HKUST Local Contest Fall 2012
September 22, 2012

Contest Time: 1:00pm - 5:30pm

Letter	Time limit	Memory limit	Name
A	2 sec	64MB	Tribonacci Numbers
B	2 sec	64MB	XOR
C	2 sec	64MB	The Sequence
D	60 sec	64MB	Simple Math
E	10 sec	64MB	The Return of the Sequence
F	2 sec	64MB	Penalty Shoot-out

Contest Organizer:
Prof. Ke Yi
Mr. Derek Hao Hu
Mr. Yin Zhu

Problemsetter:
Mr. Derek Hao Hu

Judges:
Mr. Derek Hao Hu
Mr. Naiyan Wang
Mr. Yin Zhu

Contest Rules and Regulations:

1. **This contest is an individual contest. Discussions between contestants are strictly prohibited.** Sanctions will be imposed on contestants if they are found to have violated the regulations governing integrity and honesty.

2. In this contest, **the contestants are given six programming problems**. The goal is to solve as many problems as possible. For those who solve the same number of problems, the one with lower score wins. (The scoring system will be explained below.)

3. **The programming language to be used in this contest is C/C++.** The contestants use PC^2 to submit their source codes to the judge and the source codes are compiled by Visual Studio.

4. **The contestant should read the input and write the output via standard I/O.** The contestants can assume that all test cases are of the format as stated in the problem statements. i.e. No exception handling is needed.

5. The correctness of each submission is judged by inputting test cases into the submitted program. The submission is regarded as correct if its outputs match completely with the model outputs. The submission is judged as correct or wrong. **No partial credit is given.**

6. The contestants can re-submit another source code after previous wrong submissions.

7. **All programs should not run for more than the time limit specified in the problem** (in most cases a "correct" implementation will run far less than the time limit we provide).

8. **The contestants are ranked firstly by the number of problems solved, and secondly the total time spent on solving the problems.** Time spent on solving one problem is the time between the start of contest and the submission of the correct implementation of that problem. For each problem you solved, a penalty of 20 minutes will be added to your score for each wrong submission of that problem.

9. **The contestants are allowed to bring any hard copies of books, notes, references, dictionaries and sketch papers to the contest site.** Electronic devices are forbidden.

Problem A. Tribonacci Numbers

Input file:	`Standard Input`
Output file:	`Standard Output`
Time limit:	2 seconds
Memory limit:	64 megabytes

Tribonacci numbers are extensions of the notorious Fibonacci numbers. In this problem, we study an extension of the Tribonacci numbers by defining the first three elements A_0, A_1 and A_2. The remaining elements of the Tribonacci numbers are calculated by:

$$A_i = A_{i-1} + A_{i-2} + A_{i-3}.$$

In this problem, an array of integers A is provided to you. However, one of the element in this array A is missing and is given to you as -1. You must replace this missing element with a positive number in such a way that the sequence becomes a Tribonacci sequence. If you can find such a positive number, output this number. Otherwise, output -1.

Input

The first line of the input is an integer T, which indicates the number of test cases. T test cases follow. Each test case starts with a number $N (4 \leq N \leq 20)$, indicating the number of elements in the array A. N numbers follow, each element of A will either be -1 or between 1 and 1,000,000, inclusive. **Exactly** one element of A will be -1.

Special note: The constraints for the elements in A do not necessarily apply for the replacement to the missing element.

Output

You should output T lines exactly. Each line contains either the replacement to the missing element such that the original sequence A becomes a Tribonacci sequence, or -1, meaning that the original sequence A cannot become a Tribonacci sequence by replacing the missing element.

Example

Standard Input	Standard Output

5
4 1 2 3 -1
6 10 20 30 60 -1 200
5 1 2 3 5 -1
5 1 1 -1 2 3
4 -1 7 8 1000000
	6
	110
	-1
	-1
	999985

Problem B. XOR

Input file:	`Standard Input`
Output file:	`Standard Output`
Time limit:	2 seconds
Memory limit:	64 megabytes

Given two integers L and R, what is the XOR of all numbers between L and R, inclusive? (XOR is the exclusive-or operation.)

Input

The first line of the input is an integer T, which indicates the number of test cases. T lines follow, each line contains two integers L and R ($1 \leq L, R \leq 4,000,000,000, L \leq R$).

Note that 4,000,000,000 already exceeds the maximum allowed value of int in C++.

Output

You should output T lines exactly. Each line is the value of the XOR of all numbers between L and R.

Example

Standard Input	Standard Output

5
3 10
5 5
13 42
666 1337
1234567 89101112
	8
	5
	39
	0
	89998783

Problem C. The Sequence

Input file:	`Standard Input`
Output file:	`Standard Output`
Time limit:	5 seconds
Memory limit:	64 megabytes

Derek is recently intrigued to a certain type of sequence. In such a sequence, all of its elements are non-negative integers and the product of all these elements is equal to their sum. For example, {1,2,3} is an example of such a sequence.

Now given an arbitrary sequence of length N, Derek wants to know if it is possible to change exactly one element of this sequence so that the resulting sequence is perfect.

Input

The first line of the input is an integer T, which indicates the number of test cases.

