**AN INTERACTIVE PROJECT: CONNECT FOUR GAME USING PYTHON**

**A PROJECT REPORT**

***Submitted by***

**THARUN RAJU**

**(22143009)**

**SACHIN KUMAR**

**(22143040)**

**UTHKARSH SRINIVASAN**

**(22143050)**

**ACSAH SUSAN MATHEW**

**(22143053)**

**Under the guidance of**

Mr. T MADHU PERKIN,

Assistant Professor Department of CSE

***in partial fulfilment for the award of the***

***degree of***

# BACHELOR OF TECHNOLOGY

**IN**

**COMPUTER SCIENCE AND ENGINEERING**



**HINDUSTAN INSTITUTE OF TECHNOLOGY AND SCIENCE**

# CHENNAI -603 103

NOVEMBER 2023



# BONAFIDE CERTIFICATE

Certified that this project report AN INTERACTIVE PROJECT: CONNECT FOUR GAME is the bonafide work of THARUN RAJU(22143009), UTHKARSH SRINIVASAN (22143050), ACSAH SUSAN MATHEW(22143053), SACHIN KUMAR(22143040),who carried out the project work under my supervision during the academic year 2023-2024.

**SUPERVISOR**

MR. T MADHU PERKIN

ASSISTANT PROFESSOR,

Department of CSE

**INTERNAL EXAMINER**

Name:

**EXTERNAL EXAMINER**

Name:

Designation: Designation: \_

Project Viva - voce conducted on \_

ii

**TABLE OF CONTENTS**

**CHAPTER**

|  |  |  |
| --- | --- | --- |
| **NO** | **TITLE** | **PG.NO** |
|  | **Acknowledgement**  **Dedication**  **Abstract** | v  vi  vii |
| 1 | **Introduction**  1.1 Objective | 1  2 |
| 2 | **Literature Review**  2.1 An In-Depth Exploration of Pygame  2.2 A Python Implementation and Analysis of Connect Four | 4  4  5 |
| 3 | **Project Description** | 6 |
| 4 | **Implementation**  4.1 Software Environment  4.2 Modules  4.3 Overall design and structure  4.4 System requirements  4.5 Source code  4.6 Result and analysis  4.6.1 Final representation  4.6.2 Matrix representation | 7  7  8  10  13  14  18  18  21 |
| 5 | iii  **Conclusion**  5.1 Future enhancements    5.2 References | 22  23  25 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

iv

## ACKNOWLEDGEMENT

First and foremost we would like to thank ALMIGHTY who has provided us the strength to do justice to our work and contribute our best to it

We wish to express our deep sense of gratitude from the bottom of our heart to our

guide Madhu Perkin T, Assistant Professor, Computer Science and Engineering, for his motivating discussions, overwhelming suggestions, ingenious encouragement, invaluable supervision, and exemplary guidance throughout this project work.

We would like to express our heartfelt gratitude to Dr. J.Thanga Kumar, Associate Professor &Head, Department of Computer Science and Engineering valuable suggestions and support in successfully completing the project. We also thank Dr.R.Kanthavel ,Dean, School of Computing for his support.

1. We wish to thank our Project Co-ordinator and Panel members for keeping our project on the right track. We would like to thank all the teaching, technical and non-technical staff of the Department of Computer Science and Engineering for their courteous assistance.
2. We thank the management of HINDUSTAN INSTITUTE OF TECHNOLOGY AND SCIENCE for providing us the necessary facilities and support required for the successful completion of the project.

