Gracias Sponsors!

Organizador

Diamond

Gold

Contest 1 - Inicial

A

Range Sorting

You are given an array a, consisting of n distinct integers $a_1, a_2, \ldots, a_n$.

Define the *beauty* of an array $p_1, p_2, \ldots p_k$ as the minimum amount of time needed to sort this array using an arbitrary number of *range-sort* operations. In each range-sort operation, you will do the following:

- Choose two integers l and r ($1 \le l < r \le k$).
- Sort the subarray $p_l, p_{l+1}, \ldots, p_r$ in $r - l$ seconds.

Please calculate the sum of beauty over all subarrays of array a.

A subarray of an array is defined as a sequence of consecutive elements of the array.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \le t \le 5 \cdot 10^3$). The description of the test cases follows.

The first line of each test case contains a single integer n ($1 \le n \le 5 \cdot 10^3$) — the length of the array a.

The second line of each test case consists of n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^9$). It is guaranteed that all elements of a are pairwise distinct.

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

Output

For each test case, output the sum of beauty over all subarrays of array a.

Example

input Copy

```
5
2
6 4
3
3 10 6
4
4 8 7 2
5
9 8 2 4 6
12
2 6 13 3 15 5 10 8 16 9 11 18
```

output Copy

```
1
2
8
16
232
```

Note

In the first test case:

- The subarray $[6]$ is already sorted, so its beauty is 0.
- The subarray $[4]$ is already sorted, so its beauty is 0.
- You can sort the subarray $[6, 4]$ in one operation by choosing $l = 1$ and $r = 2$. Its beauty is equal to 1.

The sum of beauty over all subarrays of the given array is equal to $0 + 0 + 1 = 1$.
In the second test case:

- The subarray $[3]$ is already sorted, so its beauty is 0.
- The subarray $[10]$ is already sorted, so its beauty is 0.
- The subarray $[6]$ is already sorted, so its beauty is 0.
- The subarray $[3, 10]$ is already sorted, so its beauty is 0.
- You can sort the subarray $[10, 6]$ in one operation by choosing $l = 1$ and $r = 2$. Its beauty is equal to $2 - 1 = 1$.
- You can sort the subarray $[3, 10, 6]$ in one operation by choosing $l = 2$ and $r = 3$. Its beauty is equal to $3 - 2 = 1$.

The sum of beauty over all subarrays of the given array is equal to $0 + 0 + 0 + 0 + 1 + 1 = 2$.

B

In Case of an Invasion, Please...

After Curiosity discovered not just water on Mars, but also an aggressive, bloodthirsty bunch of aliens, the Louvain-la-Neuve municipal government decided to take precautionary measures; they built shelters in order to shelter everyone in the city in the event of an extraterrestial attack.

Several alien-proof shelters have been erected throughout the city, where citizens can weather an alien invasion. However, due to municipal regulations and local building codes the shelters are limited in size. This makes it necessary for the government to assign every citizen a shelter to calmly direct themselves towards in the rare event of a fleet of UFOs blotting out the sun. Conditional on no shelter being assigned more people than it can fit, it is of the utmost importance that the time it takes until everyone has arrived at a shelter is minimized.

We model Louvain-la-Neuve as a network of n locations at which people live, connected by m bidirectional roads. Located at s points throughout the city are the shelters, each with a given maximum capacity. What is the minimum amount of time it takes for everyone to arrive at a shelter, when we assign people to shelters optimally?

The Louvain-la-Neuve municipal government has made sure that there is enough shelter capacity for its citizens and all shelters can be reached from any location, i.e. it is always possible to shelter everyone in some way.

Input

- On the first line are three integers, the number of locations $1 \leq n \leq 10^5$, roads $0 \leq m \leq 2 \cdot 10^5$, and shelters $1 \leq s \leq 10$.
- Then follows a line with n integers $0 \leq p_i \leq 10^9$, indicating the the number of people living at location $1 \leq i \leq n$.
- Then follow m lines containing three integers $1 \leq u, v \leq n$ and $1 \leq w \leq 10^9$ indicating that there is a bidirectional road connecting u and v that takes w time to traverse. For any two locations there is at most one road connecting them directly, and no road connects a location to itself.
- Finally follow s lines with two integers $1 \leq s_i \leq n$ and $1 \leq c_i \leq 10^9$, indicating that there is a shelter with capacity c_i at location s_i.

