Epiphany 10.0

A. Three Piles Of Coin

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

There are 3 piles of coins contain A, B and C coins respectively. You can take 2 coins from one of the three piles and 1 coin from each of the remaining two piles in one operation(You are taking exactly 4 coins in a single operation). Is it possible to make all piles empty using this operation? You can use this operation as many times as possible.

Input
The first line contains $T(1 \leq T \leq 10^5)$ - the number of test cases.

The next T lines contain 3 space-separated integers A, B and $C(1 \leq A, B, C \leq 10^9)$.

Output
Print YES if it is possible to make all piles empty, otherwise, print NO.

Example

input	Copy
2 7 8 9 3 3 4	
output	**Copy**
YES NO	

Note
For the 1st test case,

$A\ B\ C$

7 8 9 (Initially)

6 7 7 (After 1st operation)

5 6 5 (After 2nd operation)

4 4 4 (After 3rd operation)

3 3 2 (After 4th operation)

2 1 1 (After 5th operation)

0 0 0 (After 6th operation)

B. Schrute's Potion

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

"As a farmer, I know that when an animal is sick sometimes the right thing to do is put it out of its misery. With the electricity we are using to keep Meredith alive, we could power a small fan for two days. You tell me what's unethical." -Dwight Schrute

Angela's cat Sprinkles is sick and she wants Dwight(just a colleague) to make her a medicine with formula A. On the other hand Dwight wants to put the cat out of misery.

A is an array of integers of length n. Dwight can put this cat out of misery by performing certain operation on A any number of times and making all elements of A equal.

In one operation Dwight can choose three valid consecutive elements of A and add 1 to all three of them. Dwight can perform the above operation **any number of times**.

Your task is to find out whether it is possible for Dwight to make all elements of A equal after certain number of operations.

You have to answer for T testcases.

Note:Use Fast I/O for better time optimization.

Input
First line of input will be T $(1 \leq T \leq 100)$, number of testcases.

First line of each testcase will denote n $(1 \leq n \leq 50000)$, number of elements in A.

Second line will consist of n integers where i^{th} integer will denote A_i ($-10^9 \leq A_i \leq 10^9$).

Output
Print answer for each testcase in new line.

For each testcase print "Yes" if it is possible to make all elemets equal and "No" otherwise.

Example

input Copy

```
3
7
5 4 1 -1 0 3 5
3
1 2 3
5
0 0 0 2 2
```

output Copy

```
Yes
No
Yes
```

Note
In the first testcase the operations will be as follows:

One operation on index [2,3,4] and the resultant array will be {5 5 2 0 0 3 5}.

Three operations on index [3,4,5] and the resultant array will be {5 5 5 3 3 3 5}.

Two operations on index [4,5,6] and the resultant array will be {5 5 5 5 5 5 5}.

C. The Imitation Game

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

War is going on between Country A and Country B, You are working with Country A and trying to win the war. Alan Turing, you're senior decoded the algorithm how the military of Country B encodes and decodes the message to communicate with each other.

The Algorithm is not that trivial, so you only have to solve a small task to help Alan Turing to decode the message. You are given a machine called $Enigma$. Enigma takes two input and produces one output, for simplicity let Enigma be a matrix of size $[10^5+1, 10^5+1]$, each position of this matrix is calculated by this formula

$$K = Enigma[i][j]\%mod = \begin{cases} X & if\ i == 0\ or\ j == 0 \\ \sum_{p=0}^{i} Enigma[p][j-1]\%mod & Otherwise \end{cases}$$

Here mod = 1000000007, and X will be given to you.

Your work is to Code this algorithm and win the war by printing the Enigma output K.

Input
The first line contains one integer Q, the number of queries.

Next Q lines contains three integers X, i, j.

$1 \leq Q \leq 10^5$

$0 \leq X,i,j \leq 10^5$

Output
For each query print the Value of Enigma output K in a new line, note that all the queries are independent.

Example

input Copy

```
5
3 0 4
5 5 0
11 2 5
7 5 11
111 10 10
```

output Copy

```
3
5
231
30576
20507916
```

D. D-Controllable Numbers

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

Dr David is trying to solve a question to control COVID-19 pandemic. You have to help him to solve this question as fast as possible such that the world can get relief from a pandemic.

The number is called D-Controllable if a digit D appears on odd positions only. The number should not contain any leading zero(s). The position of 1st

digit from left-side is odd.

For example, 707172, 137, 7, 98615 are 7-Controllable numbers. 471356, 777, 1017, 01027 are not 7-Controllable numbers.

You have to count D-Controllable numbers between L and R(L and R are inclusive). Print it modulo $10^9 + 7$.

Input
The first line contains T($1 \le T \le 500$) - the number of test cases.

The next T lines contain three space-separated integers L, R and D.($1 \le L \le R \le 10^{1000}$, $0 \le D \le 9$). L and R do not contain leading zero(s).

Output
Print D-Controllable numbers(modulo $10^9 + 7$) between L and R in a new line for each test case.

