**Kaivan Taylor – CSC 201 – Professor Seaman**

**CSC 201 – Program Assignment Submission Sheet**

**1. Problem Statement**

Write a GUI program to create a game where a ball moves in any direction on a canvas and “bounces” around. When the disk hits something, it bounces back with an angle of reflection the same as the angle of incidence, but is incremented by a slight bit of positive, or negative randomness. The paddle is offset from the edge and a target slot at the center of the other end. The paddle must be no more than one-fourth the height of the field. The player moves the paddle using up and down arrows. The paddle cannot move left and right. If the ball hits the paddle, it bounces back. If the ball hits the slot, the player wins and it is announced.

**2. Input/Output Description**

Up/Down Arrows for Paddle 🡪 COMPUTER 🡪 Ball animated bouncing against walls/paddle until in goal and text announcing goal

**3. Hand-Worked Examples** (at least two; also to be used as tests, below, in Step 6)

**4. Algorithm Development (Decomposition Outline)**

1. Import all tkinter libraries.
2. Import random library for random integers for random angles.
3. Create a window/root variable for Tk().
4. Create a title for the root.
5. Define variables for width/height of root, sleep timer, and ismoving Boolean (set to True to run game).
6. Create a Canvas.
7. Create a static line in the middle of canvas for mid-line representation.
8. Define a variable for the radius of the ball.
9. Create a ball in using the variable for radius.
10. Create the paddle.
11. Bind arrow keys for up and down to a method that will move the paddle’s dy coordinate up and down with up and down arrows respectively.
12. Create a method that will animate the ball object by drawing it and updating the canvas.
13. Set a limit for the speed of the ball by limiting the dx increment.
14. If the limit is over, set the speed back to a value within the speed limit.
15. For the boundaries, if the ball touches the left, right, top, bottom wall, create a function that will reverse its direction opposite of each respective wall for dy/dx.
16. Test to see if the ball will stay within the window and maintains a constant speed.
17. If the ball hits the paddle, make the ball “bounce”, or reflect off of the surface of the paddle with a small change in angle using random.
18. Test to see if the ball “makes contact” with the paddle, and reflects off the paddle instead of going through the paddle.
19. Bounce the ball back off the paddle with a small increase/decrease of the angle using random.
20. If the ball is within the coordinates of the goal, print out of the user “goal”.
21. Test the coordinates that the print statement triggers and fine tune it to the correct coordinates that “match” within the goal box.
22. Call the game.

**5. Program Source Code**  *Attach* sheet(s) of your source code printed from the IDE. (Do not *include* code here!)

**6. Test(s)** Include here or attach sheet(s) of “cut-and-pasted” results from the console window and printed from MS Word using a fixed-width font. Also, remove excessive vertical whitespace to save paper.

