MODEL

Class Ball{
Radius
X
Y
Color
Speed_x
Speed_y

move(self)
Check_wall_collision : checks to see if the ball hits a wall
Check_paddle_collision : checks to see if the ball hits a paddle
}

Class Paddle{
Width
Height
X, y
Color
Speed
Move_up_flag
Move_down_flag

Move_up : moves the paddle up
Move_down: moves the paddle down
Update_position: updates the new position in the game
}

Class Score{
Score1
Score2

Update_score : updates the scores every time someone scores
}

Class PongModel: sets up the pong game with one class

VIEW

Class: PongView{
Screen_width
Screen_height
Screen

Draw_ball : renders the ball
Draw_paddle: renders the paddle
Draw_scores: renders the score at the top
Update: updates the states of the ball, paddles, and score
}

CONTROLLER

Class PongController{
Model
View
Opponent

Handle_events : this handles any game logic and determines how players move and what determines a score

Update: updates all the game logic in real time as players score points, move their paddles, and the ball's projection

Run: runs game loop
}

The model contains all the entities, the view renders those entities onto the screen and the controller uses the information about the entities from the model to update the current game state, handle inputs, and run the game loop