8-BALL POOL

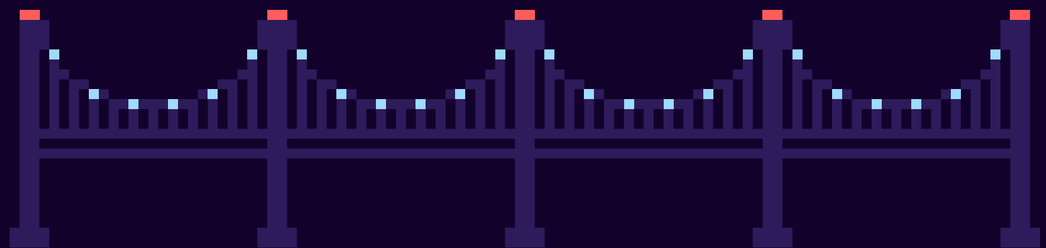
Intro to CS Final Project Presentation

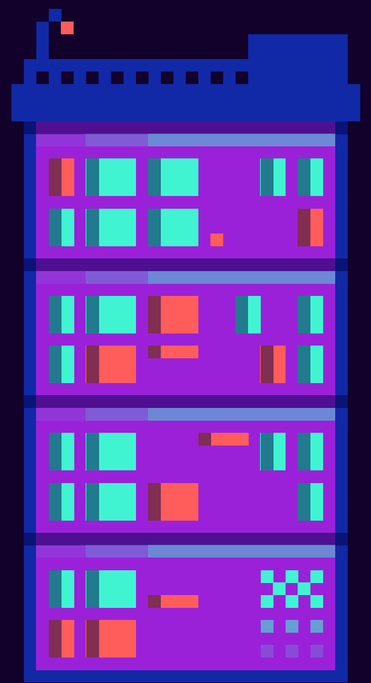
Begad, Myra, Siyona

ABOUT GAME

"8-Ball Pool" is a digital adaptation of the classic 8-ball billiards game. It is designed for two players who compete by following standard 8-ball rules. The game incorporates realistic sound effects, detailed graphics, and physics-based mechanics, such as velocity and friction, to simulate an authentic pool-playing experience.

The goal of 8-ball pool is to pocket all of your designated group of balls (either solids or stripes), then legally pocket the 8-ball to win.

