draw-tail

Setup

Right now you should have a head that moves correctly.

1. Add the following to your code: posn(225, 175)

```
t2 = posn(175, 175)
t1 = posn(125, 175)
t0 = posn(75, 175)

L = [list: t0, t1, t2]
```

2. Then change “`start`” to:

```
start = snake(posn(225, 175), posn(50, 0), L)
```

Run the reactor. See anything different?

To draw the snake blocks with put-image, you could use this command:

```
put-image(snake-head, t0.x, t0.y,
      put-image(snake-head, t1.x, t1.y,
            put-image(snake-head, t2.x, t2.y, bg)))
```

3. Try it and see that it works!

4. What would we need to do if we had one more square to the left of `t0`?

Function Design

5. Contract:

6. Purpose - restate the contract in your own words.

7. Examples:

Fill in the blanks:

```
draw-tail(L)               is   put-image(snake-head, t0.x, t0.y,
                                      put-image(snake-head, t1.x, t1.y,
                                            put-image(snake-head, t2.x, t2.y,
                                bg)))


draw-tail( [list: t0,      is   put-image(snake-head, t0.x, t0.y,
t1, t2])                              put-image(snake-head, t1.x, t1.y,
                                            put-image(snake-head, t2.x, t2.y,
                                bg)))

draw-tail( [list: t1,      is
t2])


draw-tail( [list: t2])     is


draw-tail( [list: ])       is
```

8. Function

For each example, on the left side, circle the first element in the list and label it `f`.

Then, on the left side, circle the rest of the list and label it `r`.

For each example, on the right side, can you find the expression for `draw-tail(r)`? Circle and label it.

What happens in the `empty` case?

9. Fill in the blanks:

```
fun draw-tail(tail-list):
      cases (List) tail-list:
      |empty => ______________________

      |link(f, r) => ____________________________
end
```

Implement

10. Your draw-tail brings in a list and outputs an image. How will you get the list from the game state? You will need to do this because `draw-state` brings in the data from the whole game.

11. Now you have the tail drawn on the background, change the `draw-state` function so that the head is drawn on top of this image.

12. You can use a different square for the tail. In draw-tail, before the `cases` line, define `tail-square` to be the color and outline of your liking, and use that in the `link(f, r)` statement instead.

Our next step is to detect the head crashing into a wall or the tail on its next move, and then to make the tail move with the head. After that I will give you a mechanic for placing food, and have your program grow each time it grabs a “pellet!”

Then you