A card pack has an even number $2n$ of cards $a_1, a_2, \ldots, a_{2n}$, all distinct ($a_1 < a_2 < \cdots < a_{2n}$). The pack is initially sorted, that is, the first card in the pack is a_1, the second is a_2, and so on, and the last card in the pack is a_{2n}.

A handler then performs a shuffling procedure repeatedly. The shuffling consists of two steps:

1. the pack is divided in the middle;
2. the cards in the two halves are then interleaved so that if the original sequence at the begining of step 1 is $x_1, x_2, \ldots, x_{2n}$, then at the end of step 2 the sequence of cards becomes $x_{n+1}, x_1, x_{n+2}, x_2, \ldots, x_{2n}, x_n$.

Given the number of cards in the pack, write a program to determine how many times the shuffling procedure must be executed so that the pack becomes sorted again.

Input

The input contains several test cases. A test case consists of one line, which contains an even integer P ($2 \leq P \leq 2 \times 10^5$), where P is the number of cards in the pack (notice that the value P corresponds to the value $2n$ in the description above).

Output

For each test case in the input your program must produce a single line, containing a single integer, the minimum number of times the shuffling procedure must be executed so that the set becomes sorted again.

Sample Input

```
4
6
2
100002
```

Sample Output

```
4
3
2
100002
```