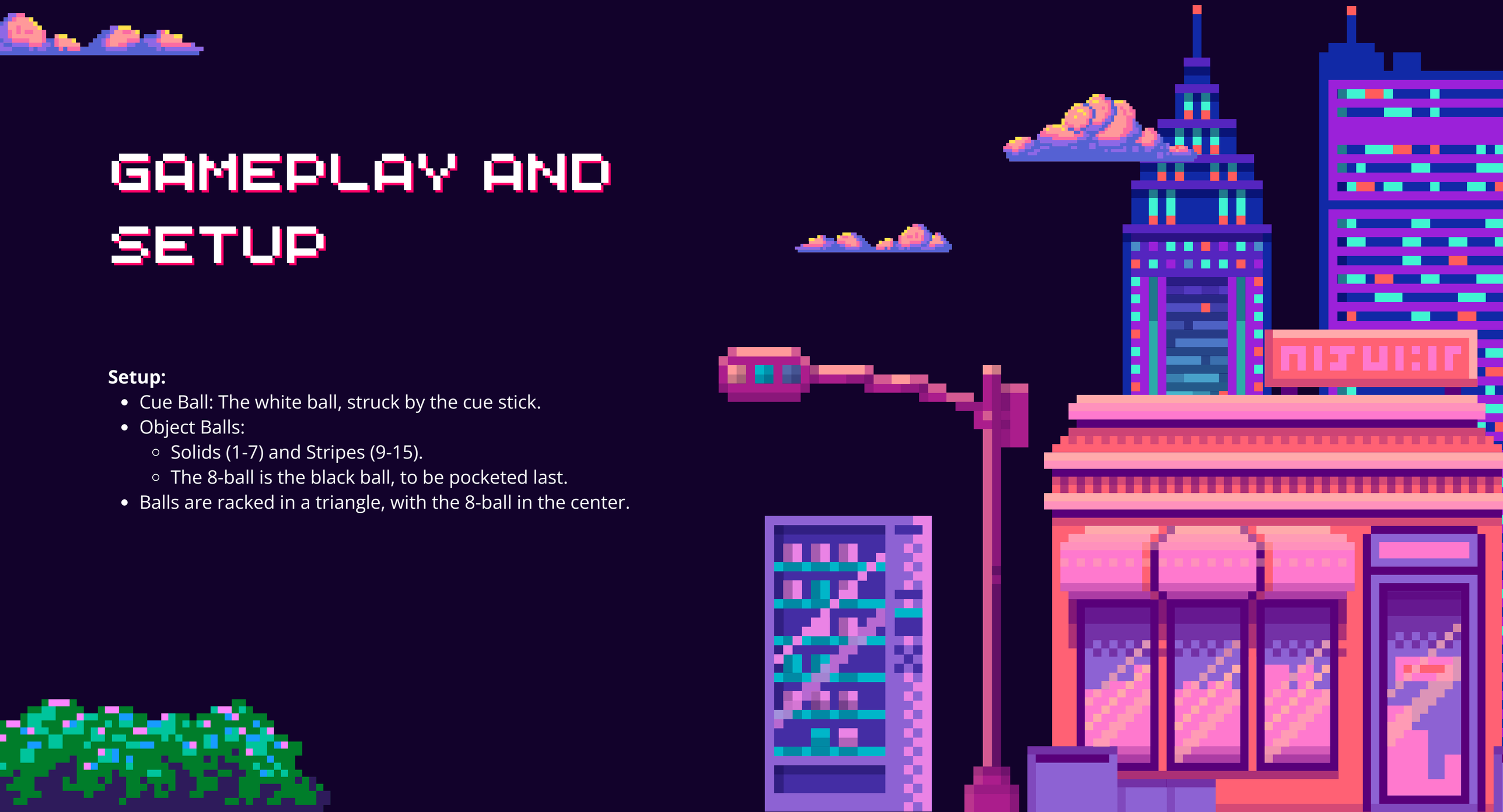




GAMEPLAY AND SETUP

Setup:

- Cue Ball: The white ball, struck by the cue stick.
- Object Balls:
  - Solids (1-7) and Stripes (9-15).
  - The 8-ball is the black ball, to be pocketed last.
- Balls are racked in a triangle, with the 8-ball in the center.



GAMEPLAY AND SETUP

Gameplay:

- Breaking: The player breaks the racked balls, and if they pocket a ball, they continue their turn and establish their group (solids or stripes).
- Turns: Players take turns attempting to pocket their group's balls. The player must strike their group ball first. If a ball is pocketed, the player continues their turn.
- 8-Ball Shot: Once all group balls are cleared, the player must call and legally pocket the 8-ball to win.

Winning the Game: The first player to legally pocket the 8-ball after clearing their group wins.

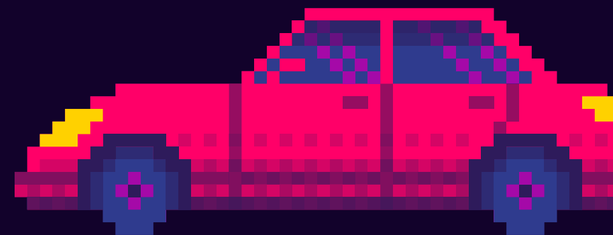
FOULS AND ILLEGAL MOVES

Ball-in-hand fouls

- Cue ball pocket: The opponent gets the ball in hand.
- No Rail / No Ball Hit: If the cue ball doesn't hit a ball or cause a ball to hit a rail.
- Wrong Ball First: The player hits an opponent's ball first or hits the 8-ball early.

Loss fouls

- Early 8-Ball Pocketing: If the 8-ball is pocketed before all group balls, it's a loss.
- Wrong Pocket for 8-Ball: Pocketing the 8-ball in an uncalled pocket is a loss.



