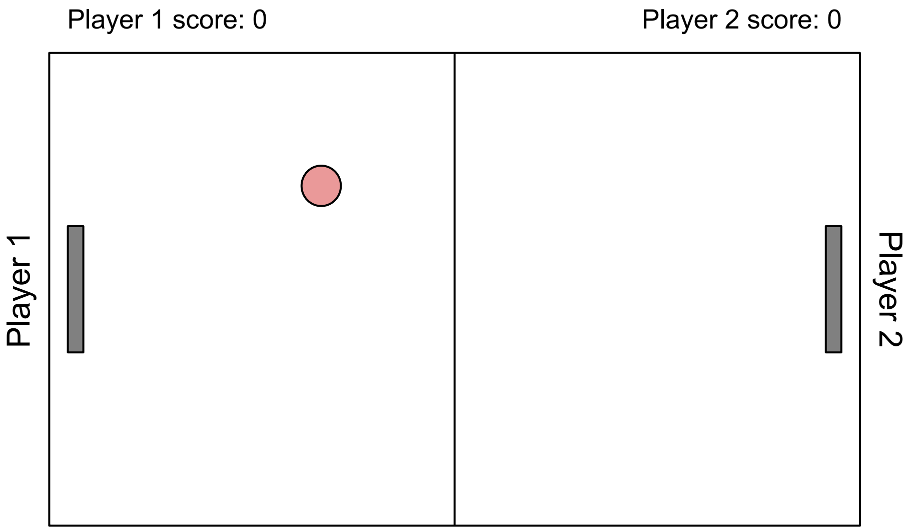
Philipps Universität Marburg

Preparation for Pong Tutorial

Tuesday, August 30 · 4:00 – 5:15 pm

Spielregeln

- Berührt der Ball den rechten/ linken Rand, so verliert der Spieler 1/2 einen Punkt.
- Die Spieler können die Paddle bewegen um den Ball daran zu hindern, die Linie zu berühren.
- Der Ball wird jeder Mal schneller, wenn er ein Paddle berührt.
- Das Spiel endet, wenn ein festgelegter Score erreicht wird.



Welche Klassen werden benötigt?

→ Gemeinsames Brainstorming an der Tafel/ Whiteboard

Now let’s walk through the code!

Open this link to browse the repo:

<LINK TO REPO>

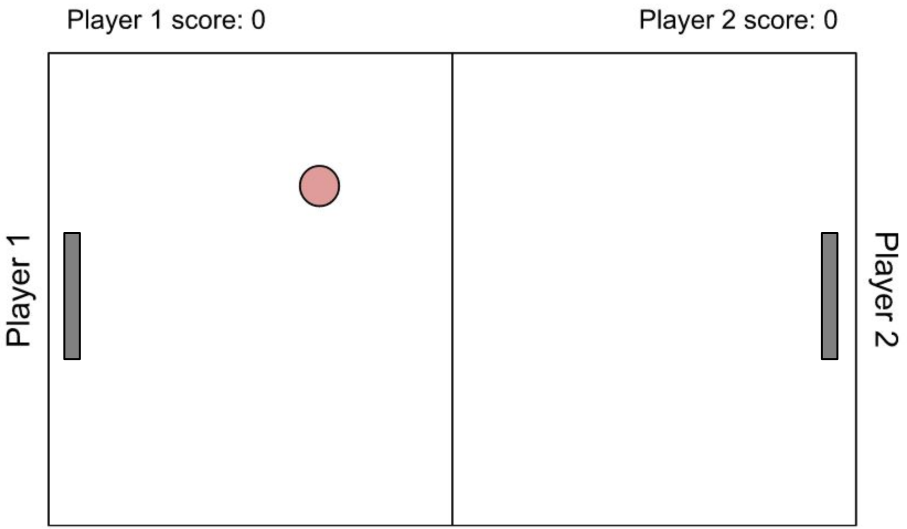
Clone the repo to your directory:

```
git clone <LINK TO REPO>
```

