Problem A. Squary Partition

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 megabytes

A set S of k distinct positive integers is called k-*squary* partition of n if it satisfies the following conditions:

- the sum of the k integers equals to n.
- there exists a subset of S which contains only $k-1$ numbers and sums up to a square of integer.

For example, a set $\{1, 2, 26, 193\}$ is a squary partition of 222 because $1 + 2 + 26 + 193 = 222$ and $1 + 2 + 193 = 196 = 14 \times 14$.

Chiaki has two integers n and k, and she would like to know whether there exists a k-squary partition of n.

Input

There are multiple test cases. The first line of input contains an integer T ($1 \le T \le 10^5$), indicating the number of test cases. For each test case:

The first line contains two integers n and k ($1 \le n \le 10^5, 2 \le k \le 30$).

Output

For each case, if there exists a k-squary partition of n, output "YES" (without the quotes) in the first line. Then, output k distinct positive integers in the second line denoting a k-squary partition of n. Otherwise, just output "NO" (without the quotes) in a single line.

Example

standard input	standard output
3 2 2 5 2 222 4	NO YES 1 4 YES 26 1 2 193

Problem B. Edges

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 megabytes

Chiaki has a simple graph with n vertices and m edges. She would like to delete an edge and then add a new edge which is not in the graph before, such that the resulting graph is connected (i.e. there is a path between every pair of vertices).

Chiaki would like to know the number of ways. Note that, Chiaki would never add a self loop into the graph.

Input

The first line contains two integers n and m ($1 \le n \le 10^5, 0 \le m \le 2 \times 10^5$) – the number of vertices and the number of edges.

Each of the next m lines contains two integers a_i and b_i ($1 \le a_i, b_i \le n, a_i \ne b_i$).

Output

Output an integer denoting the number of ways.

Example

standard input	standard output
4 4 1 2 2 3 1 3 3 4	8

Problem C. Cactusophobia

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Tree is a connected undirected graph that has no cycles. Edge cactus is a connected undirected graph without loops and parallel edges, such that each edge belongs to at most one cycle.

Chiaki has an edge cactus, each edge of this graph has some color.

Chiaki would like to remove the minimal number of edges in such way that his cactus turned to a tree. Chiaki wants to make it in such a way that there were edges of as many different colors in the resulting tree, as possible. Help him to find how many different colors can the resulting tree have.

Input

The first line contains two integers: n, m ($2 \le n \le 10^4$) – the number of vertices and the number of edges in Chiaki's graph, respectively.

The following m lines contain three integers each: u, v, c ($1 \le u, v \le n, u \ne v, 1 \le c \le m$) – the numbers of vertices connected by the corresponding edge, and its color. It is guaranteed that the described graph is indeed an edge cactus.

Output

Output one integer: the maximal number of different colors that the resulting tree can have.

Examples

standard input	standard output
4 4 1 2 4 2 3 1 3 4 2 4 2 3	3
7 9 1 2 1 2 3 4 3 1 5 1 4 5 4 5 2 5 1 6 1 6 4 6 7 6 7 1 3	6

Problem D. Beautiful Partition

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 megabytes

Today Chiaki learned about greatest common divisor of a set of integers. She liked the idea so much, that now she tries to find it in anything she sees.

So during the Computer Science lesson today after the teacher has written an array of integers at the blackboard, Chiaki has noticed that its elements can be divided into two parts M_1 and M_2 so that $\gcd(M_1)$ and $\gcd(M_2)$ are both quite big. Here $\gcd(M)$ is the greatest common divisor of all numbers in M.

Chiaki has decided to generalize the problem. For a given array she wants to find the way to divide its elements into two non-empty parts M_1 and M_2 so that $\min(\gcd(M_1), \gcd(M_2))$ was maximal possible.

Input

Input contains several tests. The first line contains the number of tests t ($1 \le t \le 1000$).

Each test is given in two lines. The first line contains an integer n ($2 \le n \le 5 \times 10^4$) – the size of the array. The second line contains n integers a_i ($1 \le a_i \le 10^9$) – the elements of the array.

The sum of values of n for all tests in one input doesn't exceed 5×10^4.

Output

For each test print one integer on a line — the maximal possible value of $\min(\gcd(M_1), \gcd(M_2))$.

Example

standard input	standard output
3 5 3 2 4 6 9 3 3 5 14 4 6 4 6 6	2 1 4

Problem E. Parsing

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	256 megabytes

Consider the problem of checking whether a string can be split into words from a specific dictionary. One of the simplest algorithms that solve this problem is a greedy algorithm: each time, find a longest prefix that is present in the dictionary as a separate word and then delete it.

This algorithm is extremely simple to implement. However, it may produce wrong result for some strings. For example, if you take a list of all words that exist in English as a dictionary, then the string "workingrass" will not be split correctly, because the prefix "working" will be removed first, and then it is impossible to split "rass" into the words. At the same time, this string can be split into words "work", "in" and "grass".

Now, Chiaki has generated a dictionary. You need to check whether the dictionary can correctly split all the string using the greedy algorithm. Otherwise, give an example of a string such that it is possible to split it into words from the dictionary, but failed using the greedy algorithm described above.

Input

The first line contains an integer n ($1 \le n \le 250$) – the number of words in the dictionary.

Each of the next n lines contains a nonempty string which consists only of lowercase English letters. The length of each string does not exceed 500.

Output

If you cannot find an example, output "Good vocabulary!". Otherwise, output the example in a single line. If there are several such examples, output the shortest one. If there are still several, output any of them.

Examples

standard input	standard output
3 ab cd abcd	Good vocabulary!
3 aba ab ac	abab

Problem F. Prince

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

After long adventures, the Prince is incredibly close to saving the Princess. He only had to overcome the corridor filled with traps.

The corridor is a straight line endless in both directions. At the initial time, the Prince is at point 0. In x meters from him there is a door leading to the room of the Princess. In one second, the Prince can walk one meter in either direction or stay where he is and do nothing.

The Prince found out that there are n traps in the corridor. The traps operate as follows: the i-th trap will appear at a_i seconds and disappear after t_i seconds. The trap occupies a part of the corridor from l_i to r_i, and if the Prince turns out to be strictly inside the trapped area, he will inevitably die. The prince can safely be on the edge of the trap, and immediately passes through the door, even if at the same time there appears a trap.

Your task is to find the minimum time that the Prince can get to the door leading to the Princess's room.

Input

The first line contains two integers n and x ($0 \le n \le 1000, 1 \le x \le 10^5$) – the number of traps and the position of the door.

Each of the next line contains four integers a_i, t_i, l_i and r_i ($1 \le a_i, t_i \le 10^6, -10^6 \le l_i < r_i \le 10^6$) denoting the i-th trap.

Output

Output the minimum time to the door. If it is impossible to reach the door, output "Impossible" (without the quotes) instead.

Example

standard input	standard output
3 2 1 1 -1 2 3 2 1 3 6 1 0 2	6