PyGame Pong Tutorial - Part 5

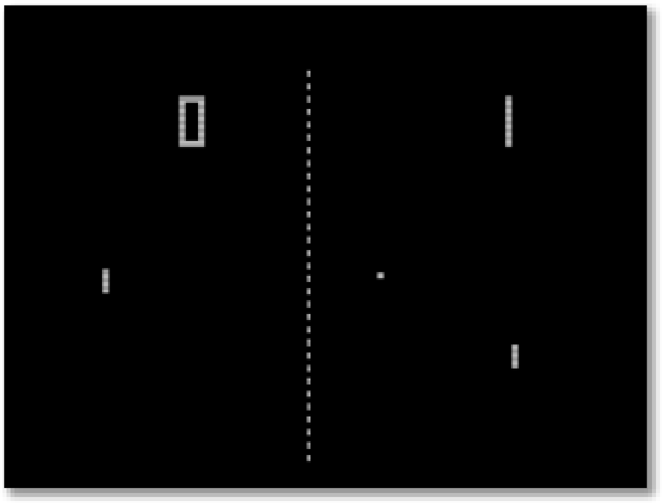
Contents

Time required: 30 minutes

Preview of the Game

Atari. - the year: 1973 - the date: - November 29th - The game is Pong.

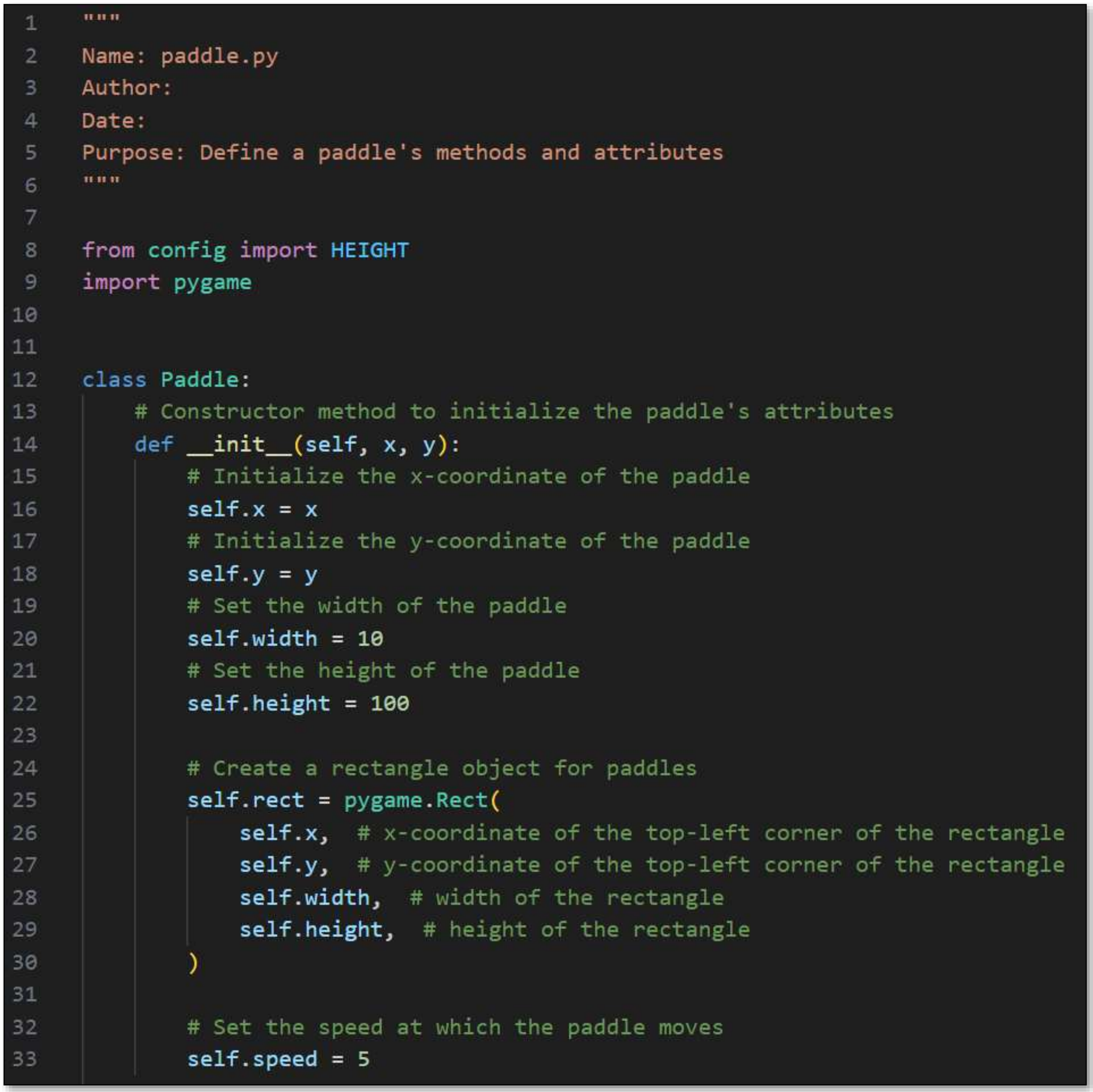
Pong Demo Video



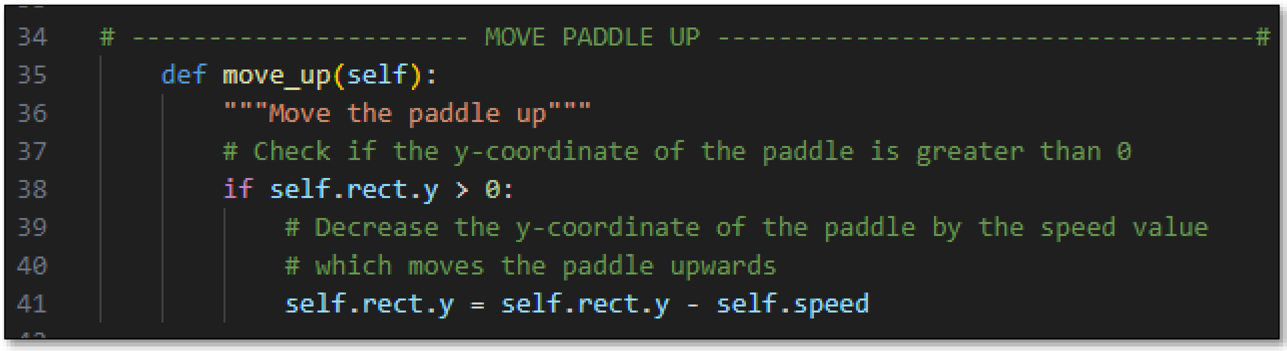
Paddles

It is time to get our paddle on. We will have a player paddle and a computer paddle with AI. Not a very smart AI, but it does move up and down.

1. Create a new Python file named: **paddle.py**
2. Add the following code.

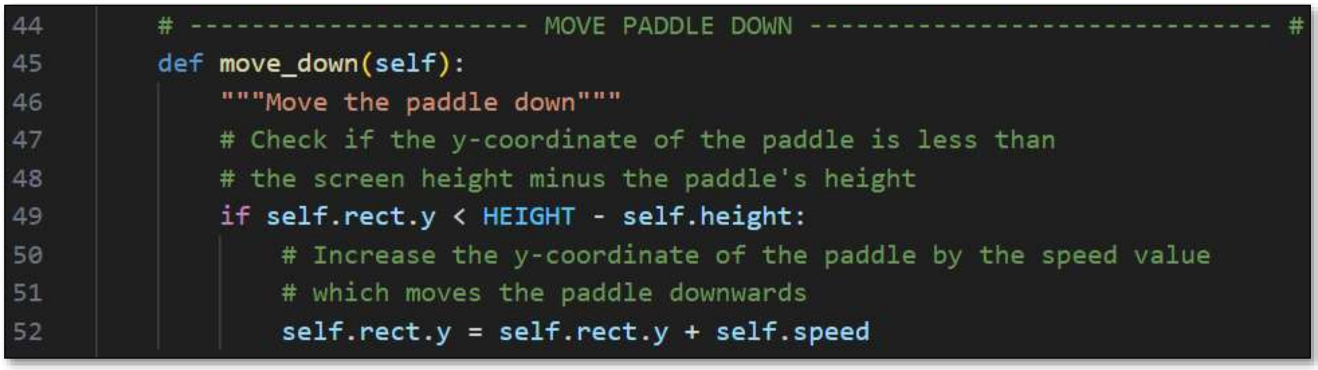


```python
"""
Name: paddle.py
Author:
Date:
Purpose: Define a paddle's methods and attributes
"""

from config import HEIGHT
import pygame


class Paddle:
    # Constructor method to initialize the paddle's attributes
    def __init__(self, x, y):
        # Initialize the x-coordinate of the paddle
        self.x = x
        # Initialize the y-coordinate of the paddle
        self.y = y
        # Set the width of the paddle
        self.width = 10
        # Set the height of the paddle
        self.height = 100

        # Create a rectangle object for paddles
        self.rect = pygame.Rect(
            self.x,  # x-coordinate of the top-left corner of the rectangle
            self.y,  # y-coordinate of the top-left corner of the rectangle
            self.width,  # width of the rectangle
            self.height,  # height of the rectangle
        )

        # Set the speed at which the paddle moves
        self.speed = 5
```