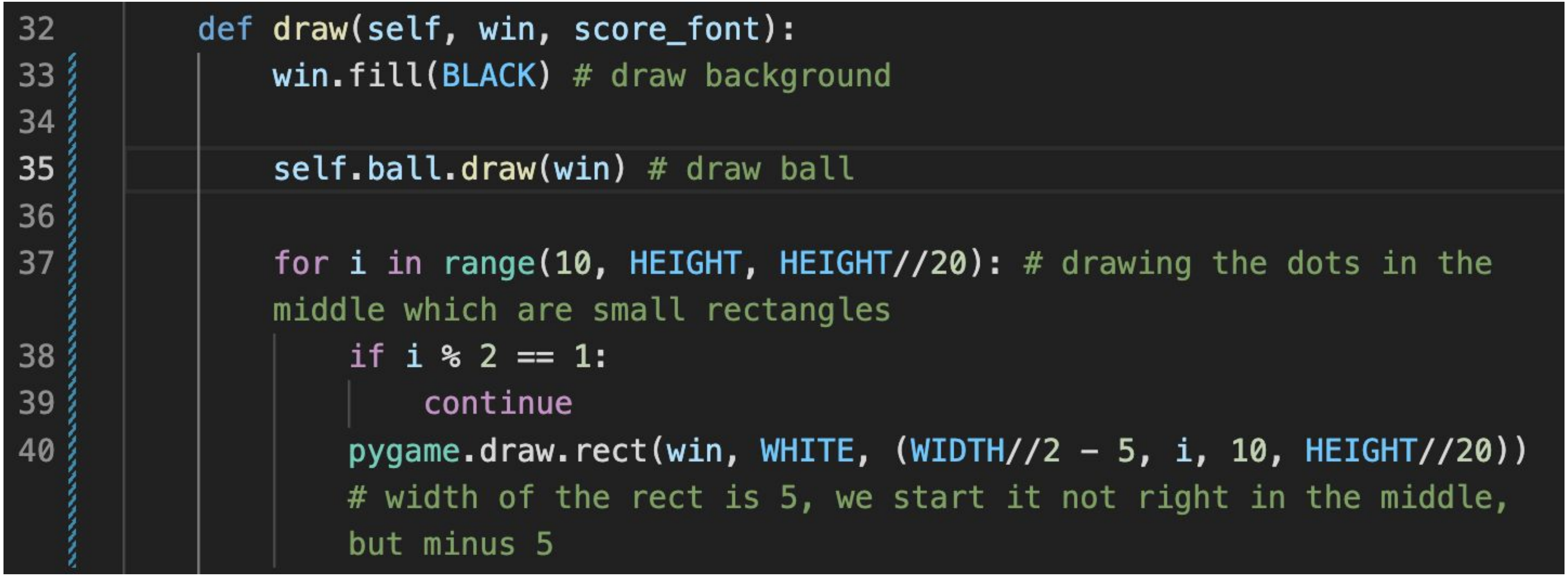
Environment of the Game

To make the environment of the game you need to draw:

- The background
- The ball
- The middle line
- The scores
- The paddles



Environment of the Game - a dive into the code!



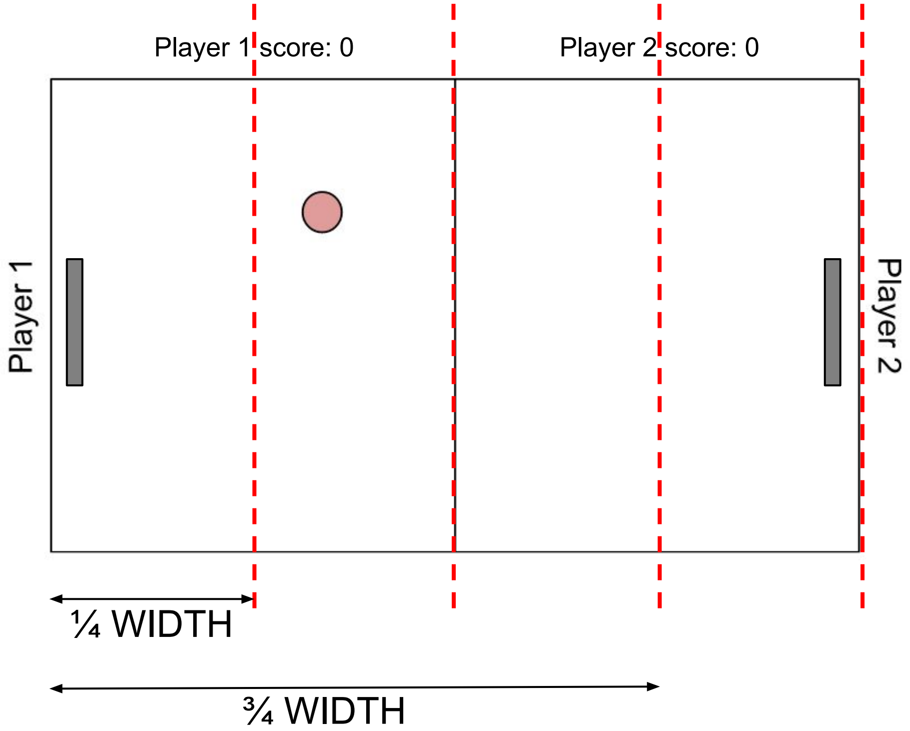
```python
    def draw(self, win, score_font):
        win.fill(BLACK) # draw background

        self.ball.draw(win) # draw ball

        for i in range(10, HEIGHT, HEIGHT//20): # drawing the dots in the middle which are small rectangles
            if i % 2 == 1:
                continue
            pygame.draw.rect(win, WHITE, (WIDTH//2 - 5, i, 10, HEIGHT//20)) # width of the rect is 5, we start it not right in the middle, but minus 5
```

