Codeforces Round #483 (Div. 2) [Thanks, Botan Investments and Victor Shaburov!]

A. Game

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

Two players play a game.

Initially there are $$$n$$$ integers $$$a_1, a_2, \ldots, a_n$$$ written on the board. Each turn a player selects one number and erases it from the board. This continues until there is only one number left on the board, i. e. $$$n - 1$$$ turns are made. The first player makes the first move, then players alternate turns.

The first player wants to minimize the last number that would be left on the board, while the second player wants to maximize it.

You want to know what number will be left on the board after $$$n - 1$$$ turns if both players make optimal moves.

Input
The first line contains one integer $$$n$$$ ($$$1 \le n \le 1000$$$) — the number of numbers on the board.

The second line contains $$$n$$$ integers $$$a_1, a_2, \ldots, a_n$$$ ($$$1 \le a_i \le 10^6$$$).

Output
Print one number that will be left on the board.

Examples

input
3 2 1 3
output
2

input
3 2 2 2
output
2

Note
In the first sample, the first player erases $$$3$$$ and the second erases $$$1$$$. $$$2$$$ is left on the board.

In the second sample, $$$2$$$ is left on the board regardless of the actions of the players.

B. Minesweeper

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

One day Alex decided to remember childhood when computers were not too powerful and lots of people played only default games. Alex enjoyed playing Minesweeper that time. He imagined that he saved world from bombs planted by terrorists, but he rarely won.

Alex has grown up since then, so he easily wins the most difficult levels. This quickly bored him, and he thought: what if the computer gave him invalid fields in the childhood and Alex could not win because of it?

He needs your help to check it.

A Minesweeper field is a rectangle $$$n \times m$$$, where each cell is either empty, or contains a digit from $$$1$$$ to $$$8$$$, or a bomb. The field is valid if for each cell:

- if there is a digit $$$k$$$ in the cell, then exactly $$$k$$$ neighboring cells have bombs.
- if the cell is empty, then all neighboring cells have no bombs.

Two cells are neighbors if they have a common side or a corner (i. e. a cell has at most $$$8$$$ neighboring cells).

Input
The first line contains two integers $$$n$$$ and $$$m$$$ ($$$1 \le n, m \le 100$$$) — the sizes of the field.

The next $$$n$$$ lines contain the description of the field. Each line contains $$$m$$$ characters, each of them is "`.`" (if this cell is empty), "`*`" (if there is bomb in this cell), or a digit from $$$1$$$ to $$$8$$$, inclusive.

Output
Print "`YES`", if the field is valid and "`NO`" otherwise.

You can choose the case (lower or upper) for each letter arbitrarily.

Examples

input
3 3 111 1*1 111
output
YES

input
2 4 *.*. 1211
output
NO

Note
In the second example the answer is "`NO`" because, if the positions of the bombs are preserved, the first line of the field should be `*2*1`.

You can read more about Minesweeper in Wikipedia's article.

C. Finite or not?

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

You are given several queries. Each query consists of three integers $$$p$$$, $$$q$$$ and $$$b$$$. You need to answer whether the result of $$$p/q$$$ in notation with base $$$b$$$ is a finite fraction.

A fraction in notation with base $$$b$$$ is finite if it contains finite number of numerals after the decimal point. It is also possible that a fraction has zero numerals after the decimal point.

Input
The first line contains a single integer $$$n$$$ ($$$1 \le n \le 10^5$$$) — the number of queries.

Next $$$n$$$ lines contain queries, one per line. Each line contains three integers $$$p$$$, $$$q$$$, and $$$b$$$ ($$$0 \le p \le 10^{18}$$$, $$$1 \le q \le 10^{18}$$$, $$$2 \le b \le 10^{18}$$$). All numbers are given in notation with base $$$10$$$.

Output
For each question, in a separate line, print `Finite` if the fraction is finite and `Infinite` otherwise.

Examples

input
2 6 12 10 4 3 10
output
Finite Infinite

input
4 1 1 2 9 36 2 4 12 3 3 5 4
output
Finite Finite Finite

```
Infinite
```

Note

$$$\frac{6}{12} = \frac{1}{2} = 0,5_{10}$$$

$$$\frac{4}{3} = 1,(3)_{10}$$$

$$$\frac{9}{36} = \frac{1}{4} = 0,01_2$$$

$$$\frac{4}{12} = \frac{1}{3} = 0,1_3$$$

D. XOR-pyramid

time limit per test: 2 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

For an array $$$b$$$ of length $$$m$$$ we define the function $$$f$$$ as

$$$ f(b) = \begin{cases} b[1] & \quad \text{if } m = 1 \\ f(b[1] \oplus b[2],b[2] \oplus b[3],\dots,b[m-1] \oplus b[m]) & \quad \text{otherwise,} \end{cases} $$$

where $$$\oplus$$$ is bitwise exclusive OR.