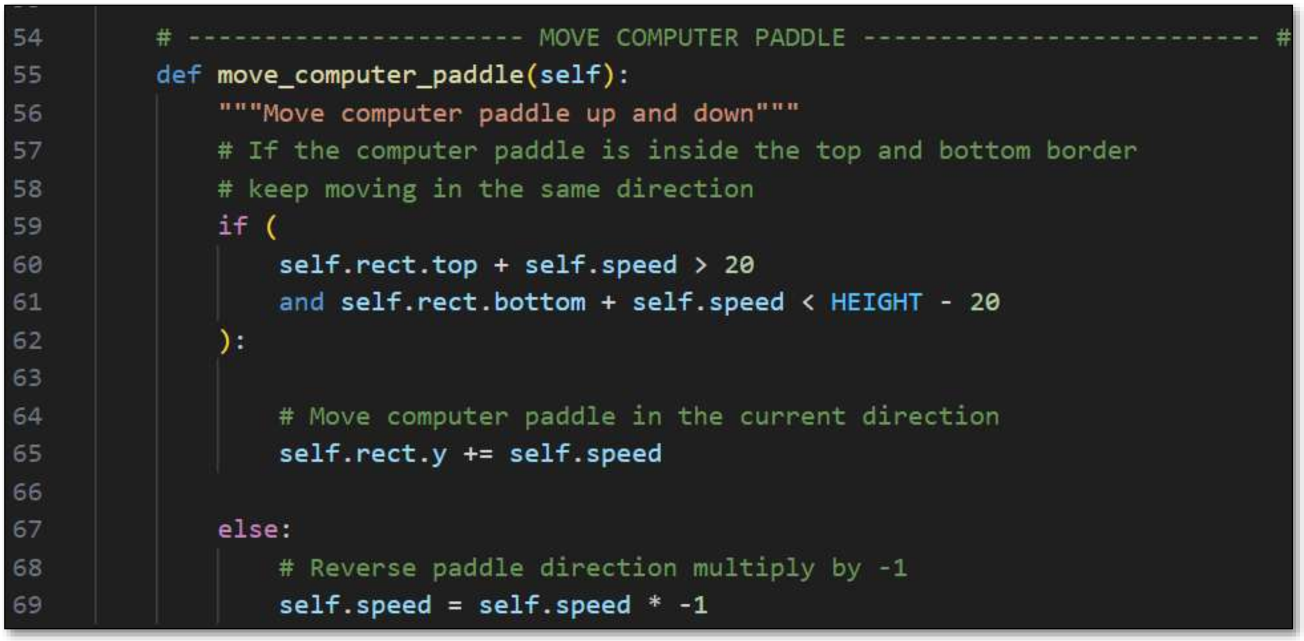
```python
# --------------------- MOVE PADDLE UP ---------------------------------#
    def move_up(self):
        """Move the paddle up"""
        # Check if the y-coordinate of the paddle is greater than 0
        if self.rect.y > 0:
            # Decrease the y-coordinate of the paddle by the speed value
            # which moves the paddle upwards
            self.rect.y = self.rect.y - self.speed
```

This method will be tied to the up cursor key to move the player paddle up.



```python
    # --------------------- MOVE PADDLE DOWN ----------------------------- #
    def move_down(self):
        """Move the paddle down"""
        # Check if the y-coordinate of the paddle is less than
        # the screen height minus the paddle's height
        if self.rect.y < HEIGHT - self.height:
            # Increase the y-coordinate of the paddle by the speed value
            # which moves the paddle downwards
            self.rect.y = self.rect.y + self.speed
```

This method moves the player paddle down the screen.