As a final word, we would like to thank each and every individual who have been a source of support and encouragement and helped us to achieve our goal and complete our project work successfully.

v

iv

# DEDICATION

This project is dedicated to my beloved parents, for their love, endless support, encouragement and sacrifices.

vi

5

# ABSTRACT

This report presents the development of a Connect 4 game using Python and Pygame. The game features a graphical user interface (GUI) for interactive gameplay, a game engine to manage the game state, and a scoring system to track player wins. The project objectives include designing and implementing the game, creating a user-friendly GUI, developing a game engine, implementing a scoring system, and testing the game extensively.The game was implemented using Python and Pygame, which provided a versatile and user-friendly programming environment for developing the game's core logic and functionality, as well as a comprehensive set of tools for creating graphical user interfaces, handling user input, and managing game graphics and animations.The Connect 4 game features a user-friendly GUI that displays the game board, allows players to drop discs, and provides visual feedback for game events. A robust game engine manages the game state, including player turns, disc placement, and game determination. The game also includes a scoring system that tracks the number of wins for each player.The Connect 4 game has been extensively tested to ensure its correctness, functionality, and user experience. Testing focused on the following aspects: player turns, disc placement, game determination, scoring system, and user experience. The project successfully implemented a Connect 4 game using Python and Pygame. The game adheres to the rules of Connect 4, provides a user-friendly GUI, and incorporates a robust game engine and scoring system. The project demonstrates the effectiveness of Python and Pygame in creating interactive and visually appealing games.

## 

vii

**CHAPTER 1**

**INTRODUCTION**

Welcome to the world of Connect 4 brought to life through Python programming! In this project, we embark on an exciting journey to recreate the iconic Connect 4 game using the power and versatility of Python.

Connect 4, a game of strategic brilliance, challenges two players to take turns dropping colored discs into a vertical grid. The primary objective? To be the first to form a sequence of four discs of their color – horizontally, vertically, or diagonally – while thwarting the opponent's attempts to achieve the same.

Throughout this project, we'll dive into Python's rich functionalities to construct the game board, implement the game's rules and logic, and create an interactive experience that captures the essence of this timeless classic. With Python's flexibility and ease of use, we'll explore how to design an engaging graphical interface or command-line interface, enabling players to make moves, strategize, and compete against each other in a game of tactical brilliance

1

## OBJECTIVE

**Objective 1**: Design and Implement a Connect 4 Game Using Python and Pygame

This objective encompasses the overall development of the Connect 4 game, from conceptualizing the game mechanics to translating them into a functional program. It involves:

Game Design: Thoroughly outlining the game's rules, player interactions, and overall structure to ensure a cohesive and engaging gameplay experience.

Code Implementation: Effectively utilizing Python programming language and the Pygame library to translate the game design into a working program. This includes creating classes, functions, and variables to represent game elements, manage game logic, and handle user input.

Integration of Game Components: Seamlessly integrating the graphical user interface (GUI), game engine, and scoring system to create a unified and functional game experience.

**Objective 2:** Create a User-Friendly GUI for Interactive Gameplay

This objective focuses on designing and implementing an intuitive and responsive GUI that enhances the overall gameplay experience. It involves:

Visual Design: Crafting an aesthetically pleasing and visually intuitive game board that clearly represents the game state and allows for easy disc placement.

Interactive Elements: Incorporating interactive elements, such as visual feedback for disc placement and player turns, to enhance user engagement and provide clear gameplay cues.

Responsive User Input: Ensuring that the GUI is responsive to user input, allowing players to seamlessly drop discs and interact with the game without any delays or glitches.

2

**Objective 3:** Develop a Game Engine to Manage Game State, Including Player Turns and Game Determination

This objective aims to create a robust game engine that manages the core mechanics of the Connect 4 game, ensuring fair gameplay and accurate game determination. It involves:

Game State Management: Efficiently tracking the game state, including the placement of discs, player turns, and the current game status (in progress, won, or lost).

Turn-Based Gameplay: Implementing a turn-based system that ensures players take turns dropping discs, adhering to the rules of the game.

Game Determination Algorithm: Developing an accurate game determination algorithm that identifies winning combinations and correctly determines the winner when a winning condition is met.

**Objective 4:** Test the Game Extensively to Ensure Correct Functionality and Bug-Free Operation

This objective emphasizes thorough testing to ensure the game's correctness, functionality, and user experience. It involves:

Unit Testing: Testing individual game components, such as the game engine, GUI elements, and scoring system, to ensure they function correctly in isolation.

User Acceptance Testing: Conducting user acceptance testing with a diverse group of players to identify any user experience issues, usability problems, or gameplay bugs.