For example, $$$f(1,2,4,8)=f(1\oplus2,2\oplus4,4\oplus8)=f(3,6,12)=f(3\oplus6,6\oplus12)=f(5,10)=f(5\oplus10)=f(15)=15$$$

You are given an array $$$a$$$ and a few queries. Each query is represented as two integers $$$l$$$ and $$$r$$$. The answer is the maximum value of $$$f$$$ on all continuous subsegments of the array $$$a_l, a_{l+1}, \ldots, a_r$$$.

Input

The first line contains a single integer $$$n$$$ ($$$1 \le n \le 5000$$$) — the length of $$$a$$$.

The second line contains $$$n$$$ integers $$$a_1, a_2, \dots, a_n$$$ ($$$0 \le a_i \le 2^{30}-1$$$) — the elements of the array.

The third line contains a single integer $$$q$$$ ($$$1 \le q \le 100\,000$$$) — the number of queries.

Each of the next $$$q$$$ lines contains a query represented as two integers $$$l$$$, $$$r$$$ ($$$1 \le l \le r \le n$$$).

Output

Print $$$q$$$ lines — the answers for the queries.

Examples

input
3 8 4 1 2 2 3 1 2
output
5 12

input
6 1 2 4 8 16 32 4 1 6 2 5 3 4 1 2
output
60 30 12 3

Note

In first sample in both queries the maximum value of the function is reached on the subsegment that is equal to the whole segment.

In second sample, optimal segment for first query are $$$[3,6]$$$, for second query — $$$[2,5]$$$, for third — $$$[3,4]$$$, for fourth — $$$[1,2]$$$.

E. Elevator

time limit per test: 3 seconds
memory limit per test: 256 megabytes

input: standard input
output: standard output

You work in a big office. It is a 9 floor building with an elevator that can accommodate up to 4 people. It is your responsibility to manage this elevator.

Today you are late, so there are queues on some floors already. For each person you know the floor where he currently is and the floor he wants to reach. Also, you know the order in which people came to the elevator.

According to the company's rules, if an employee comes to the elevator earlier than another one, he has to enter the elevator earlier too (even if these employees stay on different floors). Note that the employees are allowed to leave the elevator in arbitrary order.

The elevator has two commands:

- Go up or down one floor. The movement takes 1 second.
- Open the doors on the current floor. During this operation all the employees who have reached their destination get out of the elevator. Then all the employees on the floor get in the elevator in the order they are queued up while it doesn't contradict the company's rules and there is enough space in the elevator. Each employee spends 1 second to get inside and outside the elevator.

Initially the elevator is empty and is located on the floor 1.

You are interested what is the minimum possible time you need to spend to deliver all the employees to their destination. It is not necessary to return the elevator to the floor 1.

Input

The first line contains an integer n ($1 \le n \le 2000$) — the number of employees.

The i-th of the next n lines contains two integers a_i and b_i ($1 \le a_i, b_i \le 9, a_i \ne b_i$) — the floor on which an employee initially is, and the floor he wants to reach.

The employees are given in the order they came to the elevator.

Output

Print a single integer — the minimal possible time in seconds.

