(https://courserahelp.zendesk.com/hc/en-us/requests/new)

Peer Assessments (https://class.coursera.org/interactivepython-005/human_grading/)
/ Mini-project # 4 - "Pong"
Help (https://class.coursera.org/interactivepython-005/help/peergrading?url=https%3A%2F%2Fclass.coursera.org%2Finteractivepython-005%2Fhuman_grading%2Fview%2Fcourses%2F972530%2Fassessments%2F31%2Fsubmissions)

due in 3day 16h

Submission Phase

1. Do assignment ☐ (/interactivepython-005/human_grading/view/courses/972530/assessments/31/submissions)

Evaluation Phase

2. Evaluate peers 🔒 (/interactivepython-005/human_grading/view/courses/972530/assessments/31/peerGradingSets)
3. Self-evaluate 🔒 (/interactivepython-005/human_grading/view/courses/972530/assessments/31/selfGradingSets)

Results Phase

4. See results 🔒 (/interactivepython-005/human_grading/view/courses/972530/assessments/31/results/mine)

☐ In accordance with the Honor Code, I certify that my answers here are my own work, and that I have appropriately acknowledged all external sources (if any) that were used in this work.

Save draft Submit for grading

A reminder about the Honor Code

For previous mini-projects, we have had instances of students submitting solutions that have been copied from the web. Remember, if you can find code on the web for one of the mini-projects, we can also find that code. Submitting copied code violates the Honor Code (../../../../../../wiki/view?page=honorcode) for this class as well as Coursera's Terms of Service. Please write your own code and refrain from copying. If, during peer evaluation, you suspect a submitted mini-project includes copied code, please evaluate as usual and email the assignment details to iipphonorcode@online.rice.edu (mailto:iipphonorcode@online.rice.edu). We will investigate and handle as appropriate.

Mini-project #4 - "Pong"

In this project, we will build a version of Pong (http://en.wikipedia.org/wiki/Pong), one of the first arcade video games (1972). While Pong is not particularly exciting compared to today's video games, Pong is relatively simple to build and provides a nice opportunity to work on the skills that you will need to build a game like *Asteroids*. As usual, we have provided a **program template** (http://www.codeskulptor.org/#examples-pong_template.py) that can be used to guide your development of Pong.

Mini-project development process

1. Add code to the program template that draws a ball moving across the Pong table. We recommend that you add the positional update for the ball to the draw handler as shown in the second part of the "Motion" video.