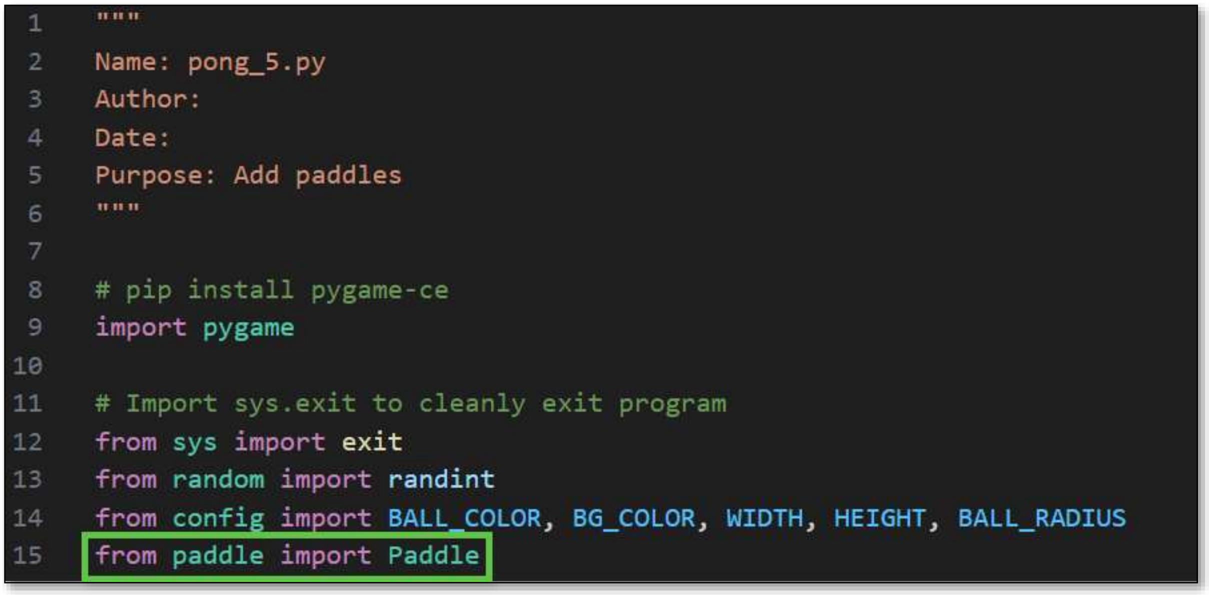


```python
    # --------------------- MOVE COMPUTER PADDLE ------------------------- #
    def move_computer_paddle(self):
        """Move computer paddle up and down"""
        # If the computer paddle is inside the top and bottom border
        # keep moving in the same direction
        if (
            self.rect.top + self.speed > 20
            and self.rect.bottom + self.speed < HEIGHT - 20
        ):

            # Move computer paddle in the current direction
            self.rect.y += self.speed

        else:
            # Reverse paddle direction multiply by -1
            self.speed = self.speed * -1
```

The computer paddle AI moves the paddle up and down the screen.

