FATİH SULTAN MEHMET VAKIF ÜNİVERSİTESİ

PYTHON PROJECT RAPOR

NAME: **HALA JABBAN**

ST NO: **2121221350**

DATA: **12.05.2025**

PING PONG GAME

In this project, we developed an interactive Ping Pong game using the Pygame library in Python. The game aims to simulate the classic table tennis experience, where two players compete by moving paddles to return the ball and prevent it from passing their side.
The game features a simple and smooth design that allows players to control their paddles using the keyboard. A scoring system is implemented, and the ball resets automatically after each goal.
Throughout this project, we applied object-oriented programming (OOP) concepts such as objects and attributes, and we handled graphics, collisions, and events using Pygame.

This project enhances skills in building 2D games and demonstrates a good understanding of using external libraries to develop interactive applications.

Game Features

🎮 **Gameplay Basics:**

- Two players control paddles and compete to score points by hitting the ball.
- Scores are tracked and displayed at the top of the screen.

📢 **Music and Sound Effects:**

- Continuous background music plays during the game.
- Sound effects are triggered when a goal is scored or when the ball hits a paddle.
- A special sound plays when a reward is collected.

🎁 **Rewards and Special Effects:**

After scoring a goal, the player receives a random reward, such as:

- Enlarging their paddle.
- Shrinking the opponent's paddle.
- Freezing the opponent's movement for a few seconds.
- Activating a second ball on the field.

🖥️ **Main Menu Interface:**

- The start screen includes three options:
  Start Game, **View Results**, and **Exit Game**.

📊 **Scoreboard Screen:**

- Displays a record of the **last 10 matches** played between the two players.

⏱️ **Gameplay Timer:**

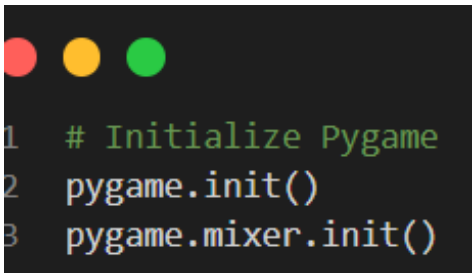
- Shows the elapsed time since the start of the current round.

⏸️ **Pause and Resume:**

- The game can be paused by pressing the **P** key and resumed the same way.

⏳ **Countdown Before Round Start:**

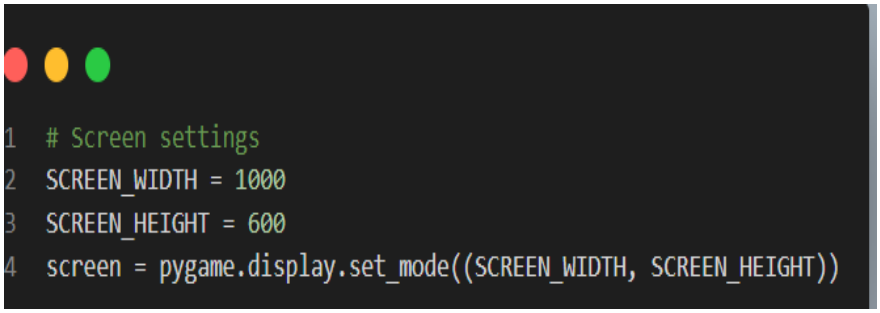
- A **3-second countdown** ("3, 2, 1,") appears before each round begin



```
# Initialize Pygame
pygame.init()
pygame.mixer.init()
```

→ Initializes the **Pygame library** to make it ready for use.

→ pygame.mixer.init() activates the **sound system**.



```
# Screen settings
SCREEN_WIDTH = 1000
SCREEN_HEIGHT = 600
screen = pygame.display.set_mode((SCREEN_WIDTH, SCREEN_HEIGHT))
```

→ Creates a game window with a size of 1000×600 pixels.
→ Defines the colors that will be used (e.g., white, black, red, blue...).

Fonts and Texts:

→ Loads font styles used to display **scores, names, and messages** in the game.
→ This adds a nice appearance to the game interface.

