This is CS50x

OpenCourseWare

David J. Malan (https://cs.harvard.edu/malan/)
malan@harvard.edu
(https://www.facebook.com/dmalan) (https://github.com/dmalan) (https://www.instagram.com/davidjmalan/)
(https://www.linkedin.com/in/malan/) (https://www.quora.com/profile/David-J-Malan) (https://twitter.com/davidjmalan)

Pong: The AI Update

Objectives

- Download LÖVE.
- Read and understand all of the Pong source code from the last track lesson.
- Implement a basic AI for either Player 1 or 2 (or both!).

Distribution Code

Download this project's distribution code (https://cdn.cs50.net/2019/fall/tracks/games/pong/pong.zip).

It's Game Time

Your first assignment in this track will be a fairly easy one, since the dive into game programming can be deep enough as it is without having to implement an entire code base from scratch! Instead, we'll take the Pong example we covered and extend it in a small but fun way by giving one of the paddles (or perhaps both!) logic for playing the game so that you don't always need a buddy to play the game with you!

Of course, the code won't run if you don't have the LÖVE framework installed, so we'll have to tackle that in addition to grabbing the code; choose the version for your system here:

Download LÖVE (https://love2d.org/)

For further information on how to actually run games, please visit the following page:

https://love2d.org/wiki/Getting_Started (https://love2d.org/wiki/Getting_Started)

Additionally, if you're new to the Lua programming language (what LÖVE expects you to write in order to work), check out the following online reference manual:

Lua Programming Manual (https://www.lua.org/manual/5.3/)

Once the code and LÖVE have been downloaded and installed, the actual change you'll be making to the code base is small, but it will require you to understand what many of the pieces do, so be sure to watch each of the track's lessons and read through the code so you have a firm understanding of how it works before diving in! In particular, take note of how paddle movement works, reading both the `Paddle` class as well as the code in `main.lua` that actually drives the movement, located in the `update` function (currently done using keyboard input for each). If our agent's goal is just to deflect the ball back toward the player, what needs to drive its movement?

Your goal:

- Implement an AI-controlled paddle (either the left or the right will do) such that it will try to deflect the ball at all times. Since the paddle can move on only one axis (the Y axis), you will need to determine how to keep the paddle moving in relation to the ball. Currently, each paddle has its own chunk of code where input is detected by the keyboard; this feels like an excellent place to put the code we need! Once either the left or right paddle (or both, if desired) try to deflect the paddle on their own, you've done it!

How to Submit

To submit your code with `submit50`, you may either: (1) upload your code to CS50 IDE and run `submit50` from inside of your IDE, or (2) install

`submit50` on your own computer by running `pip3 install submit50` (assuming you have Python 3 (https://www.python.org/downloads/) installed).

Execute the below, logging in with your GitHub username and password when prompted. For security, you'll see asterisks (`*`) instead of the actual characters in your password.

```
submit50 cs50/problems/2020/x/tracks/games/pong
```