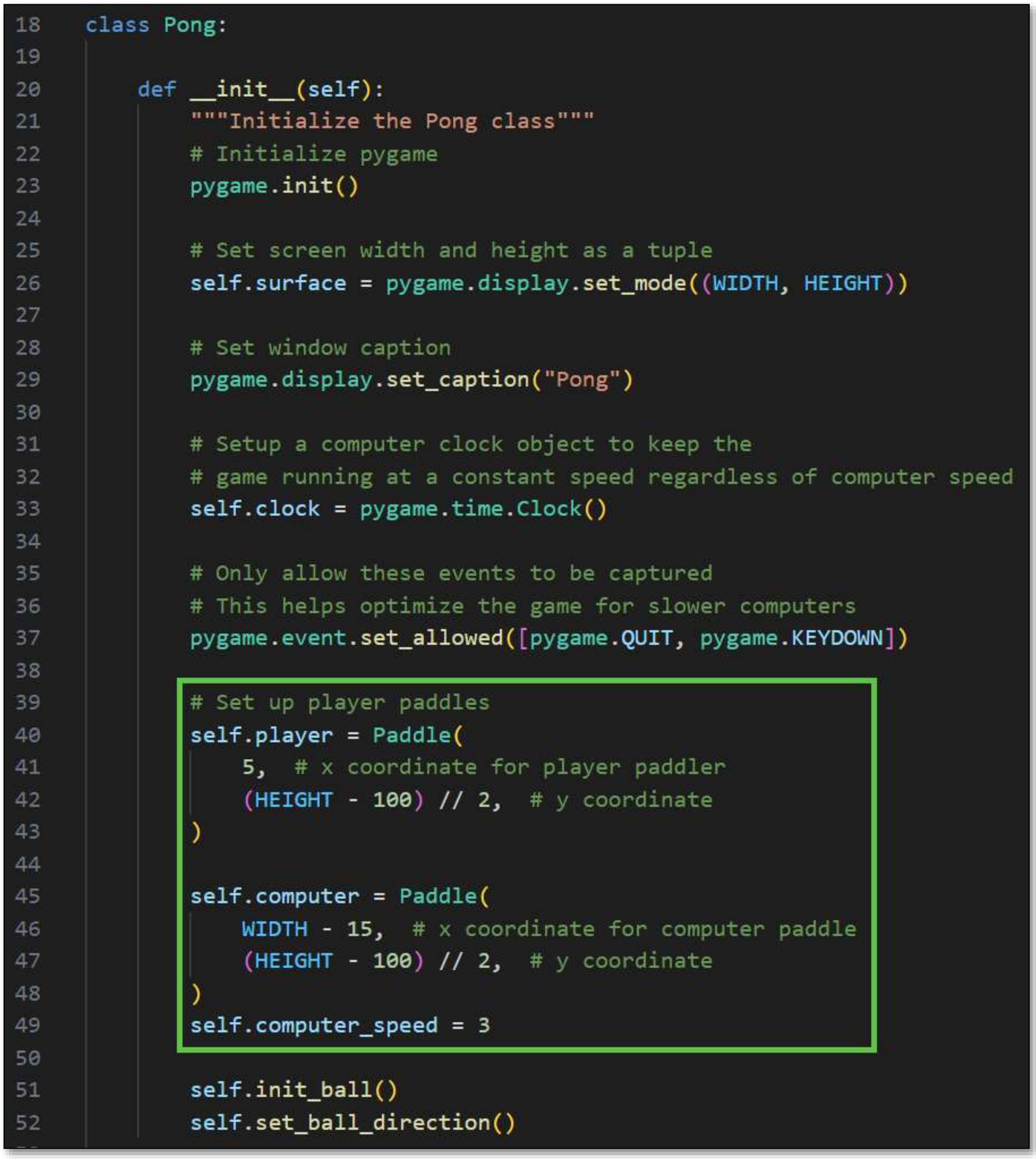
PONG 5

1. Save **pong_4.py** as **pong_5.py**
2. Import the Paddle class.



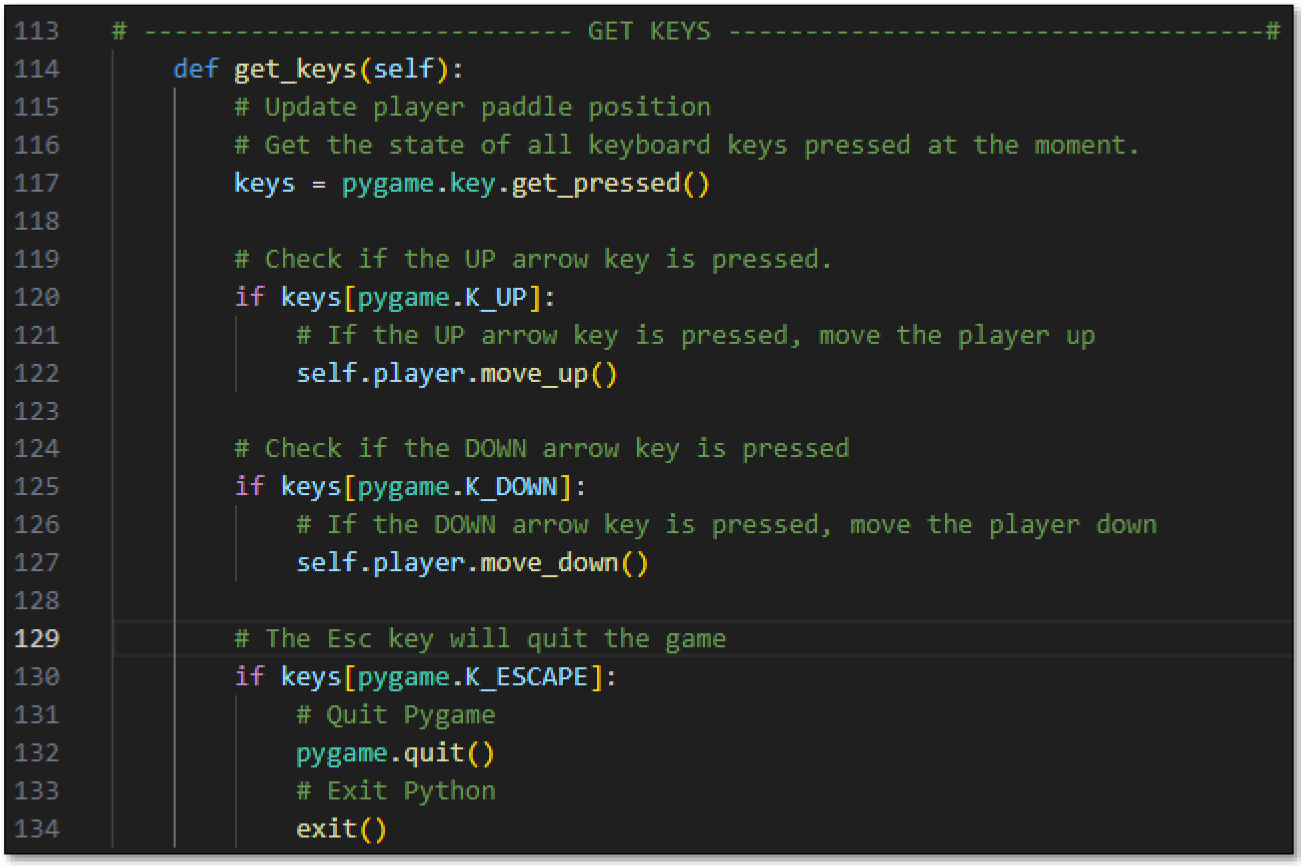
```python
"""
Name: pong_5.py
Author:
Date:
Purpose: Add paddles
"""

# pip install pygame-ce
import pygame

# Import sys.exit to cleanly exit program
from sys import exit
from random import randint
from config import BALL_COLOR, BG_COLOR, WIDTH, HEIGHT, BALL_RADIUS
from paddle import Paddle
```