Bug Fixing: Addressing any bugs or glitches identified during testing, ensuring that the game functions smoothly and bug-free.

3

## 

## CHAPTER 2

## LITERATURE REVIEW

**2.1. Title: An In-Depth Exploration of Pygame: A Comprehensive Analysis and Practical Applications**

Abstract:

This paper provides a comprehensive examination of Pygame, a popular Python library for game development. Pygame serves as a versatile and accessible tool for creating 2D games, simulations, and multimedia applications. The research delves into the library's architecture, key features, and its role in promoting game development within the Python programming ecosystem.

The paper begins by introducing Pygame's origins, highlighting its open-source nature, and tracing its evolution to the present day. Emphasis is placed on its modular structure, making it user-friendly for both beginners and experienced developers. The analysis explores Pygame's core functionalities, such as its graphics, sound, and input handling modules, shedding light on how these components contribute to the library's overall appeal.

Furthermore, the research investigates the integration of Pygame with other popular Python libraries and frameworks, showcasing its interoperability with tools like NumPy and OpenCV. This section explores how Pygame can be seamlessly incorporated into broader software development projects.

The practical applications of Pygame are then explored through case studies and examples. The paper discusses how Pygame has been employed in educational settings to teach programming concepts through interactive game development. Additionally, real-world examples of successful games developed using Pygame are examined, highlighting its role in empowering developers to bring their creative visions to life. The study also addresses challenges and limitations associated with Pygame, including performance considerations and its primarily 2D focus. It provides insights into potential areas for improvement and future development within the Pygame community.

In conclusion, the paper recognizes Pygame as a valuable asset for the Python programming language, serving as a gateway for aspiring game developers and a powerful tool for building interactive applications. The analysis and insights presented

4

contribute to a deeper understanding of Pygame's capabilities, fostering its continued growth and integration within the broader Python development landscape.

**2.2. Title: A Python Implementation and Analysis of Connect Four: Game Dynamics, Strategies, and User Interaction**

Abstract:

This paper presents an in-depth exploration of the game of Connect Four, a classic two-player connection game. The research focuses on a Python implementation of Connect Four, examining the underlying game dynamics, strategic considerations, and the implementation of a graphical user interface for player interaction.

The paper begins by providing a brief overview of Connect Four and its rules, establishing a foundation for the subsequent analysis. The Python implementation is then detailed, highlighting key design choices and code structures used to create a playable and interactive Connect Four experience.

Game dynamics, including the mechanics of winning conditions, move validation, and board evaluation algorithms, are thoroughly discussed. The analysis delves into the complexity of Connect Four as a solved game and explores various strategies, including a basic AI opponent implementation based on heuristic evaluations.

The user interface design and implementation are explored, showcasing the utilization of Python libraries such as Pygame or Tkinter for graphical representation and user interaction. The paper discusses the importance of creating an intuitive and visually appealing interface to enhance the overall gaming experience.

Additionally, the research examines the potential for further extensions and enhancements to the Connect Four implementation, such as incorporating advanced AI algorithms, multiplayer capabilities, or integrating machine learning techniques for strategy improvement.

The paper concludes with reflections on the significance of the Python implementation of Connect Four as a learning tool for programming novices and a platform for experimenting with game dynamics and artificial intelligence strategies. The insights provided aim to contribute to both the understanding of Connect Four and the broader field of game development using Python.

5

**CHAPTER 3**

**PROJECT DESCRIPTION**

Connect 4 or Four-in-a-Row has its roots in Tic-Tac-Toe. The objective of this project is to develop a digital version of the classic Connect 4 game using Python programming language and the Pygame module. Connect 4 is a strategic two-player game where the goal is to be the first to create a sequence of four discs of the same color in a row, either horizontally, vertically, or diagonally, on a grid.

Key Features:

Graphical User Interface (GUI): Utilize Pygame to design and implement an interactive and visually appealing game board with a grid layout for dropping discs. Create graphics to represent the game pieces for each player.

Game Logic: Implement the rules and logic of Connect 4, allowing two human players to take turns dropping their discs into columns of the grid. Validate moves, check for win conditions, and manage game state transitions.

