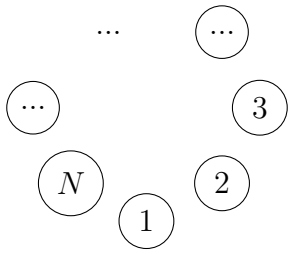
Das Blinkenlights

(2 points.) To start, run `python new-homework.py [language] das-blinkenlights`

Background. Unhappy with the dim lighting in his barn, Farmer John has just installed a fancy new chandelier consisting of N light bulbs arranged in a circle:



N may be in the range $3 \leq N \leq 16$. The cows are fascinated by this new light fixture, and enjoy playing the following game:[1] at time T, they toggle the state of each light bulb if its neighbor to the left was turned on at time $T-1$. (Define $T = 0$ to be the state the bulbs start in.) They continue this game for B units of time ($1 \leq B \leq 10^{15}$). Note that B might be very large, and note that because the bulbs are in a circle, bulb N is to the left of bulb 1.

Task. Write a function `blink(states, B)` which, given the initial states of the light bulbs, determines their final states after `B` units of time have elapsed. `states` should be a length-N vector or array of 0s and 1s indicating which light bulbs are turned on.

Notice that `B` can be very, very large, so the naive implementation of `blink` (simply update the lights `B` times) will be impractically slow. After implementing the naive strategy, devise a shortcut which takes advantage of the structure of the problem. If you cannot think of one, hints are provided below.

Examples. If `states` is `1 0 0`, the sequence of states, for each `B`, is:

1. `1 1 0`
2. `1 0 1`
3. `0 1 1`
4. `1 1 0`

and so on.

Requirements.

☐ Implement `blink` and write a series of test cases. (A good starting configuration is `1 0 0 0 0`.) Make your code modular, so the cow's strategy for deciding which lights to turn on could easily be replaced (e.g., by checking bulbs to the right instead of bulbs to the left).

[1] Farmer John didn't spring for cable TV.

- ☐ Implement a shortcut strategy. Carefully comment your code to explain your strategy. You should be able to handle *B* of 10^{15} in a few seconds.
- ☐ Write tests for your shortcut strategy, including a randomized test which compares the output of the shortcut strategy to the naive, slow version of *blink* across a range of random inputs. (Be sure that it is easy to get the input values used if the test should fail, so you can debug test failures.)
- ☐ Write a command-line wrapper script that calls *blink* with a bulb configuration specified on the command line and prints the results. An example call might be `./blink -B=1000 1 0 0 0 0` or `Rscript blink.R 1000 1 0 0 0 0`.

Hint 1. What happens if the bulb state becomes all zeros (i.e. all the bulbs are turned off)?

Hint 2. Notice that there are only 2^N possible bulb configurations, and $2^N \ll 10^{15}$. Because the cows toggle the switches deterministically, the bulb configurations must repeat within 2^N iterations.