Ghulam Ishaq Khan Institute

Faculty of Electrical Engineering

AI Project Report

Project Title: Tic Tac Toe Game

M. Abdullah 2023324

Muhammad Ahmad Hadeed 2023368

Abstract:

The Tic Tac Toe Game is a console-based implementation developed in Python that allows players to engage in a classic game of Tic Tac Toe, either in a Player vs Player (PVP) mode or Player vs Computer (PVC) mode. The game features dynamic board updates, player input validation, and multiple difficulty levels for the PVC mode, where the computer makes its moves either randomly or using a more strategic approach. The program also tracks high scores for both game modes and stores them in a JSON file. Additionally, the high scores are displayed through pie charts using the matplotlib library, showing the win/draw statistics for both PVP and PVC categories. The game saves the results of each match and can be resumed or replayed, offering players an engaging and persistent experience. This project demonstrates the integration of Python programming, file handling, and data visualization in a fun and interactive game environment.

Table of Contents:

- Title Page
- Abstract
- Table of Contents
- Introduction
    1. Purpose of Project
    2. Scope of Project
- Game Requirements
- Design
- Implementation
- Challenges and Solution
- Future Improvements
- Conclusion
- Reference

Introduction

- <u>Purpose of Project:</u>

The Tic Tac Toe game implemented in Python aims to provide an engaging, interactive, and simple console-based version of the classic two-player game. This project was developed to enhance understanding and skills in Python programming, specifically in areas such as game logic implementation, file handling, user input validation, and data visualization. The game offers two modes: Player vs Player (PVP) and Player vs Computer (PVC), allowing players to either compete with another person or challenge an AI opponent. Additionally, the project tracks high scores and displays them through visual graphs, providing a dynamic gaming experience.

- Scope of the Project:

  The primary scope of the project is to create a functional Tic Tac Toe game that can be played in two modes: PVP, where two players take turns marking spaces on the grid, and PVC, where the player competes against a computer opponent. The game features two difficulty levels for PVC mode: Moderate and Hard. The game also includes persistent storage of high scores using a JSON file, which records the results of each match, including wins, losses, and draws for both modes. These scores are visually represented using pie charts created with the `matplotlib` library.

  The project was implemented as a console-based application, ensuring that it is simple to use and easily accessible to users. While it is not a fully graphical version of the game, it demonstrates key programming concepts and provides a foundation for further improvements, such as adding a graphical user interface (GUI) or enhancing the computer's AI.

  In terms of limitations, the current implementation focuses on local gameplay, and it does not support online multiplayer functionality. Additionally, while the computer opponent in PVC mode uses basic AI strategies, there is room for further refinement in making the computer's moves more intelligent and challenging.

Game Requirements

The Tic Tac Toe game is designed to offer an interactive and engaging experience for users, supporting two primary game modes: Player vs Player (PVP) and Player vs Computer (PVC). In the PVP mode, two players alternate turns to place their respective marks, 'X' and 'O', on a 3x3 grid, with the goal of getting three consecutive marks in a row, column, or diagonal. In PVC mode, the player competes against a computer opponent, with the computer's moves varying depending on the selected difficulty level. In Moderate difficulty, the computer makes random moves, while in Hard difficulty, the computer uses a strategic approach to either block the player from winning or attempt to win itself.

The game board is represented as a 3x3 grid, and players select positions by entering numbers between 1 and 9, each corresponding to a cell on the grid. The game checks for winning conditions after every move and declares a winner when three consecutive marks are placed. If all spaces are filled without a winner, the game results in a draw. The game includes input validation to ensure that players enter valid numbers within the given range, and it prevents players from selecting already occupied spaces.

A key feature of the game is its ability to track high scores, recording the number of wins and draws for both PVP and PVC modes. These scores are saved in a JSON file for persistent storage and can be accessed at any time. To enhance the user experience, the game visualizes these scores through pie charts generated using the matplotlib library. This provides a clear representation of the distribution of wins and draws for both game modes.

The game is implemented as a console-based application using Python, ensuring compatibility across platforms that support Python 3.x. The user interface is simple and text-based, displaying the board and prompting the player for input after every move. The game ensures performance efficiency, with fast responses to user input and quick updates to the game board. Data integrity is maintained, as high scores are reliably stored and handled, and the game resets the scores if the data file is corrupted. Lastly, players have the option to replay the game after each session or quit, ensuring that the game is enjoyable and can be played multiple times without needing to restart the program.

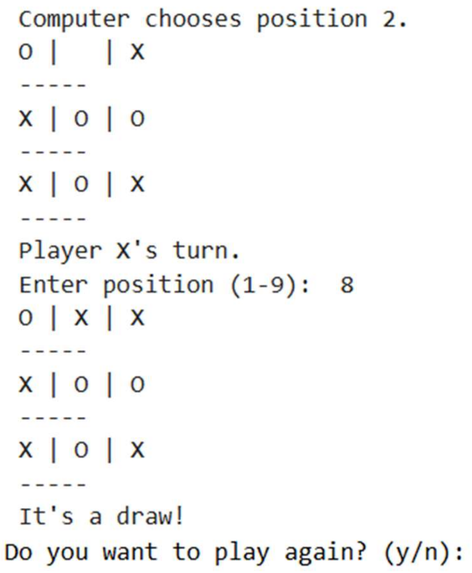
Design

The Tic Tac Toe game is designed to be an interactive console-based application in Python. The game offers two modes: Player vs Player (PVP) and Player vs Computer (PVC). In PVP mode, two players alternate turns to place 'X' and 'O' on a 3x3 grid, aiming to form three consecutive marks in a row, column, or diagonal. In PVC mode, the player competes against a computer, with the computer's moves either being random (Moderate difficulty) or strategic (Hard difficulty).

