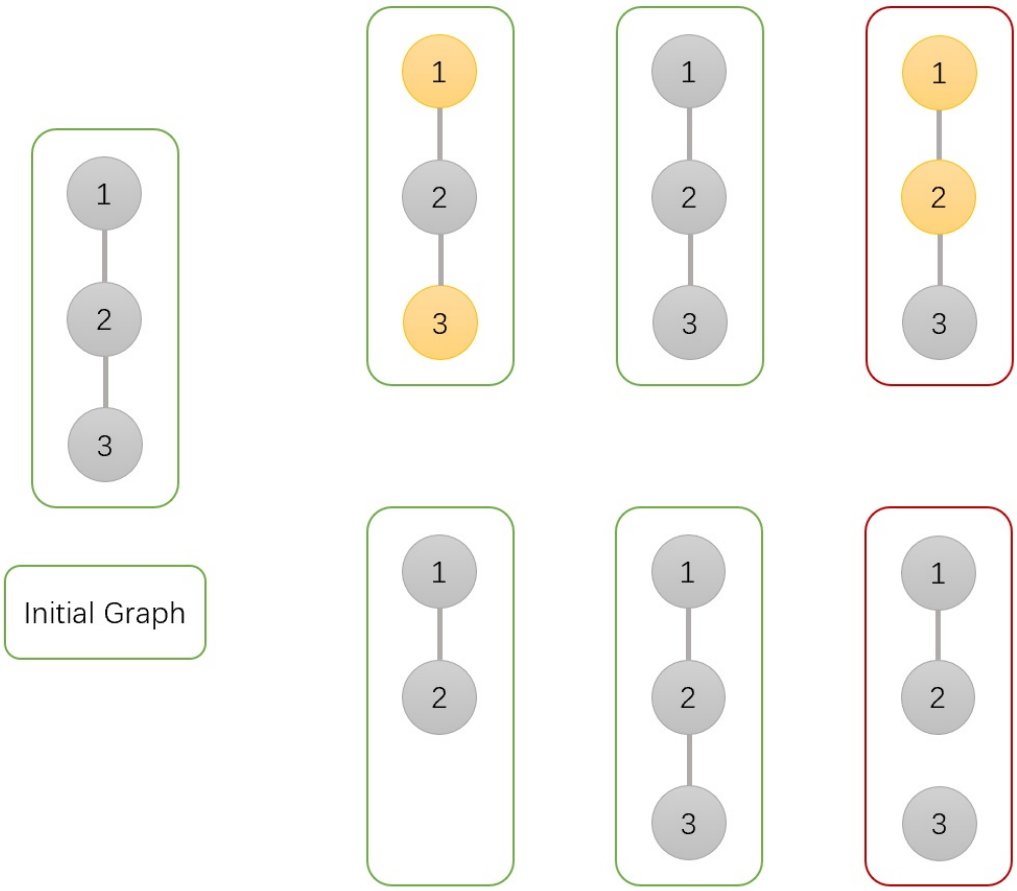
time limit per test: 2 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

Eric is the teacher of graph theory class. Today, Eric teaches independent set and edge-induced subgraph.

Given a graph $G = (V, E)$, an *independent set* is a subset of vertices $V' \subset V$ such that for every pair $u, v \in V'$, $(u, v) \notin E$ (i.e. no edge in E connects two vertices from V').

An *edge-induced subgraph* consists of a subset of edges $E' \subset E$ and all the vertices in the original graph that are incident on at least one edge in the subgraph.

Given $E' \subset E$, denote $G[E']$ the edge-induced subgraph such that E' is the edge set of the subgraph. Here is an illustration of those definitions:



Illustratio
independc
(Example
frame is M
independc

Illustrat
edge-ir
subgrap
(Examp
frame is
edge-ir
subgrap

In order to help his students get familiar with those definitions, he leaves the following problem as an exercise:

Given a tree $G = (V, E)$, calculate the sum of $w(H)$ over all except null edge-induced subgraph H of G, where $w(H)$ is the number of independent sets in H. Formally, calculate $\sum_{\emptyset \neq E' \subset E} w(G[E'])$.

Show Eric that you are smarter than his students by providing the correct answer as quickly as possible. Note that the answer might be large, you should output the answer modulo $998,244,353$.

Input

The first line contains a single integer n ($2 \leq n \leq 3 \cdot 10^5$), representing the number of vertices of the graph G.

Each of the following $n-1$ lines contains two integers u and v ($1 \leq u, v \leq n$, $u \neq v$), describing edges of the given tree.

It is guaranteed that the given edges form a tree.

Output

Output one integer, representing the desired value modulo $998,244,353$.

