**Thinking**

1. Build the actual device in code. This would mean that I would be calculating every single step along the way until the balls return to the proper order. Not very efficient.
2. Instead of calculating everything, I can calculate the first 12 hours and use a transformation template that maps the movement of all the balls after each 12 hour period. This would be significantly faster but still seems like there may be another way to do it. I would probably use an array/map with the balls in the positions after the 12 hours.
3. Instead of repeating the movement for each 12 hour period, try calculating cycles for each ball to create a position map for each ball. The idea is that this would work for groups of balls and will limit the number of times I need to calculate things. Build pattern maps until all the numbers are part of a map. Track the number of positions in a map each time. Once all the numbers have been placed in a map, find the lowest common multiple of all the maps’ number of positions. Keep in mind: the actual maps aren’t important. The only thing that matters is the number of positions in a map and that each number belongs to a map. Leave out the details to save memory.