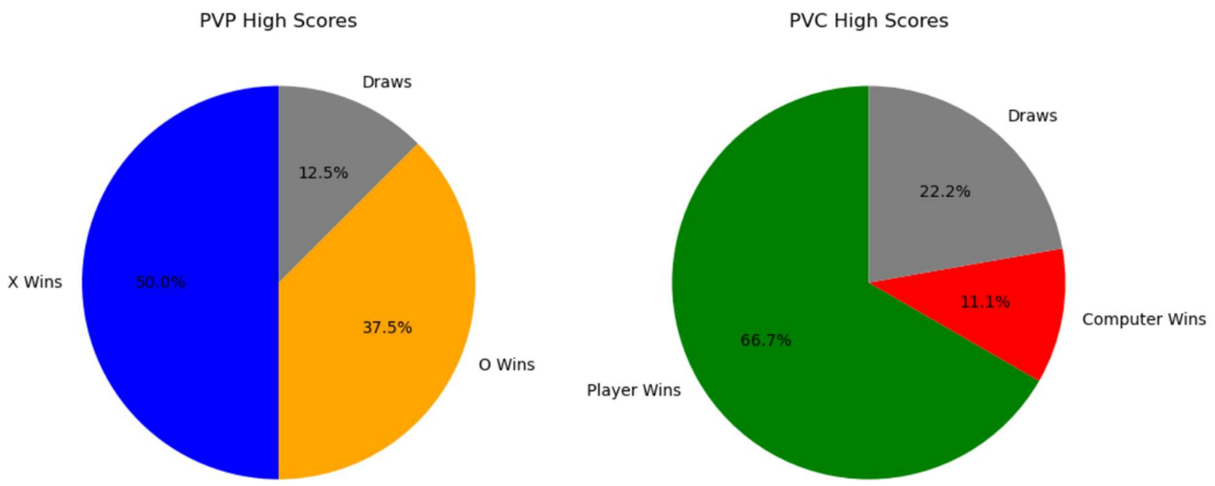
The game board is represented as a 2D list, and after each move, the board is updated and displayed to the players. Input validation ensures that only valid moves are allowed. If a player wins, the game announces the winner, and if the board is full with no winner, a draw is declared. The game tracks and updates high scores for both modes, storing them in a JSON file.

For visual enhancement, the high scores are displayed using pie charts generated by matplotlib. These charts provide a graphical representation of the win and draw statistics for both PVP and PVC modes. The game logic is divided into functions for handling game flow, board display, input validation, winner checking, and high score management, ensuring modularity and clarity.

Implementation



```
 Computer chooses position 2.
 O |   | X
 -----
 X | O | O
 -----
 X | O | X
 -----
 Player X's turn.
 Enter position (1-9):  8
 O | X | X
 -----
 X | O | O
 -----
 X | O | X
 -----
 It's a draw!
Do you want to play again? (y/n):
```



Challenges and Solution

- Storage of High Score:
  Storing and retrieving high scores while ensuring the integrity of the JSON file, especially if the file becomes corrupted or is missing. The program uses a JSON file to save the high scores after each game. It includes error handling to check if the file exists or is corrupted. If the file is missing or corrupted, the game resets the high scores to default values, ensuring the game continues to function smoothly.
- Generating Charts:
  Displaying high scores in a visually appealing manner, making it easy for players to understand the game results. The program uses the matplotlib library to create pie charts that represent the win and draw statistics for both PVP and PVC modes. This visual representation provides an engaging way to display the scores and enhances the user experience.

Future Improvements

- Enhanced AI
- GUI
- Customizable Game Board and Pieces
- Leaderboard System
- Mobile Version

Conclusion

The Tic Tac Toe game, with its various modes and features, provides an engaging and interactive experience for both single-player and multiplayer gameplay. Through careful implementation, challenges such as input validation, storing high scores, and creating visual representations of scores using charts have been effectively addressed. The integration of AI with different difficulty levels enhances the gameplay, making it more dynamic and challenging for users.

Additionally, the use of a JSON file to save and retrieve high scores ensures that player data is retained between sessions, while the graphical representation of scores adds an interesting visual element. Though the game is simple, it lays the foundation for potential future improvements, including more sophisticated AI, graphical interfaces, and online multiplayer capabilities.

Overall, this project demonstrates a well-structured approach to developing a game that is both fun and functional, with room for expansion and enhancement in the future.

References

- GeeksforGeeks article on implementing a Tic Tac Toe game in Python, which provides a detailed guide and code for building the game and AI. (GeeksforGeeks - Tic Tac Toe AI)
- Python's official documentation for working with JSON, which provides tools for parsing and writing JSON data in Python. (Python JSON Documentation)
- Matplotlib's official documentation, which guides users on how to create plots and visualizations in Python. (https://matplotlib.org/stable/contents.html)