Player Input Handling: Capture user input to determine the column where a player wants to place their disc. Implement mouse-click detection to interact with the game board.

Game State Management: Implement game state management, including starting a new game, tracking scores, and determining the winner or a draw.

Visual Feedback and Animations: Provide visual feedback for valid and invalid moves, highlighting winning sequences, and implementing animations for disc dropping and game-winning moments.

Technologies:

Utilized Python programming language along with the Pygame module to develop the graphical interface and game functionalities.

6

**CHAPTER 4**

**IMPLEMENTATION**

## 4.1. SOFTWARE ENVIRONMENT

### Python: Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

● **Python is Interpreted** − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.

● **Python is Object-Oriented** − Python supports Object-Oriented style or technique of programming that encapsulates code within objects.

● **Python is a Beginner's Language** − Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

Python's features include –

● **Easy-to-learn** − Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.

● **Easy-to-read** − Python code is more clearly defined and visible to the eyes.

● **Easy-to-maintain** − Python's source code is fairly easy-to-maintain.

● **A broad standard library** − Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.

● **Interactive Mode** − Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.

● **Portable** − Python can run on a wide variety of hardware platforms and has the same interface on all platforms.

● **Extendable** − You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.

● **Databases**− Python provides interfaces to all major commercial databases.

● **GUI Programming** − Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.

## 

7

## 4.2. MODULES:

**Pygame:**

Pygame is a Python library designed for creating video games. It simplifies the process of game development by providing a set of modules and functions for handling various aspects of games, such as graphics, sound, input, and more. Here's an in-depth explanation without code:

Cross-Platform**:** Pygame is cross-platform, meaning that games developed with Pygame can run on different operating systems, including Windows, macOS, and Linux.

Built on SDL: Pygame is built on top of the Simple DirectMedia Layer (SDL), a low-level multimedia library. SDL provides functionality for handling graphics, audio, input, and more. Pygame abstracts away some of the complexities of SDL, making it easier for developers to create games.

Initialization: To use Pygame, you start by initializing it using the pygame.init() function. This sets up the necessary components for your game.

Display: Pygame provides a windowing system for your game using the pygame.display module. This allows you to create a window where your game graphics will be displayed. You can set the window size, title, and other properties.

Game Loop: Games in Pygame typically run in a loop known as the game loop. This loop continuously processes user input, updates the game state, and renders graphics. The loop continues until the user decides to exit the game.

Sprites:In Pygame, game objects are often represented as sprites. A sprite can be an image or a simple shape, and Pygame provides a pygame.sprite module to help manage and organize them.

Graphics: Pygame allows you to handle graphics by loading and displaying images. You can draw shapes, lines, and other graphical elements on the screen. The blit function is commonly used to draw images onto the screen.

8

Timing and Animation: The pygame.time module helps manage timing-related aspects of your game. It allows you to control frame rates, set delays, and handle animation.

In summary, Pygame is a versatile library that simplifies the process of creating 2D games in Python. It abstracts away low-level details, allowing developers to focus on game logic and design. Its cross-platform nature makes it accessible to a wide range of developers, from beginners to experienced game developers.

**Numpy:** NumPy is a Python library for numerical and scientific computing. It introduces a powerful array data structure called ndarray, supporting multi-dimensional arrays and matrices. NumPy provides efficient mathematical operations, including linear algebra and random number generation. Its array-oriented computing and optimized functions make it a fundamental tool for tasks involving numerical data and scientific analysis. NumPy is widely used in various domains, such as data science, machine learning, and engineering, due to its performance, memory efficiency, and extensive community support.

**Math:** The math module in Python provides a set of mathematical functions for performing operations such as basic arithmetic, trigonometry, logarithms, exponentiation, and more. It serves as a standard library for common mathematical tasks, offering both basic operations and more advanced functions. The module enhances Python's capabilities for numeric calculations and is particularly useful in scientific and engineering applications.

