Advent of Code - Day10

Hilmir Vilberg Arnarsson

December 10, 2023

Part 1

Part 1 is self-explanatory.

Part 2

Part 2 is very tricky. Initially I tried flood-filling but this did not work since there are some weird edge cases where something is outside but has no neighbors. One could probably expand the grid somehow to get rid of those so you kind of "squeeze" between pipes, but that seemed very tricky. Inspiration is taken from this. The solution is very elegant but not very intutive at first glance.

Let us introduce the notion of *intersections*. If we start from the left in a row, then we must be outside. This is because an edge cannot possibly be inside the loop on the edge. If we look at the simple case where there are no bendy pieces, only walls, i.e., "-" or "|", then it should be apparent that if we cross a wall once then we are inside the loop if the wall is part of the loop. If we then cross another wall we will be outside of the loop. Thus if we increase the intersections by one each time we cross a wall, then we will be inside if the intersections is odd.

The complications arise when we consider the bendy pieces. It is not simply enough to increase the intersections when we cross one due to portions like this: "F–7" or "L–J". However, we should increase the intersections if we cross a section like this "F–J". The trick is to select one of the cases and increase the intersections in that case, i.e., if we cross an "F" or a "7" we increase the intersections. If we increase it with an "F", then our intersections will only be odd if we cross a "J" or "L" next. If we cross a "7" next, then we increase it again and it will be even again. Super elegant. Note that we only select one of either "F" and "7" or "L" and "J".