D. Bash and a Tough Math Puzzle

time limit per test: 2.5 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Bash likes playing with arrays. He has an array $a_1, a_2, \ldots a_n$ of n integers. He likes to guess the greatest common divisor (gcd) of different segments of the array. Of course, sometimes the guess is not correct. However, Bash will be satisfied if his guess is *almost correct*.

Suppose he guesses that the gcd of the elements in the range $[l, r]$ of a is x. He considers the guess to be almost correct if he can change **at most** one element in the segment such that the gcd of the segment is x after making the change. Note that when he guesses, he doesn't actually change the array — he just wonders if the gcd of the segment can be made x. Apart from this, he also sometimes makes changes to the array itself.

Since he can't figure it out himself, Bash wants you to tell him which of his guesses are almost correct. Formally, you have to process q queries of one of the following forms:

- $1\ l\ r\ x$ — Bash guesses that the gcd of the range $[l, r]$ is x. Report if this guess is almost correct.
- $2\ i\ y$ — Bash sets a_i to y.

Note: The array is 1-indexed.

Input

The first line contains an integer n $(1 \le n \le 5 \cdot 10^5)$ — the size of the array.

The second line contains n integers $a_1, a_2, \ldots, a_n$ $(1 \le a_i \le 10^9)$ — the elements of the array.

The third line contains an integer q $(1 \le q \le 4 \cdot 10^5)$ — the number of queries.

The next q lines describe the queries and may have one of the following forms:

- $1\ l\ r\ x$ $(1 \le l \le r \le n, 1 \le x \le 10^9)$.
- $2\ i\ y$ $(1 \le i \le n, 1 \le y \le 10^9)$.

Guaranteed, that there is at least one query of first type.

Output

For each query of first type, output "YES" (without quotes) if Bash's guess is almost correct and "NO" (without quotes) otherwise.

Examples

input
3 2 6 3 4 1 1 2 2 1 1 3 3 2 1 9 1 1 3 2

output
YES YES NO

input

```
5
1 2 3 4 5
6
1 1 4 2
2 3 6
1 1 4 2
1 1 5 2
2 5 10
1 1 5 2
```

output

```
NO
YES
NO
YES
```

Note

In the first sample, the array initially is $\{2, 6, 3\}$.

For query 1, the first two numbers already have their gcd as 2.

For query 2, we can achieve a gcd of 3 by changing the first element of the array to 3. Note that the changes made during queries of type 1 are temporary and do not get reflected in the array.

After query 3, the array is now $\{9, 6, 3\}$.

For query 4, no matter which element you change, you cannot get the gcd of the range to be 2.