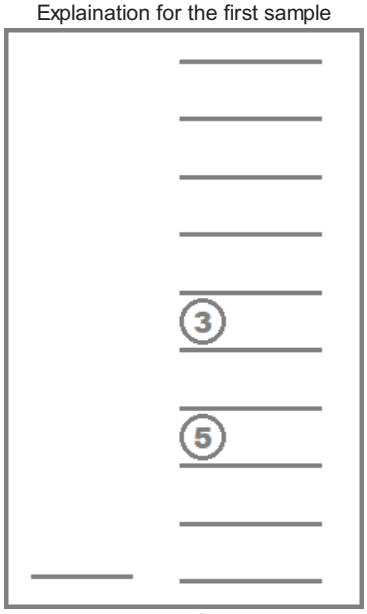
Examples

input
2 3 5 5 3
output
10

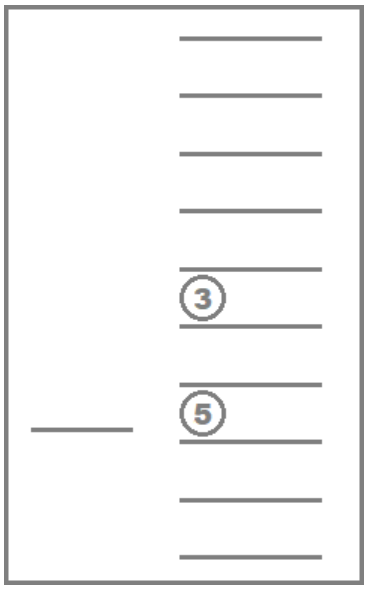
input
2 5 3 3 5
output
12

Note

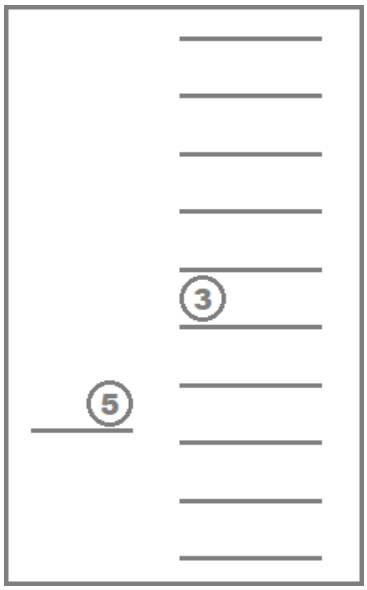
Explaination for the first sample



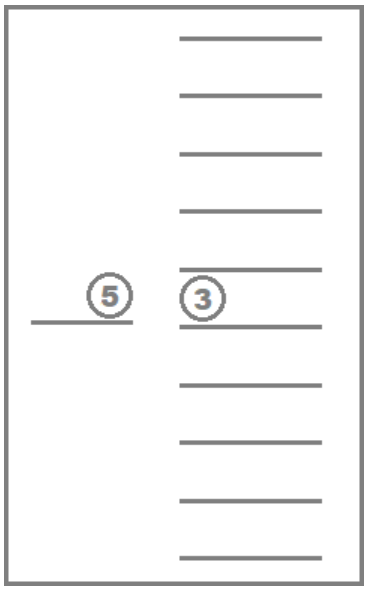
$t = 0$



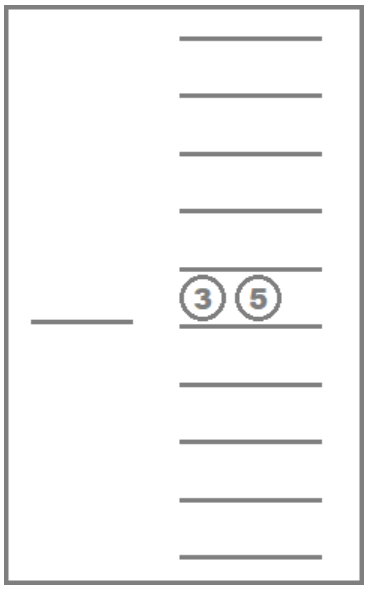
$t = 2$



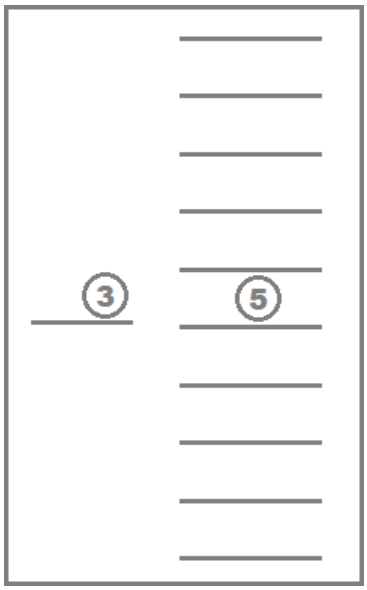
$t = 3$



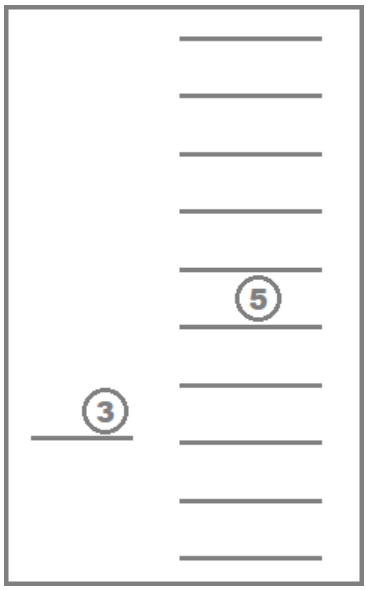
$t = 5$



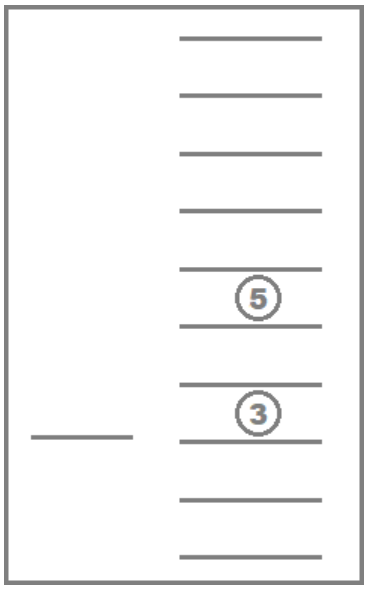
$t = 6$



$t = 7$



$t = 9$



$t = 10$