Problem G. Parentheses

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	512 mebibytes

A *correct parentheses sequence* can be defined recursively as follows:

- The empty string is a correct sequence.
- If X and Y are correct sequences, then XY (the concatenation of X and Y) is a correct sequence.
- If X is a correct sequence, then `(`X`)` is a correct sequence.

Each correct parentheses sequence can be derived using the above rules.

For a parentheses sequence, you can make some operations with it.

- Each time you can choose two indices L and R such that $L \le R$. The operation modifies the characters on indices from L to R, inclusive.
- First, the order of these characters is reversed.
- Then, each character is toggled to the opposite one. That is, each '`(`' in the specified range changes to a '`)`' and vice versa.

The *value* of a parentheses sequence is the minimal number of the operations required to change it into a correct parentheses sequence. If it is impossible, the value of the sequence is equal to 10^{100}.

For example, the value of "`()((`" is 1, the value of "`()()`" is 0, and the value of "`(((`" is 10^{100}.

You are given an integer n. For each $1 \le i \le n$, find the number A_i of different parentheses sequence of length n which has value i, and then calculate the sum $\sum_{i=0}^{n}((i+1) \cdot A_i)$.

The answer may be very large, so print it modulo the given integer m.

Input

The first line of the input contains two integers n and m ($1 \le n \le 10^6$, $1 \le m \le 10^9$).

Output

Print one integer: the answer to the problem.

Examples

standard input	standard output
1 100	0
10 100	68