Environment of the Game

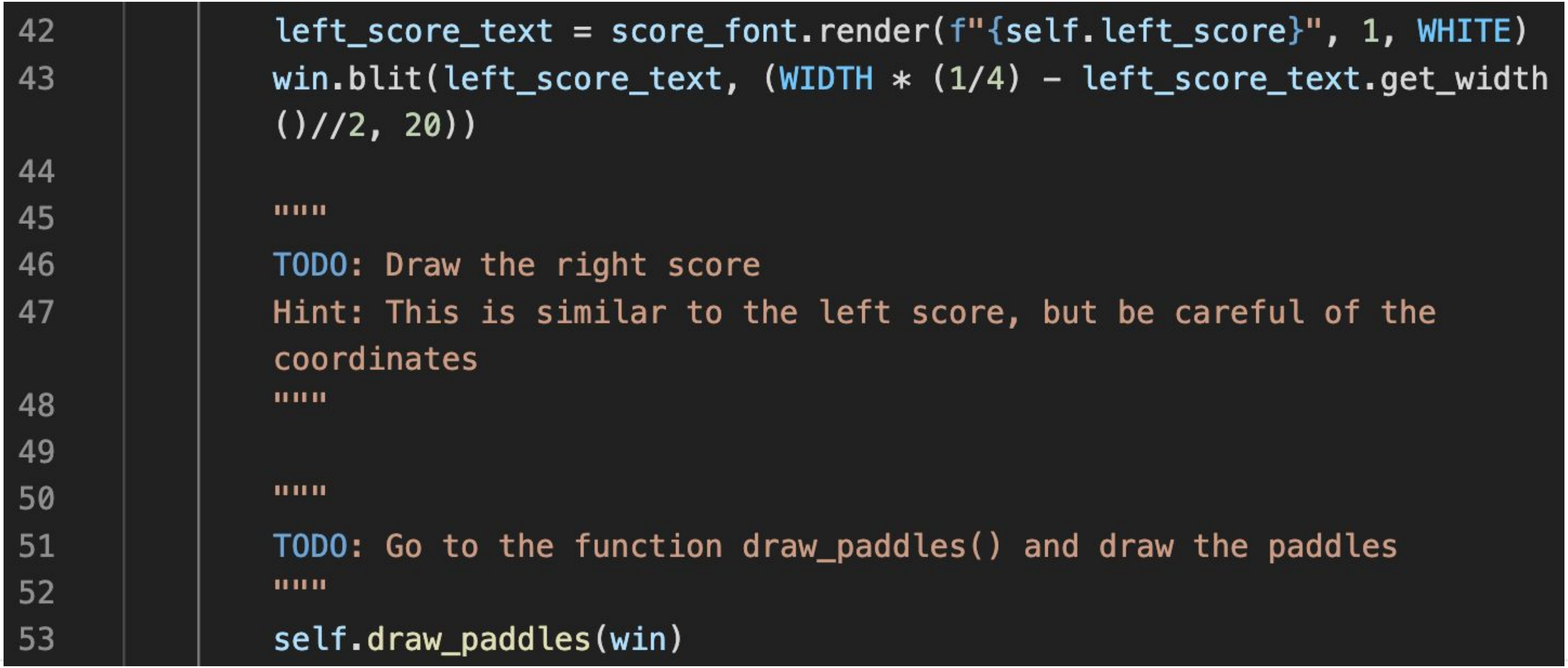
To make the environment of the game you need to draw:

- The background
- The ball
- The middle line
- **The scores**
- The paddles

In order to position the scores, you can split the background into 4 parts:



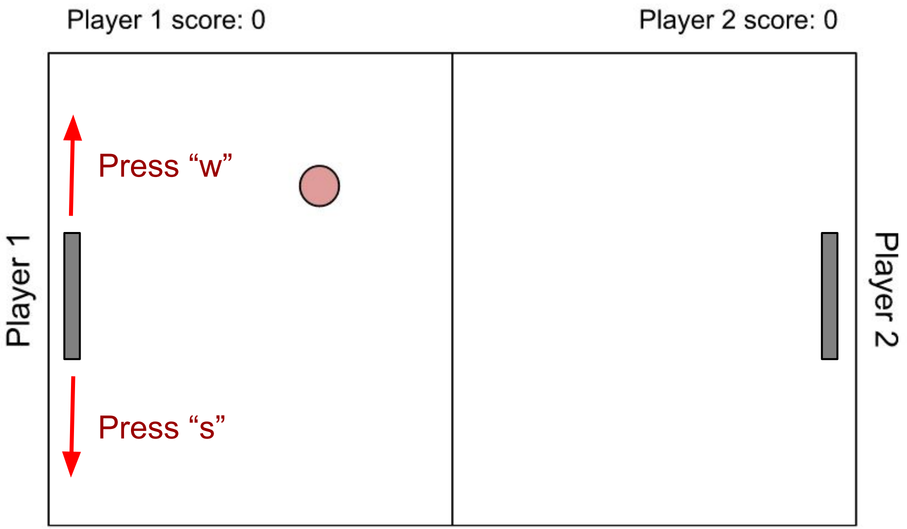
Environment of the Game - a dive into the code!



```python
        left_score_text = score_font.render(f"{self.left_score}", 1, WHITE)
        win.blit(left_score_text, (WIDTH * (1/4) - left_score_text.get_width()//2, 20))

        """
        TODO: Draw the right score
        Hint: This is similar to the left score, but be careful of the coordinates
        """


        """
        TODO: Go to the function draw_paddles() and draw the paddles
        """
        self.draw_paddles(win)
```

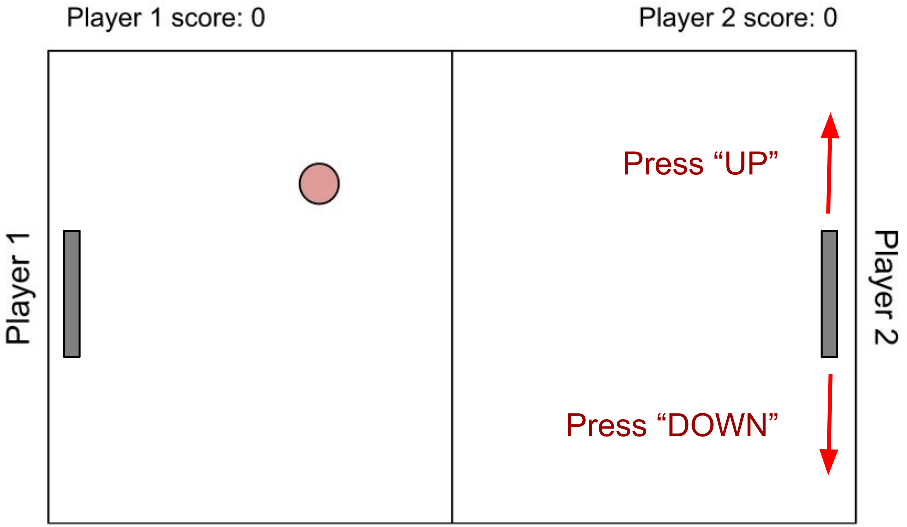
Paddle Movement

Player 1 (on the left) uses the keys "w" and "s" to move the paddle up and down



Paddle Movement

Player 2 (on the right) uses the keys "UP" and "DOWN" to move the paddle up and down