Output

Print the minimum amount of time it takes to shelter everyone.

Sample Input 1

```
2 1 1
3 2
1 2 4
1 6
```

Sample Output 1

```
4
```

Sample Input 2

```
4 5 2
2 0 0 2
1 2 6
1 3 2
2 3 3
3 4 4
4 2 6
3 2
2 2
```

Sample Output 2

```
5
```

Sample Input 3

```
7 8 3
0 1 1 1 1 0 2
1 2 1
2 3 1
3 1 1
4 6 5
4 3 1
6 7 10
7 5 3
5 6 3
6 5
1 1
2 1
```

Sample Output 3

```
6
```

Sample Input 4

```
2 1 1
0 2
1 2 1000000000
2 2
```

Sample Output 4

```
0
```

C 1D Sokoban

You are playing a game similar to Sokoban on an infinite number line. The game is discrete, so you only consider integer positions on the line.

You start on a position 0. There are n boxes, the i-th box is on a position a_i. All positions of the boxes are distinct. There are also m special positions, the j-th position is b_j. All the special positions are also distinct.

In one move you can go one position to the left or to the right. If there is a box in the direction of your move, then you push the box to the next position in that direction. If the next position is taken by another box, then that box is also pushed to the next position, and so on. **You can't go through the boxes**. **You can't pull the boxes towards you**.

You are allowed to perform any number of moves (possibly, zero). Your goal is to place as many boxes on special positions as possible. Note that some boxes can be initially placed on special positions.

Input

The first line contains a single integer t ($1 \le t \le 1000$) — the number of testcases.

Then descriptions of t testcases follow.

The first line of each testcase contains two integers n and m ($1 \le n, m \le 2 \cdot 10^5$) — the number of boxes and the number of special positions, respectively.

The second line of each testcase contains n distinct integers in the increasing order $a_1, a_2, \dots, a_n$ ($-10^9 \le a_1 < a_2 < \dots < a_n \le 10^9$; $a_i \neq 0$) — the initial positions of the boxes.

The third line of each testcase contains m distinct integers in the increasing order $b_1, b_2, \dots, b_m$ ($-10^9 \le b_1 < b_2 < \dots < b_m \le 10^9$; $b_i \neq 0$) — the special positions.

The sum of n over all testcases doesn't exceed $2 \cdot 10^5$. The sum of m over all testcases doesn't exceed $2 \cdot 10^5$.

Output

For each testcase print a single integer — the maximum number of boxes that can be placed on special positions.

Example

input Copy

```
5
5 6
-1 1 5 11 15
-4 -3 -2 6 7 15
2 2
-1 1
-1000000000 1000000000
2 2
-1000000000 1000000000
-1 1
3 5
-1 1 2
-2 -1 1 2 5
2 1
1 2
10
```

output Copy

```
4
2
0
3
1
```

Note

In the first testcase you can go 5 to the right: the box on position 1 gets pushed to position 6 and the box on position 5 gets pushed to position 7. Then you can go 6 to the left to end up on position -1 and push a box to -2. At the end, the boxes are on positions $[-2, 6, 7, 11, 15]$, respectively. Among them positions $[-2, 6, 7, 15]$ are special, thus, the answer is 4.

In the second testcase you can push the box from -1 to -10^9, then the box from 1 to 10^9 and obtain the answer 2.

The third testcase showcases that you are not allowed to pull the boxes, thus, you can't bring them closer to special positions.

In the fourth testcase all the boxes are already on special positions, so you can do nothing and still obtain the answer 3.

In the fifth testcase there are fewer special positions than boxes. You can move either 8 or 9 to the right to have some box on position 10.

D

Stressful Training

Berland SU holds yet another training contest for its students today. n students came, each of them brought his laptop. However, it turned out that everyone has forgot their chargers!

Let students be numbered from 1 to n. Laptop of the i-th student has charge a_i at the beginning of the contest and it uses b_i of charge per minute (i.e. if the laptop has c charge at the beginning of some minute, it becomes $c - b_i$ charge at the beginning of the next minute). The whole contest lasts for k minutes.

Polycarp (the coach of Berland SU) decided to buy a **single** charger so that all the students would be able to successfully finish the contest. He buys the charger at the same moment the contest starts.

