Code Jam to I/O for Women 2016 - Code Jam to I/O for Women

Analysis: Dance Around The Clock

The Small dataset's limits are small enough that you can create an array of dancers and simulate each turn, and then find dancer **K** and check their neighbors. The circular arrangement of the dancers can make this tricky. You might find the modulo operator useful to avoid special-casing the edges of the array.

In the Large dataset, a 100 million turn simulation with 100 million dancers is unlikely to finish in time, so we need to make some simplifying observations about the dance. At first, the dance might seem intricate, but it follows a very regular pattern. Observe what happens for **D** = 6, **N** = 7. (These lists start from the 12:00 position, and go in clockwise order.)

1 2 3 4 5 6
2 1 4 3 6 5
5 4 1 6 3 2
4 5 6 1 2 3
3 6 5 2 1 4
6 3 2 5 4 1
1 2 3 4 5 6
2 1 4 3 6 5

Some things to notice:

- The list repeats itself after **D** turns, and will continue to repeat every **D** turns.
- The odd-numbered dancers move clockwise together as a set, one position per turn, always keeping their same relative order. The even-numbered dancers move counterclockwise together as a set, one position per turn, always keeping their same relative order.

Let's follow dancer 3 through those turns. The left neighbors of dancer 3, after 0, 1, 2... turns, are 2, 4, 6, 2, 4, 6, 2, 4. The right neighbors of dancer 3 are 4, 6, 2, 4, 6, 2, 4, 6. Note that on every turn, the numbers of both neighbors go up by 2 (looping around once the maximum value of 6 is reached), and the right neighbor's number is always 2 larger than the left neighbor's number. This makes sense — odd-numbered dancers are always moving one position to the right as the set of even-numbered dancers moves one position past them to the left, so they successively end up between every pair of even-numbered dancers. The cycle of neighbor numbers repeats every **D** / 2 turns.

So, for an odd-numbered dancer numbered **K** out of **D**, the formulas for the left neighbor *L* and the right neighbor *R* after **N** turns are:

L = ((**K** - 2 + 2 * **N**) % **D**) + 1
R = ((**K** + 2 * **N**) % **D**) + 1

The dancers are numbered from 1 through **N**, but the modulo operator produces values between 0 and **N** - 1. The formulas above effectively transform the dancer numbers into that system and then transform it back. Alternatively, you may find it more convenient in your program to subtract 1 from every dancer's number upon reading the data, and then add 1 just before outputting the answer. Whichever strategy you use, it's worth checking a couple examples with dancers with the smallest and largest possible odd numbers.

We leave the very similar corresponding formulas for even-numbered dancers as an exercise.

The approach we've outlined here — writing a brute force Small solution, inspecting the process or output, and deriving some insights that are helpful for the Large dataset — can be very fruitful in Code Jam, as we've mentioned in our essay on Small datasets.