Sounds and Background:

→ Loads **background music** and sound effects like goal sounds or hit sounds.
→ Loads the **background image** (masa.jpg) and resizes it to fit the screen

Start Screen (Game Interface)

Before the game begins, a **start screen** is displayed to enhance the player's experience and provide a smooth introduction to the gameplay. This interface includes:

- A **welcome message**, such as *"Welcome to Ping Pong!"*
- **Instructions** to start the game, usually by pressing the **ENTER** key.
- The option to **exit** the game using the **ESC** key.
- A clean **background** (either a solid color or an image).
- **Simple design and readable fonts** to make the interface user-friendly and visually appealing.This screen gives the game a more polished and professional look while also preparing the player mentally for the match. It also demonstrates the ability to design interactive graphical interfaces using the Pygame library.

```
1
2  title_text = font_title.render("PING PONG GAME", True, red)
3  screen.blit(title_text, (SCREEN_WIDTH // 2 - title_text.get_width() // 2, 100))
4
5  start_button = pygame.Rect(SCREEN_WIDTH // 2 - 100, 300, 200, 60)
6  results_button = pygame.Rect(SCREEN_WIDTH // 2 - 100, 400, 200, 60)
7  quit_button = pygame.Rect(SCREEN_WIDTH // 2 - 100, 500, 200, 60)
8
9  pygame.draw.rect(screen, orange, start_button)
10 pygame.draw.rect(screen, green, results_button)
11 pygame.draw.rect(screen, purple , quit_button)
12
13 start_text = big_font.render("START", True, white)
14 results_text = big_font.render("RESULTS", True, white)
15 quit_text = big_font.render("QUIT", True, white)
16
17 screen.blit(start_text, (start_button.centerx - start_text.get_width() // 2, start_button.cente
18 screen.blit(results_text, (results_button.centerx - results_text.get_width() // 2, results_butt
19 screen.blit(quit_text, (quit_button.centerx - quit_text.get_width() // 2, quit_button.centery -
20
```

Game Elements:

Ball and Paddles

```
paddle1 = pygame.Rect(50, SCREEN_HEIGHT // 2 - PADDLE_HEIGHT // 2, PADDLE_WIDTH, PADDLE_HEIGHT)
paddle2 = pygame.Rect(SCREEN_WIDTH - 60, SCREEN_HEIGHT // 2 - PADDLE_HEIGHT // 2, PADDLE_WIDTH, PADDLE_HEIGHT)
```

→ Creates the ball in the middle of the screen.
→ Creates two paddles: one on the left (Player 1), one on the right (Player2).
→ Sets the speed of the ball and paddles.

- **Ball**: A rectangle of size 30x30 placed at the center with a starting speed of (5,5).
- **Paddles**: Two rectangles (left and right), each with size 20x100.
- **Second Ball**: An optional feature for the "Second Ball" reward.

Game State

```
score1 = 0
score2 = 0
max_score = 7
game_over = False
```

→ Variables to **store scores**, game-over status, pause status, etc

- **Scores**: score1, score2 initialized to 0.
- **Game control**: flags like game_over, paused, winner, reward_active, etc.
- **Timers**: countdown timer, reward timers, and game elapsed time tracker.

Rewards and Freeze Mechanics:

```
reward_options = ["Enlarge", "Shrink Opponent", "Freeze Opponent", "Second Ball"]
    reward = random.choice(reward_options)
```

→ Rewards are **special power-ups** the player gets after scoring a goal:

- **Enlarge**: Enlarges the player's paddle.

- **Shrink Opponent**: Shrinks the opponent's paddle.
- **Freeze Opponent**: Freezes the opponent temporarily.
- **Second Ball**: Adds a second ball to the game!

Functions:

- **reset_ball(direction)**: Resets the ball in the center and sets the direction.
- **apply_reward(player)**: Randomly applies a reward to a player when they score (Enlarge paddle, Shrink opponent, Freeze opponent, or Second Ball).
- **update_game()**: Moves the main ball and second ball (if exists).
- **move_ball(b, speed)**: Handles ball movement, bouncing, paddle collision, and scoring logic.
- **reset_rewards()**: Resets reward effects after their duration ends.

Freezing:

A player can freeze the opp onent's paddle, preventing them from moving for a limited time.

Game Functions:

Move Ball (move_ball)

→ Moves the ball and checks for collisions with paddles and walls.
→ If a goal is scored, increases the score and applies a reward.

Apply Reward (apply_reward):

→ Resets the ball to the center of the screen after a goal.

The rewards last for a fixed duration (3 seconds)

Reset Rewards (reset_rewards):

→ Ends the effect of rewards (returns paddles to normal size or unfreezes them).

Draw Game (draw):

→ Draws the background, ball, paddles, scores, timer, rewards, countdown, and play-again button.

Penalties and Scoring:

- Whenever a player scores a point, rewards are given, and penalties (like shrinking the opponent's paddle or freezing them) are applied.
- The game keeps track of the score, and the first player to reach **7 points** wins.

Pause Function:

- Players can pause the game by pressing the **spacebar**.
- A message appears to notify the players that the game is paused.

Countdown:

- A countdown from 3 is shown before the game starts, displaying "3, 2, 1, GO!" before the round begins.