Polycarp can choose to buy the charger with any non-negative (zero or positive) integer power output. The power output is chosen before the purchase, it can't be changed afterwards. Let the chosen power output be x. **At the beginning of each minute** (from the minute contest starts to the last minute of the contest) he can plug the charger into any of the student's laptops and use it for some **integer** number of minutes. If the laptop is using b_i charge per minute then it will become $b_i - x$ per minute while the charger is plugged in. Negative power usage rate means that the laptop's charge is increasing. The charge of any laptop isn't limited, it can become infinitely large. The charger can be plugged in no more than one laptop at the same time.

The student successfully finishes the contest if the charge of his laptop never is below zero at the beginning of some minute (from the minute contest starts to the last minute of the contest, zero charge is allowed). The charge of the laptop of the minute the contest ends doesn't matter.

Help Polycarp to determine the minimal possible power output the charger should have so that all the students are able to successfully finish the contest. Also report if no such charger exists.

Input

The first line contains two integers n and k ($1 \le n \le 2 \cdot 10^5$, $1 \le k \le 2 \cdot 10^5$) — the number of students (and laptops, correspondigly) and the duration of the contest in minutes.

The second line contains n integers $a_1, a_2, \dots, a_n$ ($1 \le a_i \le 10^{12}$) — the initial charge of each student's laptop.

The third line contains n integers $b_1, b_2, \dots, b_n$ ($1 \le b_i \le 10^7$) — the power usage of each student's laptop.

Output

Print a single non-negative integer — the minimal possible power output the charger should have so that all the students are able to successfully finish the contest.

If no such charger exists, print `-1`.

Examples

input Copy

```
2 4
3 2
4 2
```

output Copy

```
5
```

input Copy

```
1 5
4
2
```

output

```
1
```

input

```
1 6
4
2
```

output

```
2
```

input

```
2 2
2 10
3 15
```

output

```
-1
```

Note

Let's take a look at the state of laptops in the beginning of each minute on the first example with the charger of power 5:

1. charge: $[3, 2]$, plug the charger into laptop 1;
2. charge: $[3 - 4 + 5, 2 - 2] = [4, 0]$, plug the charger into laptop 2;
3. charge: $[4 - 4, 0 - 2 + 5] = [0, 3]$, plug the charger into laptop 1;
4. charge: $[0 - 4 + 5, 3 - 2] = [1, 1]$.

The contest ends after the fourth minute.

However, let's consider the charger of power 4:

1. charge: $[3, 2]$, plug the charger into laptop 1;
2. charge: $[3 - 4 + 4, 2 - 2] = [3, 0]$, plug the charger into laptop 2;
3. charge: $[3 - 4, 0 - 2 + 4] = [-1, 2]$, the first laptop has negative charge, thus, the first student doesn't finish the contest.

In the fourth example no matter how powerful the charger is, one of the students won't finish the contest.

E

Appleman and Toastman

Appleman and Toastman play a game. Initially Appleman gives one group of n numbers to the Toastman, then they start to complete the following tasks:

- Each time Toastman gets a group of numbers, he sums up all the numbers and adds this sum to the score. Then he gives the group to the Appleman.
- Each time Appleman gets a group consisting of a single number, he throws this group out. Each time Appleman gets a group consisting of more than one number, he splits the group into two non-empty groups (he can do it in any way) and gives each of them to Toastman.

After guys complete all the tasks they look at the score value. What is the maximum possible value of score they can get?

Input

The first line contains a single integer n $(1 \le n \le 3 \cdot 10^5)$. The second line contains n integers a_1, a_2, ..., a_n $(1 \le a_i \le 10^6)$ — the initial group that is given to Toastman.

Output

Print a single integer — the largest possible score.

Examples

input

```
3
3 1 5
```

output

```
26
```

input

```
1
10
```

output

```
10
```

Note

Consider the following situation in the first example. Initially Toastman gets group [3, 1, 5] and adds 9 to the score, then he give the group to Appleman. Appleman splits group [3, 1, 5] into two groups: [3, 5] and [1]. Both of them should be given to Toastman. When Toastman receives group [1], he adds 1 to score and gives the group to Appleman (he will throw it out). When Toastman receives group [3, 5], he adds 8 to the score and gives the group to Appleman. Appleman splits [3, 5] in the only possible way: [5] and [3]. Then he gives both groups to Toastman. When Toastman receives [5], he adds 5 to the score and gives the group to Appleman (he will throws it out). When Toastman receives [3], he adds 3 to the score and gives the group to Appleman (he will throws it out). Finally Toastman have added 9 + 1 + 8 + 5 + 3 = 26 to the score. This is the optimal sequence of actions.

