



## B. Progressive Square

time limit per test: 2 seconds

memory limit per test: 256 megabytes

input: standard input

output: standard output

A *progressive square* of size  $n$  is an  $n \times n$  matrix. Maxim chooses three integers  $a_{1,1}$ ,  $c$ , and  $d$  and constructs a *progressive square* according to the following rules:

$$a_{i+1,j} = a_{i,j} + c$$

$$a_{i,j+1} = a_{i,j} + d$$

For example, if  $n = 3$ ,  $a_{1,1} = 1$ ,  $c = 2$ , and  $d = 3$ , then the *progressive square* looks as follows:

$$\begin{pmatrix} 1 & 4 & 7 \\ 3 & 6 & 9 \\ 5 & 8 & 11 \end{pmatrix}$$

Last month Maxim constructed a *progressive square* and remembered the values of  $n$ ,  $c$ , and  $d$ . Recently, he found an array  $b$  of  $n^2$  integers in random order and wants to make sure that these elements are the elements of **that specific** square.

It can be shown that for any values of  $n$ ,  $a_{1,1}$ ,  $c$ , and  $d$ , there exists exactly one *progressive square* that satisfies all the rules.

### Input

The first line contains an integer  $t$  ( $1 \leq t \leq 10^4$ ) — the number of test cases.

The first line of each test case contains three integers  $n$ ,  $c$ , and  $d$  ( $2 \leq n \leq 500$ ,  $1 \leq c, d \leq 10^6$ ) — the size of the square and the values of  $c$  and  $d$  as described in the statement.

The second line of each test case contains  $n \cdot n$  integers  $b_1, b_2, \dots, b_{n \cdot n}$  ( $1 \leq b_i \leq 10^9$ ) — the elements found by Maxim.

It is guaranteed that the sum of  $n^2$  over all test cases does not exceed  $25 \cdot 10^4$ .

### Output

For each test case, output "YES" in a separate line if a *progressive square* for the given  $n$ ,  $c$ , and  $d$  can be constructed from the array elements  $a$ , otherwise output "NO".

You can output each letter in any case (lowercase or uppercase). For example, the strings "yEs", "yes", "Yes", and "YES" will be accepted as a positive answer.

### Example

input	Copy
5	
3 2 3	
3 9 6 5 7 1 10 4 8	
3 2 3	
3 9 6 5 7 1 11 4 8	
2 100 100	
400 300 400 500	
3 2 3	
3 9 6 6 5 1 11 4 8	
4 4 4	
15 27 7 19 23 23 11 15 7 3 19 23 11 15 11 15	
output	Copy
NO	
YES	

### Codeforces Round 938 (Div. 3)

**Finished**

#### → Practice?

Want to solve the contest problems after the official contest ends? Just register for practice and you will be able to submit solutions.

[Register for practice](#)

#### → Virtual participation

Virtual contest is a way to take part in past contest, as close as possible to participation on time. It is supported only ICPC mode for virtual contests. If you've seen these problems, a virtual contest is not for you - solve these problems in the archive. If you just want to solve some problem from a contest, a virtual contest is not for you - solve this problem in the archive. Never use someone else's code, read the tutorials or communicate with other person during a virtual contest.

[Start virtual contest](#)

#### → Problem tags

[constructive algorithms](#) [data structures](#)  
[implementation](#) [sortings](#)

No tag edit access

#### → Contest materials

- Announcement

YES  
NO  
NO

---

[Codeforces](#) (c) Copyright 2010-2024 Mike Mirzayanov  
The only programming contests Web 2.0 platform  
Server time: Apr/09/2024 08:03:47<sup>UTC+5.5</sup> (h2).  
Desktop version, switch to [mobile version](#).  
[Privacy Policy](#)

Supported by



ITMO UNIVERSITY