Surely you have made the experience that when too many people use the Internet simultaneously, the net becomes very, very slow.

To put an end to this problem, the University of Ulm has developed a contingency scheme for times of peak load to cut off net access for some cities of the country in a systematic, totally fair manner. Germany's cities were enumerated randomly from 1 to n. Freiburg was number 1, Ulm was number 2, Karlsruhe was number 3, and so on in a purely random order.

Then a number m would be picked at random, and Internet access would first be cut off in city 1 (clearly the fairest starting point) and then in every mth city after that, wrapping around to 1 after n, and ignoring cities already cut off. For example, if $n = 17$ and $m = 5$, net access would be cut off to the cities in the order [1,6,11,16,5,12,2,9,17,10,4,15,14,3,8,13,7]. The problem is that it is clearly fairest to cut off Ulm last (after all, this is where the best programmers come from), so for a given n, the random number m needs to be carefully chosen so that city 2 is the last city selected.

Your job is to write a program that will read in a number of cities n and then determine the smallest integer m that will ensure that Ulm can surf the net while the rest of the country is cut off.

Input

The input file will contain one or more lines, each line containing one integer n with $3 \leq n < 150$, representing the number of cities in the country.

Input is terminated by a value of zero (`0`) for n.

Output

For each line of the input, print one line containing the integer m fulfilling the requirement specified above.

Sample Input

```
3
4
5
6
7
8
9
10
11
12
0
```

Sample Output

```
2
5
2
4
3
11
2
3
8
16
```