F Loppinha the boy who likes sopinha

Loppinha loves to go to the gym with his friends. He takes his training sessions very seriously, and he always follows his schedule very strictly. He just created a new plan where he has set exactly what he is going to do at every single one of the N minutes of the session.

Loppinha loves having a sopinha (small soup) before training. The soup contains K grams of protein. As you can imagine, he needs that protein to be able to endure his very tough exercises. He is burning protein at every minute he is working out, and the amount of protein he burns in a minute depends on how many minutes he has been working out without a minute of rest. If he has trained for p minutes without resting, in the $(p+1)$-th minute of workout he is going to burn $p+1$ grams of protein.

Of course, if he doesn't have enough protein at any moment he dies. For example, if he has 3 grams of protein at a moment and at that minute his workout requires 4, if he does the workout he dies. Given his schedule and the amount of protein K he has before starting the training session, Loppinha, who is willing to change some minutes of his workout, wants to know what is the minimum amount of minutes he has to change from working out to resting so he does not die.

Input

The first line of the input contains two integers N $(1 \le N \le 450)$ and K $(1 \le K \le 10^7)$, indicating the number of minutes in Loppinha's training session and the amount of proteins in his soup. The next line contains a string with N characters, either 0 or 1, where 0 indicates a minute of rest and a 1 indicates a minute of workout.

Output

Output a single integer - the minimum number of minutes Loppinha has to switch from workout to rest so that he can finish his exercises without dying.

Examples

input Copy

```
4 2
1101
```

output Copy

```
1
```

input Copy

```
10 5
1101100111
```

output Copy

```
3
```

input Copy

```
3 1
111
```

output Copy

```
2
```

Note

In the first test case, if he changes the second minute to rest, he will consume exactly 2 grams of protein. Notice that if he changes the last minute to rest instead, he would need 3 grams of protein to complete his workout session without dying.

G Contest Start

There are n people participating in some contest, they start participating in x minutes intervals. That means the first participant starts at time 0, the second participant starts at time x, the third — at time $2 \cdot x$, and so on.

Duration of contest is t minutes for each participant, so the first participant finishes the contest at time t, the second — at time $t + x$, and so on. When a participant finishes the contest, their dissatisfaction equals to the number of participants that started the contest (or starting it now), but haven't yet finished it.

Determine the sum of dissatisfaction of all participants.

Input

The first line contains a single integer k $(1 \leq k \leq 1000)$ — the number of test cases.

Each of the next k lines contains three integers n, x, t $(1 \leq n, x, t \leq 2 \cdot 10^9)$ — the number of participants, the start interval and the contest duration.

Output

Print k lines, in the i-th line print the total dissatisfaction of participants in the i-th test case.

Example

input

```
4
4 2 5
3 1 2
3 3 10
2000000000 1 2000000000
```

output

```
5
3
3
1999999999000000000
```

Note

In the first example the first participant starts at 0 and finishes at time 5. By that time the second and the third participants start, so the dissatisfaction of the first participant is 2.

The second participant starts at time 2 and finishes at time 7. By that time the third the fourth participants start, so the dissatisfaction of the second participant is 2.

The third participant starts at 4 and finishes at 9. By that time the fourth participant starts, so the dissatisfaction of the third participant is 1.

The fourth participant starts at 6 and finishes at 11. By time 11 everyone finishes the contest, so the dissatisfaction of the fourth participant is 0.

In the second example the first participant starts at 0 and finishes at time 2. By that time the second participants starts, and the third starts at exactly time 2. So the dissatisfaction of the first participant is 2.

The second participant starts at time 1 and finishes at time 3. At that time the third participant is solving the contest.

H

Chat room

Vasya has recently learned to type and log on to the Internet. He immediately entered a chat room and decided to say hello to everybody. Vasya typed the word s. It is considered that Vasya managed to say hello if several letters can be deleted from the typed word so that it resulted in the word "`hello`". For example, if Vasya types the word "`ahhellllloou`", it will be considered that he said hello, and if he types "`hlelo`", it will be considered that Vasya got misunderstood and he didn't manage to say hello. Determine whether Vasya managed to say hello by the given word s.

Input

The first and only line contains the word s, which Vasya typed. This word consisits of small Latin letters, its length is no less that 1 and no more than 100 letters.