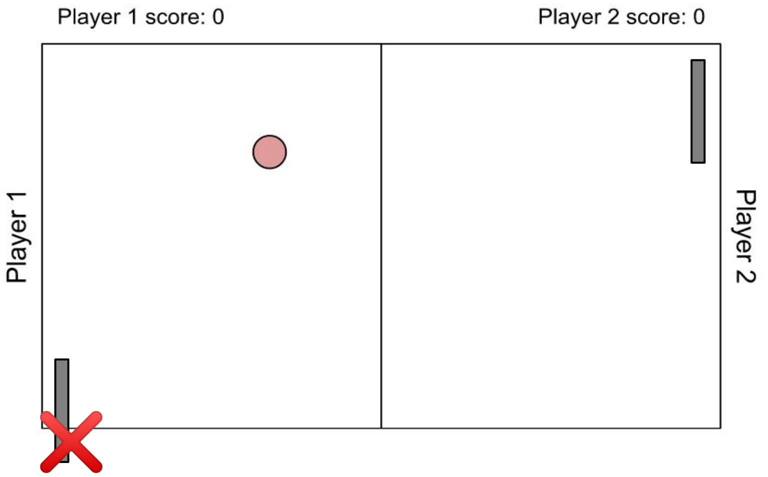


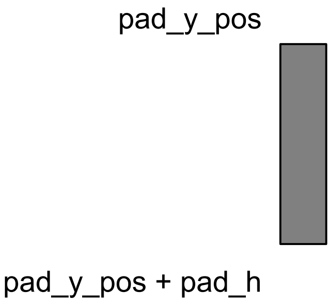
Paddle Movement

The paddles can't be moved outside the borders!

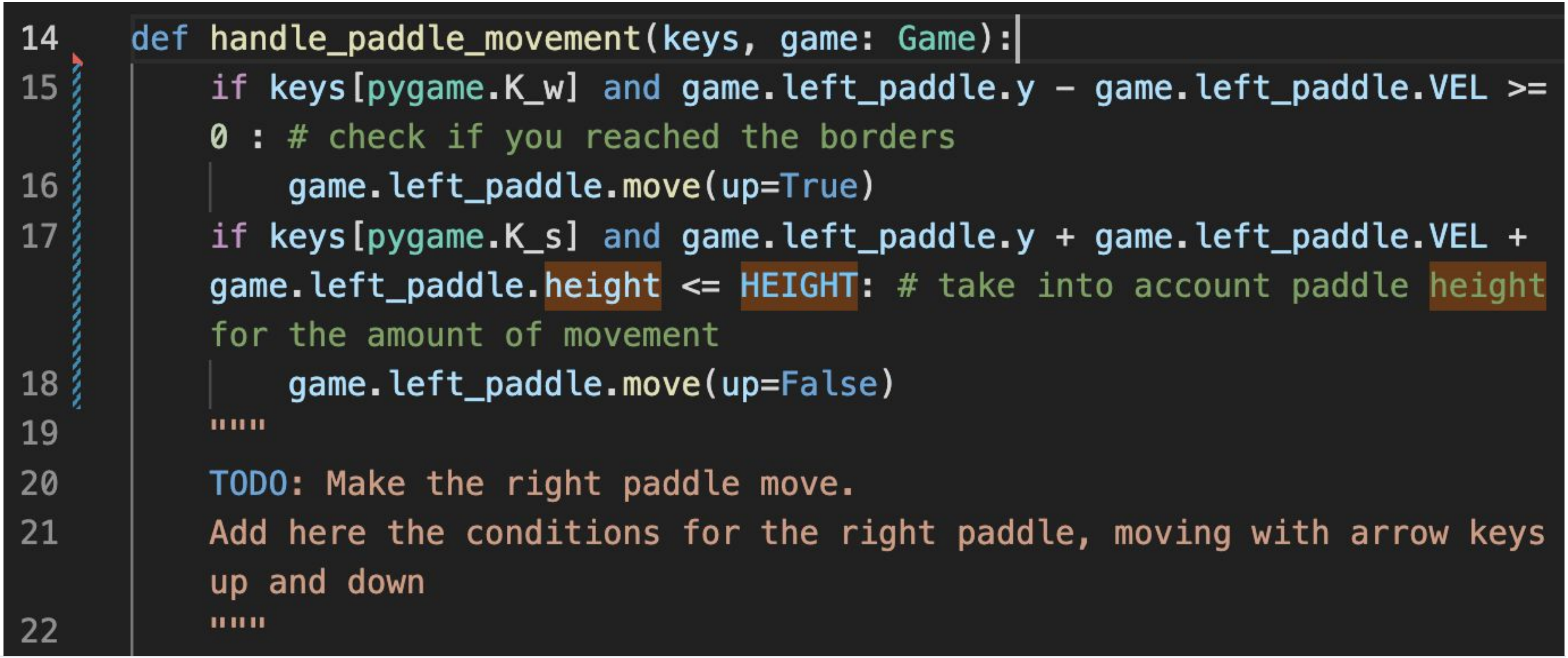
For Player 1:

```
if (key_w is pressed) and
(paddle_position + paddle_velocity >= 0):
    paddle_moves_up()

if (key_s is pressed) and
(paddle_position + pad_h +
paddle_velocity <= background_height):
    paddle_moves_down()
```





