ROCK-PAPER-SCISSORS-LIZARD-SPOCK GAME

Introduction:

- ➢ What is Rock-Paper-Scissors-Lizard-Spock?
- ➢ Popular extension of the classic Rock-Paper-Scissors game
- ➢ Adds more complexity and fun to the game
- ➢ Purpose: Build an interactive GUI game using Python and Tkinter

Features of the Game:

- GUI interface built with Tkinter
- Support for 5 choices: Rock, Paper, Scissors, Lizard, Spock
- Best of N rounds (user chooses number of rounds)
- Score tracking (User, Computer, Ties)
- Difficulty levels: Easy, Medium, Hard
- Visual representation using images for choices
- Timer countdown for user input (5 seconds)
- Sound effects for win/lose/tie (optional)
- Game history log
- Theme switching: Light and Dark mode
- Leaderboard to save and show previous game scores

User Interface Overview:

- Top panel for rounds input, difficulty, theme, and start button
- Middle panel shows scores and countdown timer
- Images for user and computer choices
- Choice buttons for user interaction
- Text area showing history of rounds played
- Button for showing leaderboard

Game Logic:

- **Random computer choice based on difficulty:**
  - *Easy: random*
  - *Medium: tries to counter last user choice*
  - *Hard: weighted to beat user’s last move*
- **Decision rules based on game’s winning logic:**
  - *Rock beats Scissors and Lizard*
  - *Paper beats Rock and Spock*
  - *Scissors beats Paper and Lizard*
  - *Lizard beats Paper and Spock*
  - *Spock beats Rock and Scissors*
- Scores updated accordingly

Timer and User Input:

- 5 seconds countdown timer for each round
- If user does not choose in time, computer auto-selects for user
- Timer displayed below scores

Additional Features:

- ➢ Sound effects (win/lose/tie) played asynchronously
- ➢ Light and dark themes for better user experience
- ➢ Game history logged in text box with round details
- ➢ Leaderboard stored in a local text file
- ➢ Image loading for visual feedback on choices

Code Structure:

- RPSGame class encapsulates all game logic and UI components
- **Initialization:** setup UI, load images, initialize variables
- Methods for game rounds, user input, computer AI, timer, sounds, themes
- Event-driven programming with Tkinter buttons and timers
- File I/O for leaderboard persistence

<u>How to Run:</u>

- ➢ Requires Python 3.x and Tkinter (comes preinstalled)
- ➢ **<u>Optional:</u>** playsound module for sound effects (pip install playsound)
- ➢ Place images (rock.png, paper.png, etc.) in the same directory
- ➢ Run the script with python filename.py in IDLE or terminal
- ➢ Interact with GUI window to play

Future Enhancements:

- ➢ Add network multiplayer mode
- ➢ Add animations for choices
- ➢ Use database instead of text file for leaderboard
- ➢ Add more sound effects and music
- ➢ Improve AI with machine learning techniques
- ➢ Mobile app version with Kivy or React Native

Conclusion:

- Developed an interactive, feature-rich GUI game in Python
- Covered game logic, UI design, user experience features
- Demonstrated use of threading, timers, file I/O in a project
- Good foundation for expanding into more complex projects