Main Game Loop:

```
running = True
while running:
    clock.tick(60)
```

→ This is the **heart of the game** where:

- It captures events (key presses, pause, restart).
- Moves the ball and paddles.
- Updates the screen every frame (60 frames per second).

Player Controls:

- **Player 1**:

  W = Up S = Down

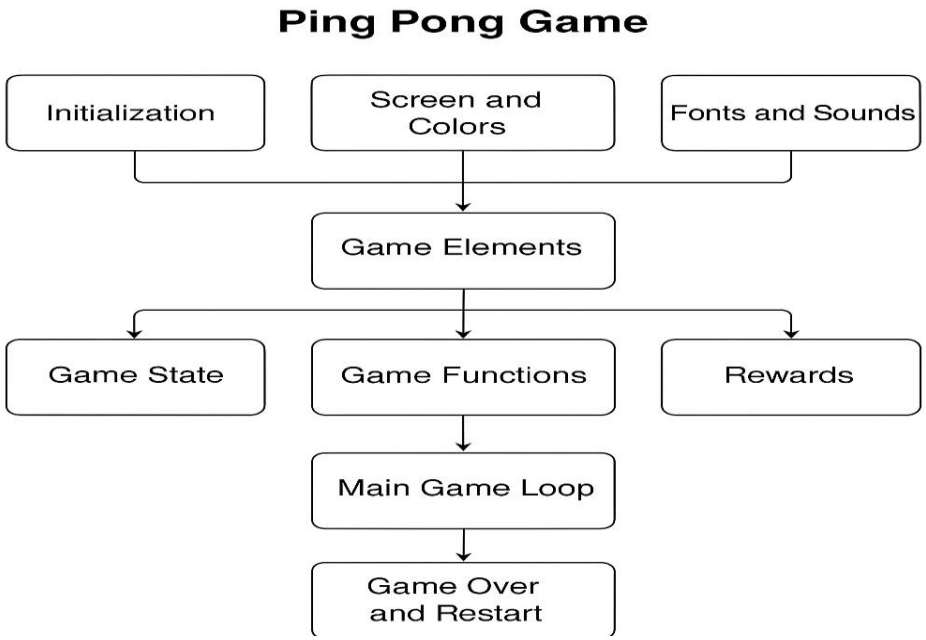
- **Player 2**:

  ↑ = Up
  ↓ = Down

- **Space**: Pause / Resume

Game Over and Restart:

→ When a player reaches **7 goals**, it shows who won and a "PLAY AGAIN" button to restart the game.



Player interaction with the game:

Player interaction in this game depends on tactics and timing. The player who can predict the movement of the ball and better control the speed of the racket will win.

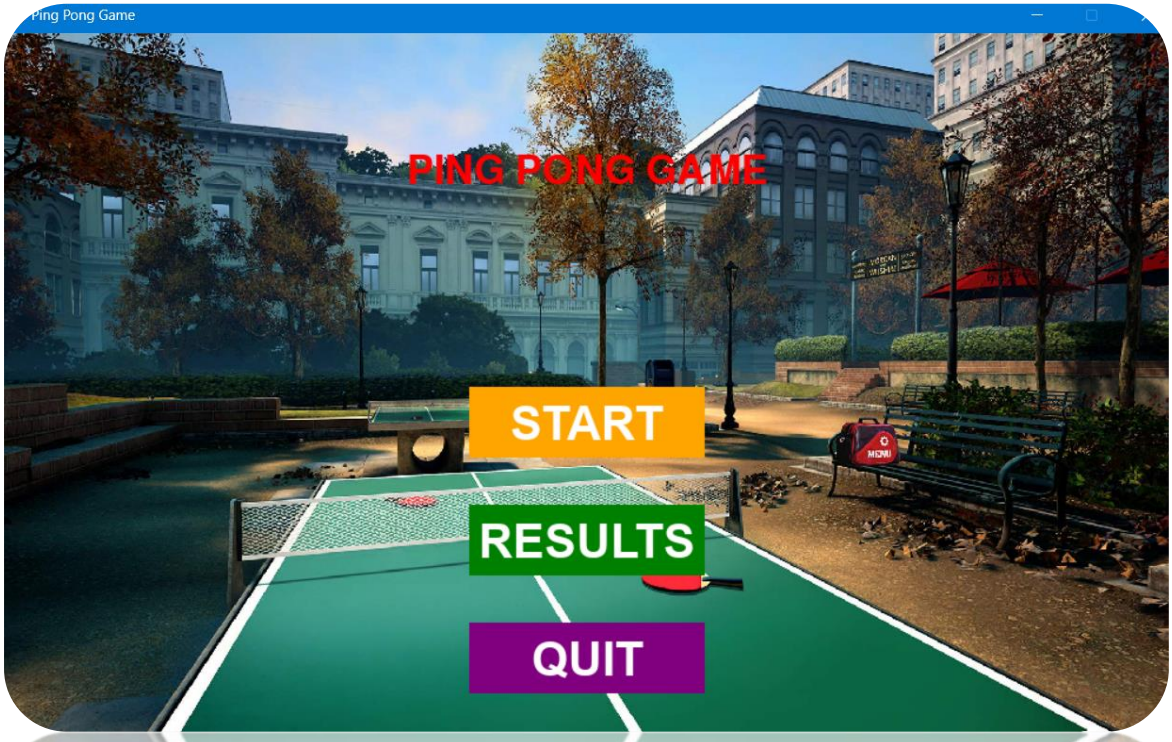
In conclusion:

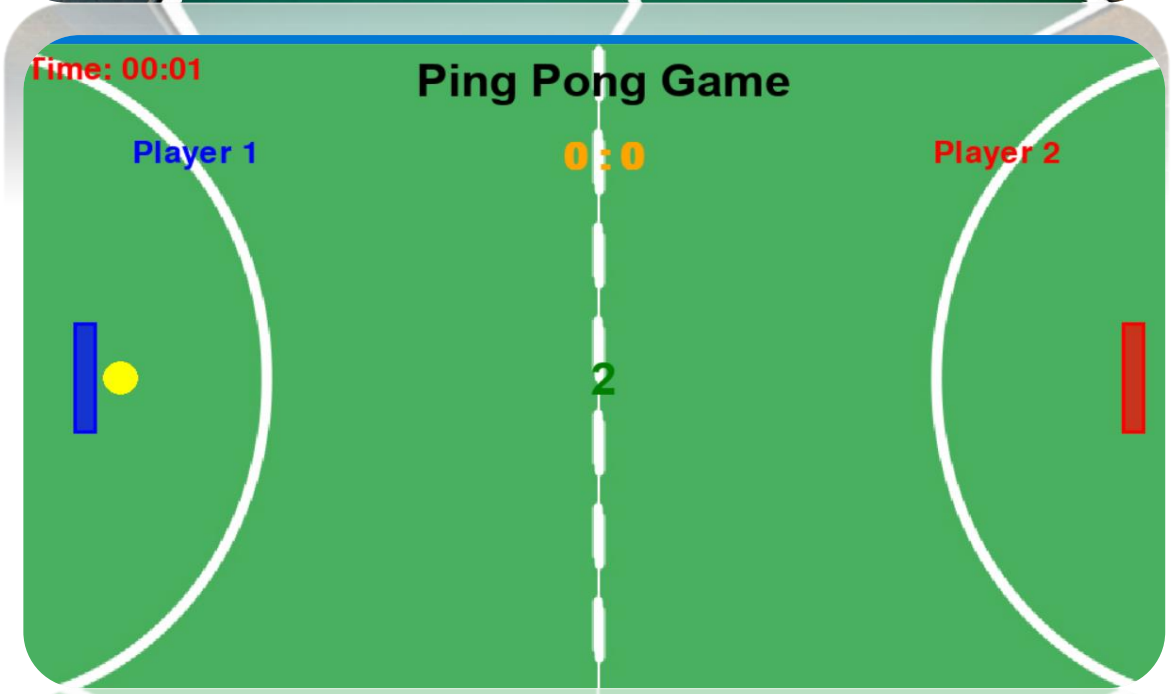
our **Ping Pong** game combines **simplicity** and **challenge** to create an exciting and engaging experience. With features like **interactive rewards**, **smooth paddle control**, and the **second ball**, the game offers both fun and strategic depth. The **pause functionality**, **countdown**, and **freezing opponents** add variety and excitement, making each round unique.

The game is more than just a competition—it's a **dynamic adventure** that encourages players to think strategically and stay engaged. By blending modern programming techniques with appealing graphics and sound, we've created a game that's enjoyable, immersive, and always thrilling.

Ultimately, **Ping Pong** is a fun and interactive experience, offering players endless excitement and challenges every time they play.

Welcome to the world of Ping Pong! 🤩






Time: 01:22
Ping Pong Game
Player 1
2 : 1
Player 2
Player 1 Reward: Freeze Opponent


Time: 00:58
Ping Pong Game
Player 1
5 : 7
Player 2
Player 2 Wins
PLAY AGAIN
MAIN MENU


MATCH HISTORY
Player 2 won 5 : 7
Player 2 won 5 : 7
BACK