--- Day 16: Dragon Checksum ---

You're done scanning this part of the network, but you've left traces of your presence. You need to overwrite some disks with random-looking data to cover your tracks and update the local security system with a new checksum for those disks.

For the data to not be suspiscious, it needs to have certain properties; purely random data will be detected as tampering. To generate appropriate random data, you'll need to use a modified dragon curve.

Start with an appropriate initial state (your puzzle input). Then, so long as you don't have enough data yet to fill the disk, repeat the following steps:

- Call the data you have at this point "a".
- Make a copy of "a"; call this copy "b".
- Reverse the order of the characters in "b".
- In "b", replace all instances of `0` with `1` and all `1`s with `0`.
- The resulting data is "a", then a single `0`, then "b".

For example, after a single step of this process,

- `1` becomes `100`.
- `0` becomes `001`.
- `11111` becomes `11111000000`.
- `111100001010` becomes `1111000010100101011110000`.

Repeat these steps until you have enough data to fill the desired disk.

Once the data has been generated, you also need to create a checksum of that data. Calculate the checksum only for the data that fits on the disk, even if you generated more data than that in the previous step.

The checksum for some given data is created by considering each non-overlapping pair of characters in the input data. If the two characters match (`00` or `11`), the next checksum character is a `1`. If the characters do not match (`01` or `10`), the next checksum character is a `0`. This should produce a new string which is exactly half as long as the original. If the length of the checksum is even, repeat the process until you end up with a checksum with an odd length.

For example, suppose we want to fill a disk of length `12`, and when we finally generate a string of at least length `12`, the first `12` characters are `110010110100`. To generate its checksum:

- Consider each pair: `11`, `00`, `10`, `11`, `01`, `00`.

- These are same, same, different, same, different, same, producing `110101`.
- The resulting string has length `6`, which is even, so we repeat the process.
- The pairs are `11` (same), `01` (different), `01` (different).
- This produces the checksum `100`, which has an odd length, so we stop.

Therefore, the checksum for `110010110100` is `100`.

Combining all of these steps together, suppose you want to fill a disk of length `20` using an initial state of `10000`:

- Because `10000` is too short, we first use the modified dragon curve to make it longer.
- After one round, it becomes `10000011110` (`11` characters), still too short.
- After two rounds, it becomes `10000011110010000111110` (`23` characters), which is enough.
- Since we only need `20`, but we have `23`, we get rid of all but the first `20` characters: `10000011110010000111`.
- Next, we start calculating the checksum; after one round, we have `0111110101`, which `10` characters long (even), so we continue.
- After two rounds, we have `01100`, which is `5` characters long (odd), so we are done.

In this example, the correct checksum would therefore be `01100`.

The first disk you have to fill has length `272`. Using the initial state in your puzzle input, what is the correct checksum?

Your puzzle answer was `11101010111100010`.

--- Part Two ---

The second disk you have to fill has length `35651584`. Again using the initial state in your puzzle input, what is the correct checksum for this disk?

Your puzzle answer was `01001101001000101`.

Both parts of this puzzle are complete! They provide two gold stars: **

At this point, you should return to your advent calendar and try another puzzle.

Your puzzle input was `10111011111001111`.

You can also [Share] this puzzle.