Output

If Vasya managed to say hello, print "YES", otherwise print "NO".

Examples

input Copy

```
ahhellllloou
```

output Copy

```
YES
```

input Copy

```
hlelo
```

output Copy

```
NO
```

I Competitive Programmer

Bob is a competitive programmer. He wants to become red, and for that he needs a strict training regime. He went to the annual meeting of grandmasters and asked n of them how much effort they needed to reach red.

"Oh, I just spent x_i **hours** solving problems", said the i-th of them.

Bob wants to train his math skills, so for each answer he wrote down the number of **minutes** ($60 \cdot x_i$), thanked the grandmasters and went home. Bob could write numbers with leading zeroes — for example, if some grandmaster answered that he had spent 2 hours, Bob could write 000120 instead of 120.

Alice wanted to tease Bob and so she took the numbers Bob wrote down, and for each of them she did one of the following independently:

- rearranged its digits, or
- wrote a random number.

This way, Alice generated n numbers, denoted y_1, ..., y_n.

For each of the numbers, help Bob determine whether y_i can be a permutation of a number divisible by 60 (possibly with leading zeroes).

Input

The first line contains a single integer n ($1 \leq n \leq 418$) — the number of grandmasters Bob asked.

Then n lines follow, the i-th of which contains a single integer y_i — the number that Alice wrote down.

Each of these numbers has between 2 and 100 digits '0' through '9'. They can contain leading zeroes.

Output

Output n lines.

For each i, output the following. If it is possible to rearrange the digits of y_i such that the resulting number is divisible by 60, output "`red`" (quotes for clarity). Otherwise, output "`cyan`".

Example

input Copy

```
6
603
006
205
228
1053
0000000000000000000000000000000000000000000000
```

output Copy

```
red
red
cyan
cyan
cyan
red
```

Note

In the first example, there is one rearrangement that yields a number divisible by 60, and that is 360.

In the second example, there are two solutions. One is 060 and the second is 600.

In the third example, there are 6 possible rearrangments: 025, 052, 205, 250, 502, 520. None of these numbers is divisible by 60.

In the fourth example, there are 3 rearrangements: 228, 282, 822.

In the fifth example, none of the 24 rearrangements result in a number divisible by 60.

In the sixth example, note that $000\dots0$ is a valid solution.

J Retakes

It's well known that student Nikita is a procrastinator. He has failed n exams, therefore he has to retake them. It's possible to retake no more than one exam per day.

i-th exam can be retaken at any day starting from the day t_i. However, lecturers don't want to see students doing nothing during the term. That's why they take money for retakes.

Retake of i-th exam costs $c_i + d$ roubles, where c_i is the initial cost of the retake and d is the number of days passed from the day when this exam became available to retake. Thus, if student retakes the exam just as it becomes available, he expends c_i roubles, if he retakes it at the next day — he expends $c_i + 1$ roubles, and so on.

Nikita is poor, lives at the dormitory and, in general, works for food, so he has no money. So every time when he needs money, he borrows from Pavel. Namely, every time when he decides to retake the exam, he comes to Pavel, borrows exactly that sum that he needs to pay for retake, no more and no less, and goes to retake.

Pavel has the special form of amnesia: he remembers only maximal debt. For example, if one borrows from him 100 roubles, then 200 roubles, then 200 roubles one more time, and, finally, 150 roubles, Pavel will be absolutely sure that the debt is equal to 200 roubles. Knowing this fact, make the retake schedule for Nikita in such a way that the sum of money to refund to Pavel is minimal. That is, for every exam find a day when Nikita should retake it. It's known that Nikita retakes any exam at the first attempt.

Input

The first line contains the only integer n ($1 \le n \le 200000$) — number of exams that Nikita should retake.

Each of the following n lines contains two integers t_i and c_i ($1 \le t_i \le 10^6$, $0 \le c_i \le 10^9$) separated by space — the first day the i-th exam can be retaken from, and its initial cost. Numbers $t_1, \ldots, t_n$ are given in non-descending order, i.e. $t_i \le t_j$ for all $i \le j$.

Output

Write n integers separated by spaces. i-th number should be a day when Nikita should retake i-th exam. If there are many schedules such that the debt is minimal, output any of them.