Paddle Movement - a dive into the code!



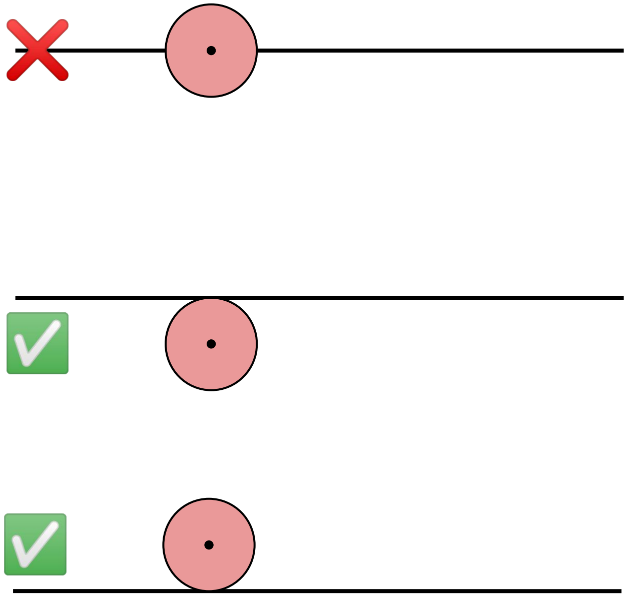
```python
def handle_paddle_movement(keys, game: Game):
    if keys[pygame.K_w] and game.left_paddle.y - game.left_paddle.VEL >= 0 : # check if you reached the borders
        game.left_paddle.move(up=True)
    if keys[pygame.K_s] and game.left_paddle.y + game.left_paddle.VEL + game.left_paddle.height <= HEIGHT: # take into account paddle height for the amount of movement
        game.left_paddle.move(up=False)
    """
    TODO: Make the right paddle move.
    Add here the conditions for the right paddle, moving with arrow keys up and down
    """
```

Collision with the border

The ball collides with the border when its **circumference** touches the border

```
if (ball_y_pos - ball_r <=
top_border_position):
    ball_collides = True
```

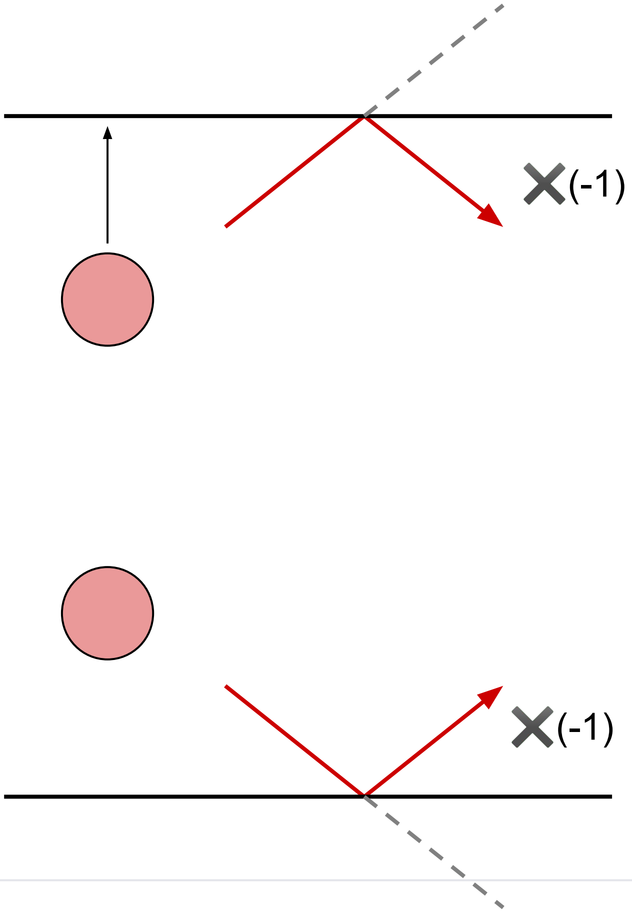
```
if (ball_y_pos + ball_r >=
bottom_border_position):
    ball_collides = True
```



Collision with the border

If the ball collides with the upper (or lower) border:

- Its velocity remains the same
- But it goes on the opposite direction on the y-axis



Collision with the paddle

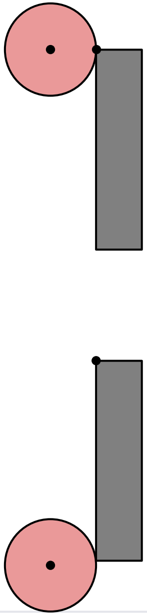
(same for left and right)

On the y-axis:

```
ball_y_pos >= pad_y_pos
```

and

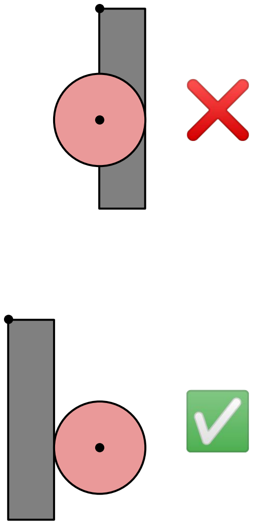
```
ball_y_pos <= pad_y_pos + pad_h
```



Collision with the left paddle

On the x-axis

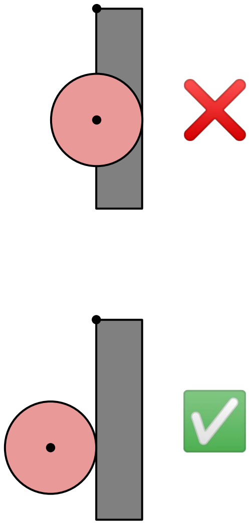
```
ball_x_pos -  ball_r >= pad_x_pos + pad_w
```



Collision with the right paddle

On the x-axis

```
ball_x_pos +  ball_r <= pad_x_pos
```



Handling the ball’s velocity

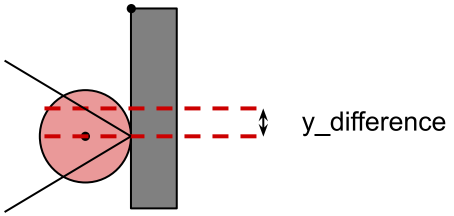
(same for left and right)

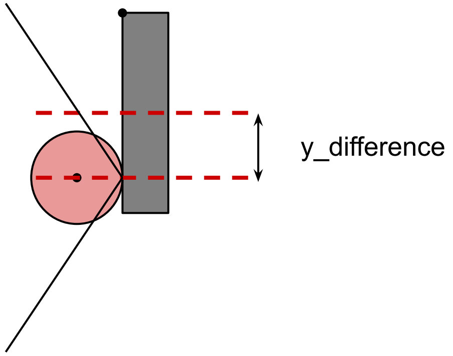
On the x-axis

```
ball_x_velocity *= -1
```

On the y-axis

```
reduction_factor = (pad_h / 2) / | ball_x_velocity |
y_vel = y_difference / reduction_factor
ball_y_velocity = -1 * y_vel
```


y_difference


y_difference