T lines follow. Each line contains several integers. The first integer of each line is N, indicating the length of the sequence. The rest of the integers in the line are the elements of the sequence. It is guaranteed that $1 \le N \le 50$ and that each element of the sequence is within $[0, 10^9]$.

Output

You should output T lines exactly. The output of each line is "Yes" or "No", indicating whether it is possible to change exactly one element of the sequence such that the sequence satisfies the requirement mentioned in the problem description.

Example

Standard Input	Standard Output

6
3 1 3 4
3 1 2 3
6 1 4 2 4 2 4
6 1000000 1 1 1 1 2
1 8
3 2 0 2
	Yes
	No
	No
	Yes
	Yes
	No

Problem D. Simple Math

Input file:	`Standard Input`
Output file:	`Standard Output`
Time limit:	60 seconds
Memory limit:	64 megabytes

Given an infinite sequence A, where:

$$A_i = \begin{cases} 0 & \text{if } i \le 0, \\ A_{[i/p]-x} + A_{[i/q]-y} & \text{if } i \ge 1. \end{cases}$$

Now given an integer $n (0 \le n \le 10^{13})$, can you tell me the value of A_n?

(Note: $[x]$ denotes the floor function of x.)

Input

The first line of the input contains an integer T, indicating the number of test cases.

T lines follow, each line five integers n, p, q, x, y. $(0 \le n \le 10^{13}, 2 \le p, q \le 10^9, 0 \le x, y \le 10^9)$.

Output

You should output T lines where each line indicates the value of A_n.

Example

Standard Input	Standard Output

4
10000000 2 3 10000000 10000000
12 2 3 1 0
0 2 2 0 0
123 45 67 8 9
	2
	8
	1
	2

Problem E. The Return of the Sequence

Input file:	`Standard Input`
Output file:	`Standard Output`
Time limit:	10 seconds
Memory limit:	64 megabytes

Derek is recently intrigued to a certain type of sequence......Yes, it's the same story again. **But please pay attention to the problem description**.

In such a sequence, all of its elements are integers and the product of all these elements is equal to their sum. For example, {1,2,3} is an example of such a sequence.

Now given an arbitrary sequence of length N. In one move you can choose one element of the sequence and increase or decrease it by 1. Derek wants to know the minimum number of moves required to make the original sequence satisfy the requirement mentioned above.

Input

The first line of the input is an integer T, which indicates the number of test cases.

T lines follow. Each line contains several integers. The first integer of each line is N, indicating the length of the sequence. The rest of the integers in the line are the elements of the sequence. It is guaranteed that $1 \leq N \leq 50$ and that each element of the sequence is within $[-10^9, 10^9]$.

Output

You should output T lines exactly. The output of each line is an integer, which is the minimum number required.

Example

Standard Input	Standard Output

7
1 -1000000000
2 -1 1
2 4 3
3 17 95 -79
3 10 9 8
3 -2 -29 -13
4 -7 -31 2 -14
	0
	2
	3
	33
	21
	38
	48

Problem F. Penalty Shoot-out

Input file:	`Standard Input`
Output file:	`Standard Output`
Time limit:	2 seconds
Memory limit:	64 megabytes

A penalty shoot-out is a method used in soccer to decide which team progresses to the next stage of a tournament (or wins the tournament) following a tied game. During a penalty shoot-out, each of the teams takes k shots. Each of these shots can either result in scoring a goal or not. The team who scores the most goals during the penalty shoot-out is declared the winner.

To lower the complexity of the problem, if after k shots two teams are still tied, we'll assume that the game ends in a draw.

By analyzing previous data, people find that some teams have a higher probability of scoring goals in penalty shoot-out than other teams. (One notorious example is that Germany national team almost always win penalty shoot-outs and that the England national team almost always loses the penalty shoot-outs).

Winsty, an amateur machine learning researcher, wants to use a probability value p, where $0 < p < 1$, to model the "strength" of each team, which indicates that for a particular team, each penalty shot taken by that team has a probability p to score a goal. For simplicity, we assume all shots are independent.

It is quite hard to model the "strength" of each team accurately by a probability value, so Derek, a professional soccer tipster, has come to help. He has analyzed that team 1 will win with probability $p_1\%$, team 2 will win with probability $p_2\%$ and that two teams will draw with probability $p_{Draw}\%$. Winsty wants to know if there exists two valid probabilities, q_1 and q_2, describing the "strengths" of these two teams, respectively, such that those three outcomes would happen with the probabilities given by Derek? Now he has come to you for help!

Input

The first line of the input contains an integer T, indicating the number of test cases. T test cases follow, each test case contain four integers, p_1, p_{Draw}, p_2, k. It is guaranteed that $0 \le p_1, p_{Draw}, p_2 \le 100$ and that $p_1 + p_{Draw} + p_2 = 100$. k is a positive integer between 1 and 5.

Output

For each test case, output "YES" or "NO", indicating whether such a distribution of outcome is possible.

Example

Standard Input	Standard Output

5
0 100 0 1
50 0 50 1
30 0 70 5
30 10 60 2
30 40 30 2
	YES
	NO
	NO
	NO
	YES