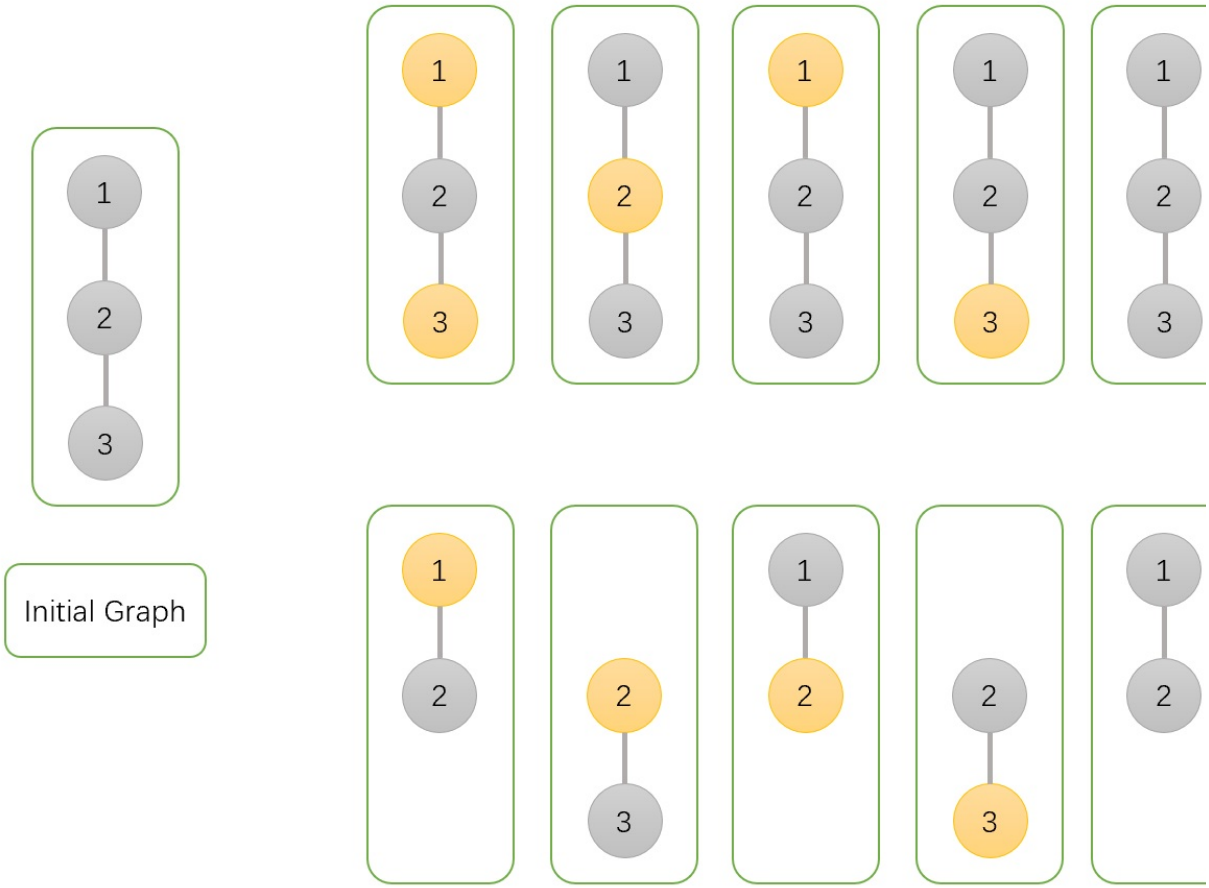
Examples

input
2 2 1
output
3

input
3 1 2 3 2
output
11

Note

For the second example, all independent sets are listed below.



G. No Monotone Triples

time limit per test: 2 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

Given a sequence of integers a of length n, a tuple (i, j, k) is called monotone triples if

- $1 \leq i < j < k \leq n$;
- $a_i \leq a_j \leq a_k$ or $a_i \geq a_j \geq a_k$ is satisfied.

For example, $a = [5, 3, 4, 5]$, then $(2, 3, 4)$ is monotone triples for sequence a while $(1, 3, 4)$ is not.

Bob is given a sequence of integers a of length n in a math exam. The exams itself contains questions of form L, R, for each of them he is asked to find any subsequence b **with size greater than** 2 **(i.e.** $|b| \geq 3$**)** of sequence $a_L, a_{L+1}, \ldots, a_R$.

Recall that an sequence b is a subsequence of sequence a if b can be obtained by deletion of several (possibly zero, or all) elements.

However, he hates monotone stuff, and he wants to find a subsequence **free from monotone triples**. Besides, he wants to find one subsequence with the **largest** length among all subsequences free from monotone triples for every query.

Please help Bob find out subsequences meeting the above constraints.

Input

The first line contains two integers n, q ($3 \leq n \leq 2 \cdot 10^5$, $1 \leq q \leq 2 \cdot 10^5$) — the length of sequence a and the number of queries.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($1 \leq a_i \leq 10^9$), representing the sequence a.

Then each of the following q lines contains two integers L, R ($1 \leq L, R \leq n$, $R - L \geq 2$).

Output

For each query, output 0 if there is no subsequence b satisfying the constraints mentioned in the legend. You can print the empty line after but that's not mandatory.

Otherwise, output one integer k ($k > 2$) denoting the length of sequence b, then output k integers $i_1, i_2, \ldots, i_k$ ($L \leq i_1 < i_2 < \ldots < i_k \leq R$) satisfying that $b_j = a_{i_j}$ for $1 \leq j \leq k$.

If there are multiple answers with the maximum length, print any of them.

Example

input
6 2 3 1 4 1 5 9 1 3 4 6
output
3 1 2 3 0

Note

For the first query, the given sequence itself is monotone triples free.

For the second query, it can be shown that there is no subsequence b with length greater than 2 such that b is monotone triples free.