Problem D. XOR of Suffix Sums

Input file:	`standard input`
Output file:	`standard output`
Time limit:	6 seconds
Memory limit:	1024 megabytes

Given an array which is initially empty, you need to perform q operations:

- Given two non-negative integers t and v, take out the element from the end of the array for t times and then append v to the end of the array. It is guaranteed that t does not exceed the length of the array before this operation.

After each operation, let $a_1, a_2, \ldots, a_n$ be the current array, find the bitwise **XOR** of $s_1, s_2, \ldots, s_n$, where $s_i = a_i + a_{i+1} + \ldots + a_n$ is the sum of the suffix starting from position i.

Since the answers may be very large, output them modulo 2 097 152.

The bitwise XOR of non-negative integers A and B, $A \oplus B$ is defined as follows:

- When $A \oplus B$ is written in base two, the digit in the 2^d's place ($d \geq 0$) is 1 if exactly one of the digits in that place of A and B is 1, and 0 otherwise.

For example, we have $3 \oplus 5 = 6$ (in base two: $011 \oplus 101 = 110$).

Generally, the bitwise XOR of k non-negative integers $p_1, p_2, \ldots, p_k$ is defined as

$$(\ldots((p_1 \oplus p_2) \oplus p_3) \oplus \ldots \oplus p_k)$$

and we can prove that this value does not depend on the order of $p_1, p_2, \ldots, p_k$.

Input

The first line contains an integer q ($1 \leq q \leq 5 \times 10^5$), denoting the number of operations.

Each of the following q lines contains two non-negative integers t and v ($0 \leq v \leq 10^9$), describing an operation, where t does not exceed the length of the array before this operation.

Output

Output q lines, each of which contains an integer, denoting the answer.

Examples

standard input	standard output
5 0 1 0 2 1 3 0 6 2 100000	1 1 7 5 1
1 0 1000000000	1755648

Note

After the first operation, the current array is [1], the suffix sum array is [1], and the bitwise XOR of the suffix sums is 1.

After the second operation, the current array is $[1, 2]$, the suffix sum array is $[3, 2]$, and the bitwise XOR of the suffix sums is 1.

After the third operation, the current array is $[1, 3]$, the suffix sum array is $[4, 3]$, and the bitwise XOR of the suffix sums is 7.

After the fourth operation, the current array is $[1, 3, 6]$, the suffix sum array is $[10, 9, 6]$, and the bitwise XOR of the suffix sums is 5.

After the fifth operation, the current array is $[1, 100\,000]$, the suffix sum array is $[100\,001, 100\,000]$, and the bitwise XOR of the suffix sums is 1.