GAME DEMO

https://zone.msn.com/en/8ballchampion

INTENDED CLASSES

01

BALL:

Attributes

- ID: int
- radius: int
- type: Str
- img: image
- position: int[]
- velocity: float[]
- Friction: float

Methods

- check_collision()
- update_position()
- display()

CueBall(Ball)

INTENDED CLASSES

02

TABLE:

Attributes

- pockets: Pocket[]
- dimensions: int[]
- cue_ball: CueBall
- balls: Ball[]
- img: image

Methods

- check_pocket()
- setup()
- display()

INTENDED CLASSES

03

STICK:

Attributes

- angle: float
- power: float
- img: image
- dimensions: int[]

Methods

- strike()
- adjust()
- display()

04

POCKET:

Attributes

- position: int[]
- radius: int

Methods

- has_ball()

INTENDED CLASSES

05

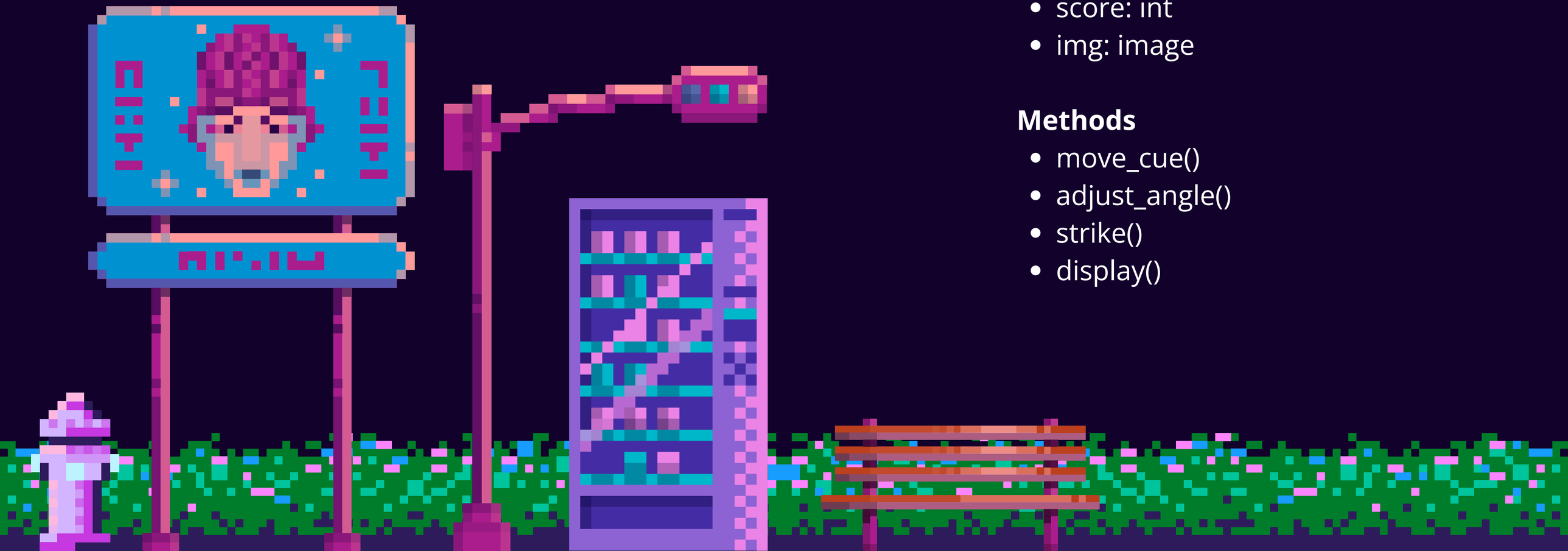
PLAYER:

Attributes

- name: Str
- ball_group: Str
- score: int
- img: image

Methods

- move_cue()
- adjust_angle()
- strike()
- display()