Example

input Copy

```
2
4 20 7
12 100 4
```

output Copy

```
16
80
```

Note
For the 1st test case, all the numbers except 17 are 7-Controllable numbers between 4 and 20.

E. Alice and Bob

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

Alice and Bob are playing a game.

They have a gaming machine and an array **A** of **N** coins initially showing **HEAD** on top as shown below.

The machine has its own array **AUX** of N coins.

Alice chooses two integers N and K.Then Bob chooses a type X($1 \le X \le 4$) representing initial AUX array.

- ($X == 1$) : $TAIL$ on every coin of AUX array, — TTTTTTT
- ($X == 2$) : $HEAD$ on every coin, — HHHHHHH
- ($X == 3$) : $HEAD$ on even position, $TAIL$ elsewhere — THTHTHTH
- ($X == 4$) : $TAIL$ on even position, $HEAD$ elsewhere — HTHTHTHT

They feed their array A into the gaming machine, and the machine will modify their array A and return it back to them.

The machine runs the following program

```
begin
    For i=1 to N
        if A[i] is HEAD
            For j=1 to j<=i
                temp = i + j - 1
                if temp > N
                    temp = temp - N
                Flip AUX[temp]
    A = AUX
end
```

In other words, if $A[i]$ is HEAD then the machine will flip i coins in AUX array starting from index i in a cyclic manner.(By flipping means $HEAD$ will be converted into $TAIL$ and $TAIL$ will be converted into $HEAD$)

Note that **after this task machine will be in the initial state. ie AUX array will be set to the same as it was before feeding to the machine.**

Alice and Bob play this game K times. ie they feed array A in the machine, take back modified array A from machine K times. If at the end of K games number of $HEADS$ in array A is greater than number of $TAILS$ then **Alice will win, else BOB will win.**

Your task is to choose initial state of AUX array on behalf of Bob, such that Bob will win at end.

Input
The input consists of multiple test cases. The first line contains an integer t ($1 \le t \le 5$) — the number of test cases. The description of the test cases follows.

The first and only line of each test case contains two integers N ($1 \le N \le 25$) , K ($0 \le K \le 10^{12}$)

Output
For each test case,

if Bob can win, print "BOB" followed by an integer X($1 \le X \le 4$). — type number.(if there are multiple answers print any)

If Bob can not win, print "ALICE"

Examples

input Copy

```
1
4 1
```

output Copy

```
BOB 4
```

input Copy

```
1
4 3
```

output Copy

```
BOB 2
```

input Copy

```
3
8 12
15 4
10 104
```

output Copy

```
BOB 1
BOB 1
ALICE
```

Note

In the first test, Below image show execution by step, for ($K = 1$), when ($X = 4$)

	Index-1	Index-2	Index-3	Index-4
A				
AUX				
AUX After i=1				
AUX After i=2				
AUX After i=3				
AUX After i=4				

After end of this program we have($HEADS = 0$) and ($TAILS = 4$). means BOB won.

For second test,array A changes according to below for ($X = 2$)

	Index-1	Index-2	Index-3	Index-4
Initial array 'A'				
Array 'A' after K=1				
Array 'A' after K=2				
Array 'A' after K=3				

so Bob won.

F. Santa's GIFT

time limit per test: 2 s.
memory limit per test: 256 MB
input: standard input
output: standard output

It's Christmas time and Santa wants to give some elixirs as gifts to the people.

There are N cities connected by two-way roads. Santa will drop some elixirs E_i on the road i and these elixirs E_i will be added to both cities connected by this road.

Santa wants to drop elixirs on every road such that at the end total elixirs on city i becomes W_i and below condition satisfied:

Each W_i is between $-N$ and N (inclusive) and parity of W_i is the same as the parity of the number of roads connected to city i. Value of E_i is between $-2 \cdot N^2$ and $2 \cdot N^2$ (inclusive).

Input

The first line contains two integers N and M ($2 \le N \le 10^5$, $N - 1 \le m \le 10^5$) — the number of cities and the number of roads.

The next line contains N integers $W_1, W_2, \ldots, W_n$ ($-N \le W_i \le N$). It is guaranteed that the parity of W_i is the same as the parity of the

number of roads connected to city i.

The next m lines describe roads. The i-th of these lines contains two integers u_i and v_i ($1 \le u_i, v_i \le n$; $u_i \ne v_i$), meaning that the i-th road connects cities u_i and v_i.

It is guaranteed that the given graph is connected and does not contain loops and multiple edges.

Output

If there is no solution, print "NO".

Otherwise print "YES" and then m lines, the i-th of them is E_i ($-2 \cdot N^2 \le E_i \le 2 \cdot N^2$).

Examples

input Copy

```
3 3
2 2 2
1 2
2 3
1 3
```

output Copy

```
YES
1
1
1
```

input Copy

```
4 3
-1 0 2 1
1 2
2 3
3 4
```

output Copy

```
YES
-1
1
1
```

input Copy

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

output Copy

```
YES
3
5
3
-1
-3
5
```

input Copy

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

output Copy

```
NO
```