3. Create a player and a computer paddle object.



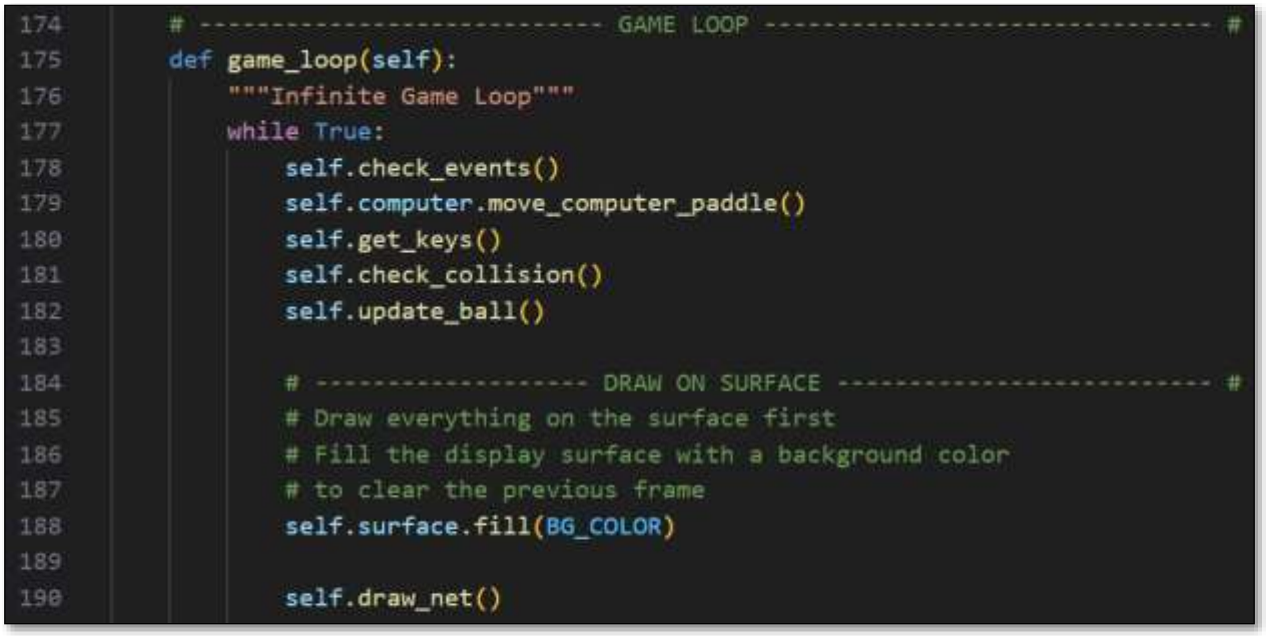
```python
class Pong:

    def __init__(self):
        """Initialize the Pong class"""
        # Initialize pygame
        pygame.init()

        # Set screen width and height as a tuple
        self.surface = pygame.display.set_mode((WIDTH, HEIGHT))

        # Set window caption
        pygame.display.set_caption("Pong")

        # Setup a computer clock object to keep the
        # game running at a constant speed regardless of computer speed
        self.clock = pygame.time.Clock()

        # Only allow these events to be captured
        # This helps optimize the game for slower computers
        pygame.event.set_allowed([pygame.QUIT, pygame.KEYDOWN])

        # Set up player paddles
        self.player = Paddle(
            5,  # x coordinate for player paddler
            (HEIGHT - 100) // 2,  # y coordinate
        )

        self.computer = Paddle(
            WIDTH - 15,  # x coordinate for computer paddle
            (HEIGHT - 100) // 2,  # y coordinate
        )
        self.computer_speed = 3

        self.init_ball()
        self.set_ball_direction()
```

