CP #73: Roman Roulette

The historian Flavius Josephus relates how, in the Romano-Jewish conflict of 67 A.D., the Romans took the town of Jotapata which he was commanding. Escaping, Jospehus found himself trapped in a cave with 40 companions. The Romans discovered his whereabouts and invited him to surrender, but his companions refused to allow him to do so. He therefore suggested that they kill each other, one by one, the order to be decided by lot. Tradition has it that the means for effecting the lot was to stand in a circle, and, beginning at some point, count round, every third person being killed in turn. The sole survivor of this process was Josephus, who then surrendered to the Romans. Which begs the question: had Josephus previously practised quietly with 41 stones in a dark corner, or had he calculated mathematically that he should adopt the 31st position in order to survive?

Having read an account of this gruesome event you become obsessed with the fear that you will find yourself in a similar situation at some time in the future. In order to prepare yourself for such an eventuality you decide to write a program to run on your hand-held PC which will determine the position that the counting process should start in order to ensure that you will be the sole survivor.

In particular, your program should be able to handle the following variation of the processes described by Josephus. n > 0 people are initially arranged in a circle, facing inwards, and numbered from 1 to n. The numbering from 1 to n proceeds consecutively in a clockwise direction. Your allocated number is 1. Starting with person number i, counting starts in a clockwise direction, until we get to person number k (k > 0), who is promptly killed. We then proceed to count a further k people in a clockwise direction, starting with the person immediately to the left of the victim. The person number k so selected has the job of burying the victim, and then returning to the position in the circle that the victim had previously occupied. Counting then proceeeds from the person to his immediate left, with the k-th person being killed, and so on, until only one person remains.

For example, when n = 5, and k = 2, and i = 1, the order of execution is 2, 5, 3, and 1. The survivor is 4.

Input Format

Your program must read input lines containing values for n and k (in that order. Input will be terminated by a line containing values of ‘0’ for n and k.

Your program may assume a maximum of 100 people taking part in this event.

Output Format

For each input line output the number of the person with which the counting should begin in order to ensure that you are the sole survivor. For example, in the above case the safe starting position is 3.

Sample Input 0

```
5 2
1 5
0 0
```

Sample Output 0

```
3
1
```