INTENDED CLASSES

06

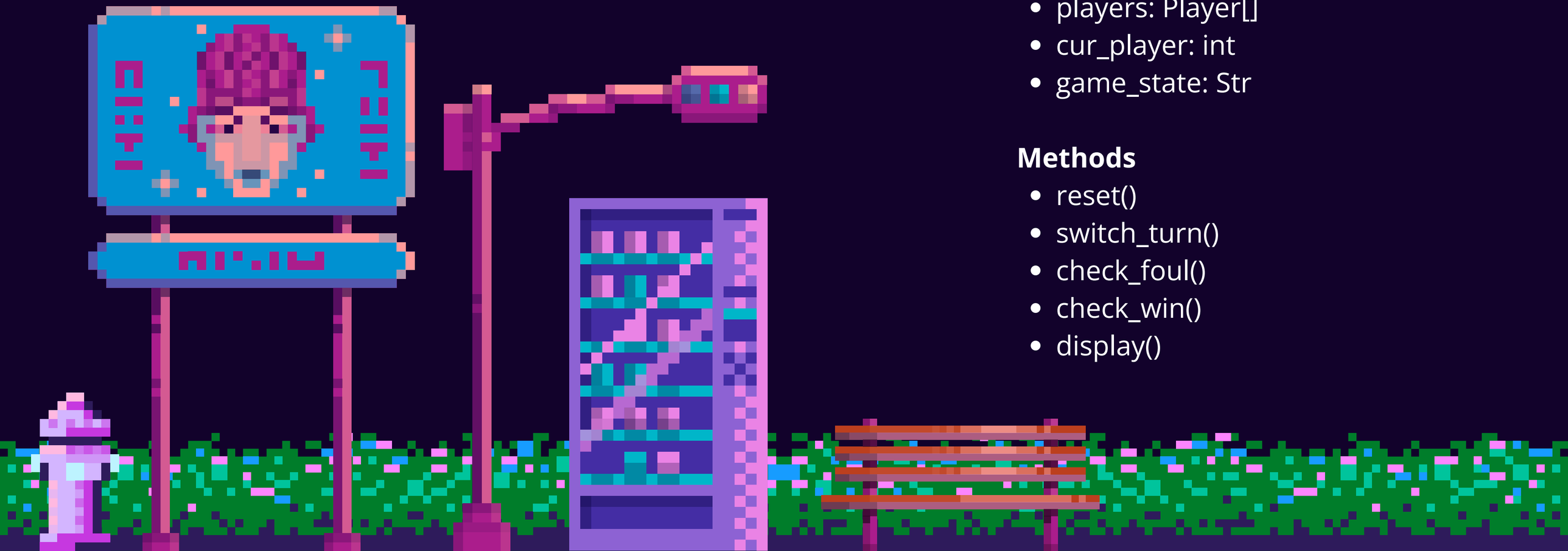
GAME:

Attributes

- table: Table
- score: int
- players: Player[]
- cur_player: int
- game_state: Str

Methods

- reset()
- switch_turn()
- check_foul()
- check_win()
- display()



FORMULAS

Let P=(x,y) be the position of the ball and V = (vx,vy) its velocity vector.

Collision with Vertical Rail

- **Condition**:
  - If the ball hits the left rail: Px ≤ 0 + radius
  - If the ball hits the right rail: Px ≥ W − radius
- **Equation**:

$$V' = (-v_x, v_y)$$

Collision with a Horizontal Rail

- **Condition**:
  - If the ball hits the upper rail: Py ≤ 0 + radius
  - If the ball hits the lower rail: Py ≥ H − radius
- **Equation**:

$$V' = (v_x, -v_y)$$

FORMULAS

Collision with Another Ball

- **Condition**:

$$distance(P_1, P_2) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \leq 2 \cdot radius$$

- **Equation**:

$$\triangle P = (x_2 - x_1, y_2 - y_1)$$

$$\hat{n} = \frac{\triangle P}{||\triangle P||}$$

$$v_{rel} = (V_1 - V_2) \cdot \hat{n}$$

$$V_1' = V_1 - v_{rel} \cdot \hat{n}$$

$$V_2' = V_2 + v_{rel} \cdot \hat{n}$$

DATA REQUIRED

01

Graphics:

- Balls
- Stick
- Table
- Player avatars

02

Audio:

- Strike
- Collide
- Pocket
- Win
- Music

03

Numerics:

- Ball radius
- Table dimensions
- Pocket spacings
- Friction

OUR WISHLIST

1. Ball animation
2. Menu: (New Game, Settings, How To Play...)
3. Player usernames
4. Themes
5. Avatar Selector

THANK
YOU