Examples

stdin	stdout
3 1 3 1 1 2 4	1 3 2
4 3 5 4 3 4 6 5 3	3 5 4 6

K

Twenty Four, Again

Yes, we know ... we've used Challenge 24 before for contest problems. In case you've never heard of Challenge 24 (or have a very short memory) the object of the game is to take 4 given numbers (the *base values*) and determine if there is a way to produce the value 24 from them using the four basic arithmetic operations (and parentheses if needed). For example, given the four base values `3 5 5 2`, you can produce 24 in many ways. Two of them are: `5*5-3+2` and `(3+5)*(5-2)`. Recall that multiplication and division have precedence over addition and subtraction, and that equal-precedence operators are evaluated left-to-right.

This is all very familiar to most of you, but what you probably don't know is that you can *grade* the quality of the expressions used to produce 24. In fact, we're sure you don't know this since we've just made it up. Here's how it works: A perfect grade for an expression is 0. Each use of parentheses adds one point to the grade. Furthermore, each inversion (that is, a swap of two adjacent values) of the original ordering of the four base values adds two points. The first expression above has a grade of 4, since two inversions are used to move the 3 to the third position. The second expression has a better grade of 2 since it uses no inversions but two sets of parentheses. As a further example, the initial set of four base values `3 6 2 3` could produce an expression of grade 3 — namely `(3+6+3)*2` — but it also has a perfect grade 0 expression — namely `3*6+2*3`. Needless to say, the lower the grade the "better" the expression.

Two additional rules we'll use: 1) you cannot use unary minus in any expression, so you can't take the base values `3 5 5 2` and produce the expression `-3+5*5+2`, and 2) division can only be used if the result is an integer, so you can't take the base values `2 3 4 9` and produce the expression `2/3*4*9`.

Given a sequence of base values, determine the lowest graded expression resulting in the value 24. And by the way, the initial set of base values `3 5 5 2` has a grade 1 expression — can you find it?

Input

Input consists of a single line containing 4 base values. All base values are between 1 and 100, inclusive.

Output

Display the lowest grade possible using the sequence of base values. If it is not possible to produce 24, display `impossible`.

Sample Input 1	Sample Output 1
`3 5 5 2`	`1`

Sample Input 2	Sample Output 2
`1 1 1 1`	`impossible`

L Graph Without Long Directed Paths

You are given a connected undirected graph consisting of n vertices and m edges. There are no self-loops or multiple edges in the given graph.

You have to direct its edges in such a way that the obtained directed graph does not contain any paths of length two or greater (where the length of path is denoted as the number of traversed edges).

Input

The first line contains two integer numbers n and m ($2 \le n \le 2 \cdot 10^5$, $n - 1 \le m \le 2 \cdot 10^5$) — the number of vertices and edges, respectively.

The following m lines contain edges: edge i is given as a pair of vertices u_i, v_i ($1 \le u_i, v_i \le n$, $u_i \ne v_i$). There are no multiple edges in the given graph, i. e. for each pair (u_i, v_i) there are no other pairs (u_i, v_i) and (v_i, u_i) in the list of edges. It is also guaranteed that the given graph is connected (there is a path between any pair of vertex in the given graph).

Output

If it is impossible to direct edges of the given graph in such a way that the obtained directed graph does not contain paths of length at least two, print "`NO`" in the first line.

Otherwise print "YES" in the first line, and then print **any** suitable orientation of edges: a binary string (the string consisting only of '`0`' and '`1`') of length m. The i-th element of this string should be '`0`' if the i-th edge of the graph should be directed from u_i to v_i, and '`1`' otherwise. Edges are numbered in the order they are given in the input.

Example

input Copy

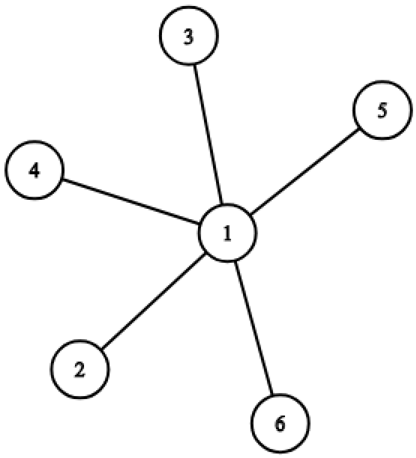
```
6 5
1 5
2 1
1 4
3 1
6 1
```

output Copy

```
YES
10100
```

Note

The picture corresponding to the first example:



And one of possible answers: