C. Adding Powers

time limit per test: 2 seconds
memory limit per test: 256 megabytes

Suppose you are performing the following algorithm. There is an array $v_1, v_2, \ldots, v_n$ filled with zeroes at start. The following operation is applied to the array several times — at i-th step (0-indexed) you can:

- either choose position pos ($1 \le pos \le n$) and increase v_{pos} by k^i;
- or not choose any position and skip this step.

You can choose how the algorithm would behave on each step and when to stop it. The question is: can you make array v equal to the given array a ($v_j = a_j$ for each j) after some step?

Input

The first line contains one integer T ($1 \le T \le 1000$) — the number of test cases. Next $2T$ lines contain test cases — two lines per test case.

The first line of each test case contains two integers n and k ($1 \le n \le 30$, $2 \le k \le 100$) — the size of arrays v and a and value k used in the algorithm.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($0 \le a_i \le 10^{16}$) — the array you'd like to achieve.

Output

For each test case print YES (case insensitive) if you can achieve the array a after some step or NO (case insensitive) otherwise.

Example

input

```
5
4 100
0 0 0 0
1 2
1
3 4
1 4 1
3 2
0 1 3
3 9
0 59049 810
```

output

```
YES
YES
NO
NO
YES
```

Note

In the first test case, you can stop the algorithm before the 0-th step, or don't choose any position several times and stop the algorithm.

In the second test case, you can add k^0 to v_1 and stop the algorithm.

In the third test case, you can't make two 1 in the array v.

In the fifth test case, you can skip 9^0 and 9^1, then add 9^2 and 9^3 to v_3, skip 9^4 and finally, add 9^5 to v_2.