2. Add code to the function `spawn_ball` that spawns a ball in the middle of the table and assigns the ball a fixed velocity (for now). Ignore the parameter `direction` at this point.
3. Add a call to `spawn_ball` in the function `new_game` which starts a game of Pong. Note that the program templates includes an initial call to `new_game` in the main body of your program to get a game going immediately.
4. Modify your code such that the ball collides with and bounces off of the top and bottom walls. Experiment with different hard-coded initial velocities to test your code.
5. Add randomization to the velocity in `spawn_ball(direction)` The velocity of the ball should be upwards and towards the right if `direction == RIGHT` and upwards and towards the left if `direction == LEFT`. The exact values for the horizontal and vertical components of this velocity should be generated using `random.randrange()`. For the horizontal velocity, we suggest a speed of around `random.randrange(120, 240)` pixels per second. For the vertical velocity, we suggest a speed of around `random.randrange(60, 180)` pixels per second. (You will need to set the signs of velocities appropriately.)
6. Add code to the draw handler that tests whether the ball touches/collides with the left and right gutters. (Remember that the gutters are offset from the left and right edges of the canvas by the width of the paddle as described in the "Pong" video.) When the ball touches a gutter, use either `spawn_ball(LEFT)` or `spawn_ball(RIGHT)` to respawn the ball in the center of the table headed towards the opposite gutter.
7. Next, add code that draws the left and right paddles in their respective gutters. The vertical positions of these two paddles should depend on two global variables. (In the template, the variables were `paddle1_pos` and `paddle2_pos`.)
8. Add code that modifies the values of these vertical positions via an update in the draw handler. The update should reference two global variables that contain the vertical velocities of the paddles. (In the template, the variables were `paddle1_vel` and `paddle2_vel`.)
9. Update the values of these two vertical velocities using key handlers. The "w" and "s" keys should control the vertical velocity of the left paddle while the "Up arrow" and "Down arrow" key should control the velocity of the right paddle. In our version of Pong, the left paddle moves up at a constant velocity if the "w" key is pressed and moves down at a constant velocity if the "s" is pressed and is motionless if neither is pressed. (The motion if both are pressed is up to you.) To achieve this effect, you will need to use both a keydown and a keyup handler to increase/decrease the vertical velocity in an appropriate manner.
10. Restrict your paddles to stay entirely on the canvas by adding a check before you update the paddles' vertical positions in the draw handler. In particular, test whether the current update for a paddle's position will move part of the paddle off of the screen. If it does, don't allow the update.
11. Modify your collision code for the left and right gutters in step 6 to check whether the ball is actually striking a paddle when it touches a gutter. If so, reflect the ball back into play. This collision model eliminates the possibility of the ball striking the edge of the paddle and greatly simplifies your collision/reflection code.
12. To moderately increase the difficulty of your game, increase the velocity of the ball by 10% each time it strikes a paddle.
13. Add scoring to the game as shown in the Pong video lecture. Each time the ball strikes the left or right gutter (but not a paddle), the opposite player receives a point and ball is respawned appropriately.
14. Finally, add code to `new_game` which resets the score before calling `spawn_ball`. Add a "Restart" button that calls `new_game` to reset the score and relaunch the ball.

Your final version of Pong should be remarkably similar to the original arcade Pong game. Our full implementation of Pong took a little more than 100 lines of code with comments.

Grading Rubric - 19 pts total (scaled to 100 pts)

- 1 pt - The ball spawns in the middle of the canvas with either an upward left or an upward right velocity. No credit if the ball moves only horizontally left or right. Bleh, that would be boring!
- 2 pts - The ball bounces off of the top and bottom walls correctly. (1 pt each)
- 2 pts - The ball respawns in the middle of the screen when it strikes the left or right gutter but not the paddles. (1 pt for each side) Give credit for this item even if the ball hits the edge of the canvas instead of the gutter.
- 1 pt - The left and right gutters (instead of the edges of the canvas) are properly used as the edges of the table.
- 1 pt - The ball spawns moving towards the player that won the last point.
- 2 pts - The 'w' and 's' keys correctly control the velocity of the left paddle as described above. Please test each key in isolation. (1 pt if the paddle moves, but in an incorrect manner in response to 'w' and 's' key presses.)

- 2 pts - The up and down arrows keys correctly control the velocity of the right paddle as described above. Please test each key in isolation. (1 pt if the paddle moves, but in an incorrect manner in response to up and down arrow key presses.)
- 2 pts - The edge of each paddle is flush with the gutter. (1 pt per paddle)
- 2 pts - The paddles stay on the canvas at all times. (1 pt per paddle)
- 2 pts - The ball correctly bounces off the left and right paddles. (1 pt per paddle)
- 1 pt - The scoring text is positioned and updated appropriately. The positioning need only approximate that in the video.
- 1 pt - The game includes a "Restart" button that resets the score and relaunches the ball.

In the submission phase, cut and paste the URL for your cloud-saved mini-project into the box below. Click the Honor Code box and hit the "Submit for grading" button when you are ready to submit your mini-project. (You may submit as many times as you like before the deadline so we suggest that you use "Submit" instead of "Save".) **IMPORTANT: Please use the "Review your work" link that appears at the top of the subsequent submission page to verify that you submitted a working link for the final version of your mini-project.**

☐ In accordance with the Honor Code, I certify that my answers here are my own work, and that I have appropriately acknowledged all external sources (if any) that were used in this work.

Save draft   Submit for grading