4. Get the up and down arrow key pressed by the player.

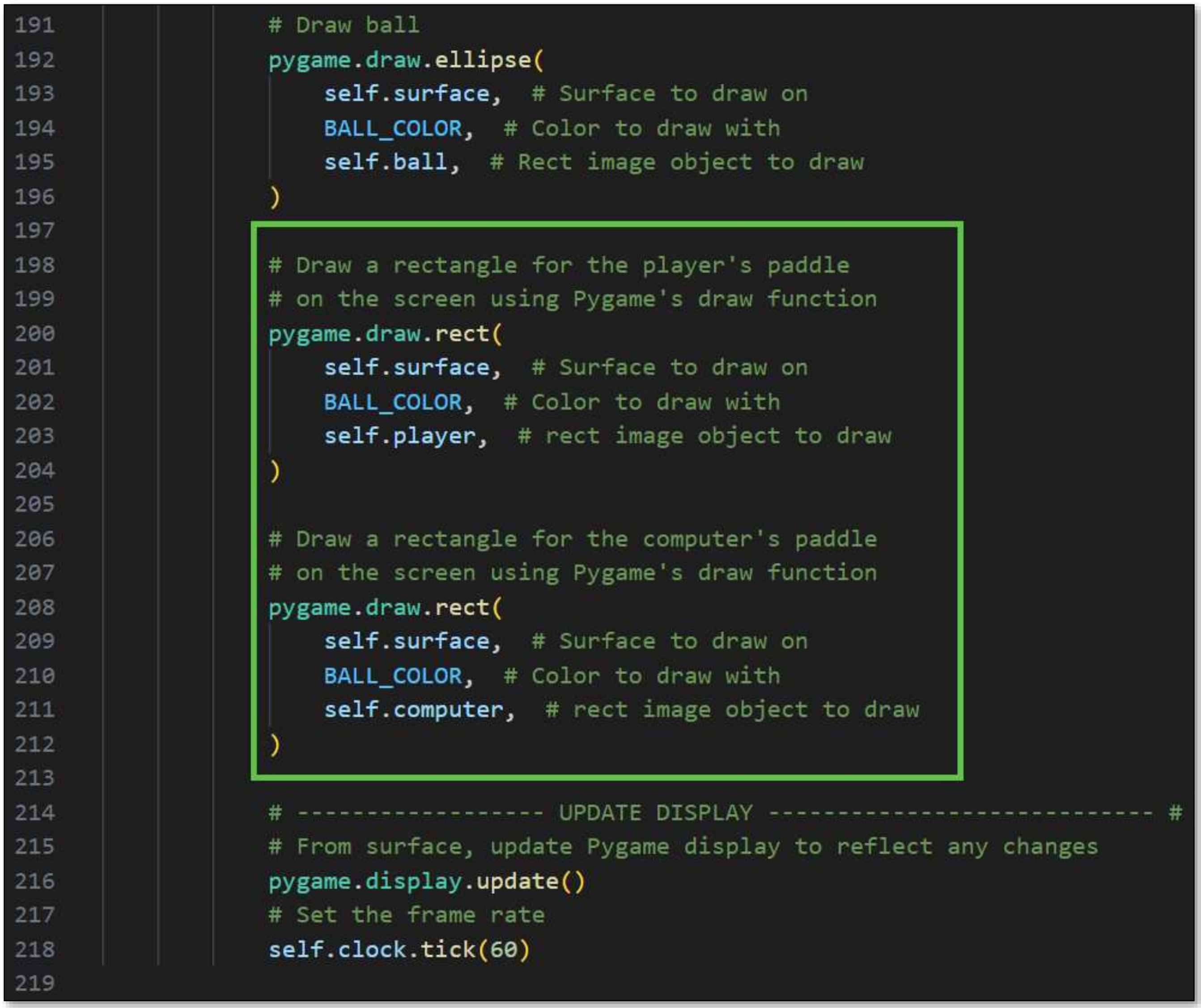


```python
# ---------------------------- GET KEYS -----------------------------------#
    def get_keys(self):
        # Update player paddle position
        # Get the state of all keyboard keys pressed at the moment.
        keys = pygame.key.get_pressed()

        # Check if the UP arrow key is pressed.
        if keys[pygame.K_UP]:
            # If the UP arrow key is pressed, move the player up
            self.player.move_up()

        # Check if the DOWN arrow key is pressed
        if keys[pygame.K_DOWN]:
            # If the DOWN arrow key is pressed, move the player down
            self.player.move_down()

        # The Esc key will quit the game
        if keys[pygame.K_ESCAPE]:
            # Quit Pygame
            pygame.quit()
            # Exit Python
            exit()
```

