CS2500 Fall 2023 - Fundamentals of Computer Science (Fundies) 1

Homework 11: Pipe Fantasy (Part 3)

This is the final part of Pipe Fantasy, where you will turn it into a highly addictive game.

Before You Start ...

Here are a few things to keep in mind before you start working on it:

1. Everything we said about Part 1 and Part 2 still applies!
2. This assignment will ask you to build on top of your solution to Part 2. However, if you have not completed Part 2 or are not happy with your solution, you may use our solution for Part 2.
3. As usual, you do not have to write tests for any functions that returns an image.

Cross Pipe Painting

Task 1: Modify your program so that the cross pipe only paints the goo in the direction that it is flowing.

Suggestion: your `pipe->image` function may need to take an extra argument that represents the direction of goo flow.

Scoring

The goal of the game is to have a longest goo path. The score is computed using the following formula:

```
(* 50 (- length-path num-replaced))
```

Above, `length-path` is the length of the path of the goo in number of tiles, and `num-replaced` is the number of pipes that the user has replaced. However, note that goo can pass through a cross-pipe twice!

Task 2: Write a function `get-score: GameState -> Integer` that computes the current score of the game.

Hint: ~~It should be possible to write this function without making any changes to the data definitions that you already have.~~ You may want to add a field to `GameState` that keeps track of the number of pipes that the user has replaced.

Task 3: Display the current score while the game is running.

Protecting Pipes With Goo

In the last part, you implemented goo flow propagation after the user had placed all pipes. Now, goo will flow while pipes are being placed. So, you should make sure that the user can not place an incoming pipe on a pipe that already has goo in it.

Task 4: Modify your `place-pipe-on-click` function to implement this feature: if the user tries to place an incoming pipe on a pipe that already has goo in it, then the pipe should not be placed.

Automatic Goo Propagation

The goo should start flowing soon after the game begins, and then continue to propagate cell-by-cell on a timer.

Task 5: Add a field to `GameState` that counts the time until the next step of goo propagation. Update `gamestate-init` appropriately so that the goo starts flowing 140 ticks (5 seconds) after the game begins.

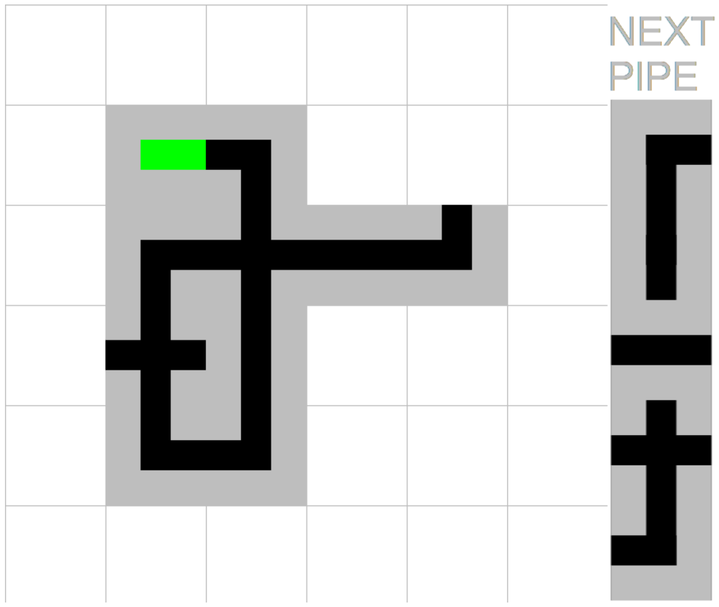
Task 6: Write an `on-tick` handler for the `big-bang` that does the following:

1. If there is time left before the next goo propagation, then count down the time by 1 tick.
2. If the countdown timer reaches 0, then propagate the goo by one cell and reset the countdown timer to 28 ticks (1 second).
3. If the goo cannot propagate anymore, then the function should not change the `GameState`.

Note: Feel free to set the ticks value to some other value during testing.

Game Demo

Task 7: In a comment at the top of your file, tell us what to run to see your game in action with the following scenario:



We recommend that you define a gamestate constant for this scenario. Your set of incoming pipes should be the same as the one we provided. (However, if you choose to display less than 5 incoming pipes in your part 2, you do not have to modify your code to display 5 incoming pipes for this part.)