

3343 Last Digits

Exponentiation of one integer by another often produces very large results. In this problem, we will compute a function based on repeated exponentiation, but output only the last n digits of the result. Doing this efficiently requires careful thought about how to avoid computing the full answer.

Given integers b , n , and i , we define the function $f(x)$ recursively by $f(x) = b^{f(x-1)}$ if $x > 0$, and $f(0) = 1$. Your job is to efficiently compute the last n decimal digits of $f(i)$.

Input

The input consists of a number of test cases. Each test case starts with the integer b ($1 \leq b \leq 100$) called the **base**. On the next line is the integer i ($1 \leq i \leq 100$) called the **iteration count**. And finally, the last line contains the number n ($1 \leq n \leq 7$), which is the number of decimal digits to output.

The input is terminated when $b = 0$.

Output

For each test case, print on one line the last n digits of $f(i)$ for the base b specified. If the result has fewer than n digits, pad the result with zeroes on the left so that there are exactly n digits.

Sample Input

```
2
4
7
10
10
6
3
10
7
0
```

Sample Output

```
0065536
000000
4195387
```