5. Modify the Game Loop.

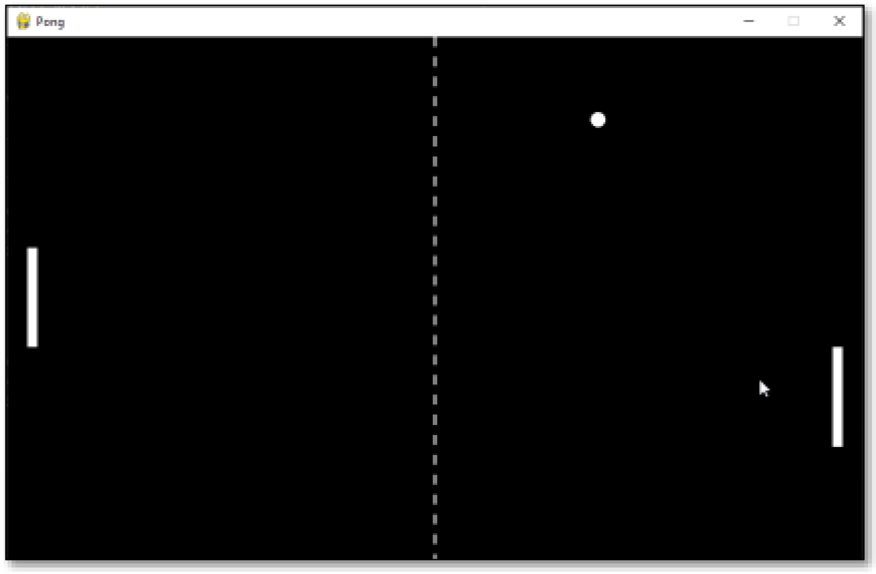


```python
    # ---------------------------- GAME LOOP ------------------------------- #
    def game_loop(self):
        """Infinite Game Loop"""
        while True:
            self.check_events()
            self.computer.move_computer_paddle()
            self.get_keys()
            self.check_collision()
            self.update_ball()

            # ------------------ DRAW ON SURFACE ------------------------- #
            # Draw everything on the surface first
            # Fill the display surface with a background color
            # to clear the previous frame
            self.surface.fill(BG_COLOR)

            self.draw_net()
```



```
            # Draw ball
            pygame.draw.ellipse(
                self.surface,  # Surface to draw on
                BALL_COLOR,  # Color to draw with
                self.ball,  # Rect image object to draw
            )

            # Draw a rectangle for the player's paddle
            # on the screen using Pygame's draw function
            pygame.draw.rect(
                self.surface,  # Surface to draw on
                BALL_COLOR,  # Color to draw with
                self.player,  # rect image object to draw
            )

            # Draw a rectangle for the computer's paddle
            # on the screen using Pygame's draw function
            pygame.draw.rect(
                self.surface,  # Surface to draw on
                BALL_COLOR,  # Color to draw with
                self.computer,  # rect image object to draw
            )

            # ------------------ UPDATE DISPLAY ---------------------------- #
            # From surface, update Pygame display to reflect any changes
            pygame.display.update()
            # Set the frame rate
            self.clock.tick(60)
```

Example run:



The ball moves . . . the paddles move . . .

Scoring is next.

Assignment Submission

1. Attach a screenshot showing the operation of the program.
2. Zip up the program files folder and submit in Blackboard.