Snakes on Bikes

Project 4: PyGame

Eric Elder Jacobsen & Luis Zuniga

Main Idea: Our project is a combination of the games “Snake” and “Tron,” using PyGame. The idea is that there would be two players, each controlling one bike. They try to cut each other off with their trail, like in Tron, but the trail is of limited length and must be grown by collecting food (or whatever the pickups are called). The most basic version of the game would be just the snake-bikes, but more advanced versions could incorporate different kinds of pickup to add complexity: maybe something that makes the player faster or slower, something that adds or subtracts extra length, or anything else. This will involve shapes and animations in PyGame, as well as things like collision detection, both for crashes and for picking up food. It will also involve user input, most likely from the arrows and WASD keys to steer the two snake-bikes. There probably won’t be any libraries involved other than PyGame.

Individual Learning Goals:

Eric:
I have never used PyGame, or made an animations-centric game like this. I would like to work with time events to learn how to do that, and to make the graphics of the game look crisp and intentional. Since this is the first partner project of the class, I also want to have a good team-programming plan. I’m fine with splitting up the work for some of the more basic code, but for the main parts of the program I would like to do partner programming, with two people at the same computer.

Luis:
I anticipate that creating objects in PyGame will be quite different. Now that we’re moving into graphical code, I’d like to learn and adapt to the new landscape. Learning to pair with a partner will probably be interesting, and I look forward to it.

Mid-Project Goal: By the midpoint of the project, we want to have a snake-bike that moves with input from the arrow keys. It will not necessarily have collision detection or food pickups by that stage. We anticipate that input and movement will be about half of the work for this project.

Biggest risks: Having code that appears to work, but doesn’t work the way we thought it would. Having parts of the code (like new classes) that work separately, but cannot communicate with each other. All the new syntax and commands that come with the PyGame library. Coding from two computers without merge conflicts.