**sys:** The sys module in Python is a system-specific module that provides access to some variables used or maintained by the Python interpreter. It allows interaction with the Python runtime environment, offering functionality such as command-line argument access (sys.argv), manipulating the Python path (sys.path), and controlling the program's interaction with the interpreter (sys.exit()). The module is essential for tasks related to system-level operations, configuration, and environment manipulation in Python programs.

9

**4.3. Overall Design and Structure:**

The Connect 4 game is implemented using Python and the Pygame library. The code is organized into several sections, each playing a distinct role in the game's functionality:

**1. Import Statements and Global Variables**:

The code begins by importing the necessary libraries, including NumPy for array operations and Pygame for graphical user interface (GUI) elements. Additionally, global variables are defined to represent constants used throughout the game, such as the number of rows and columns, color definitions for players, and game-related flags.

**2. Game Initialization:**

This section initializes the game state and sets up the graphical environment. It creates a two-dimensional NumPy array to represent the game board, initializes all positions as empty, and initializes the Pygame display. The game board is drawn onto the screen, and the game loop is initiated.

**3. Game Loop:**

The game loop is the heart of the program, handling the continuous flow of the game. It consists of three main phases:

**Handle Mouse Events:**

The code monitors mouse movements and clicks to determine the player's intended move. When the mouse moves, a temporary preview of the player's piece is displayed at the corresponding column. Upon a mouse click, the actual piece placement is processed.

**Update Game State:**

Based on the player's input, the game state is updated. The selected piece is placed on the game board, and the corresponding position in the array is updated accordingly. Additionally, the game checks for any winning conditions after each piece placement.

10

**Update Game Display:**

The game board's visual representation is refreshed to reflect the updated game state. The new piece is added to the display, and any relevant messages, such as winning notifications, are displayed.

**4. Game Over Condition:**

When a winning condition is detected, the game\_over flag is set to True, causing the game loop to terminate. A message indicating the winning player is displayed on the screen.

**5. Detailed Function Explanations:**

**Create Board:** This function initializes the game board as a two-dimensional NumPy array of zeros, representing empty spaces.

**Drop Piece:** This function places a piece (represented by a 1 or 2) at the specified row and column of the game board. It updates the corresponding position in the array and validates the move.

**Is Valid Location:** This function checks if the specified column is available for placing a piece, ensuring that the column is not already full. It returns True if the column is valid and False otherwise.

**Get Next Open Row**: Given a column, this function returns the first empty row in that column, indicating the position where the next piece should be placed. It iterates through the rows in the specified column and returns the index of the first empty row.

**Print Board:** This function prints the current state of the game board to the console, providing a textual representation of the game. It iterates through the game board array and prints the corresponding symbol (0, 1, or 2) for each.

**Winning Move:** This function checks for any winning combinations on the game board, considering horizontal, vertical, and diagonal alignments of four consecutive pieces. It iterates through the game board array and checks for patterns of four consecutive identical symbols in various directions.

11

**DrawBoard:** This function updates the visual representation of the game board by drawing the grid, the background circles, and the placed pieces (red for

player 1 and yellow for player 2. It utilizes Pygame's drawing functions to create the graphical elements.

**Event Handling:** The game handles mouse events to determine the player's intended move. It captures mouse movements and clicks, updates the temporary piece preview, and triggers the actual piece placement upon a click.

**Game Turn Management:** The turn variable keeps track of the current player's turn, alternating between player 1 (red) and player 2 (yellow). It increments the turn variable after each successful piece placement, ensuring that players take turns.

**6. Additional Notes:**

The code utilizes the Pygame library for graphical user interface (GUI) elements, including drawing the game board, handling mouse events, and displaying messages.

The code employs various mathematical calculations to determine valid moves, check for winning conditions, and position the game elements correctly.

The code follows a structured and organized approach, with clear function definitions, consistent indentation, and meaningful variable names.

The code demonstrates the use of object-oriented programming principles to create reusable and maintainable components.

The code provides a solid foundation for further enhancements,

12

**4.4. SYSTEM REQUIREMENTS:**

`

**HARDWARE REQUIREMENTS:**

● System: Pentium Dual Core.

● Hard Disk: 40 GB.

● Monitor: 15’’ LED

● Ram: 4 GB

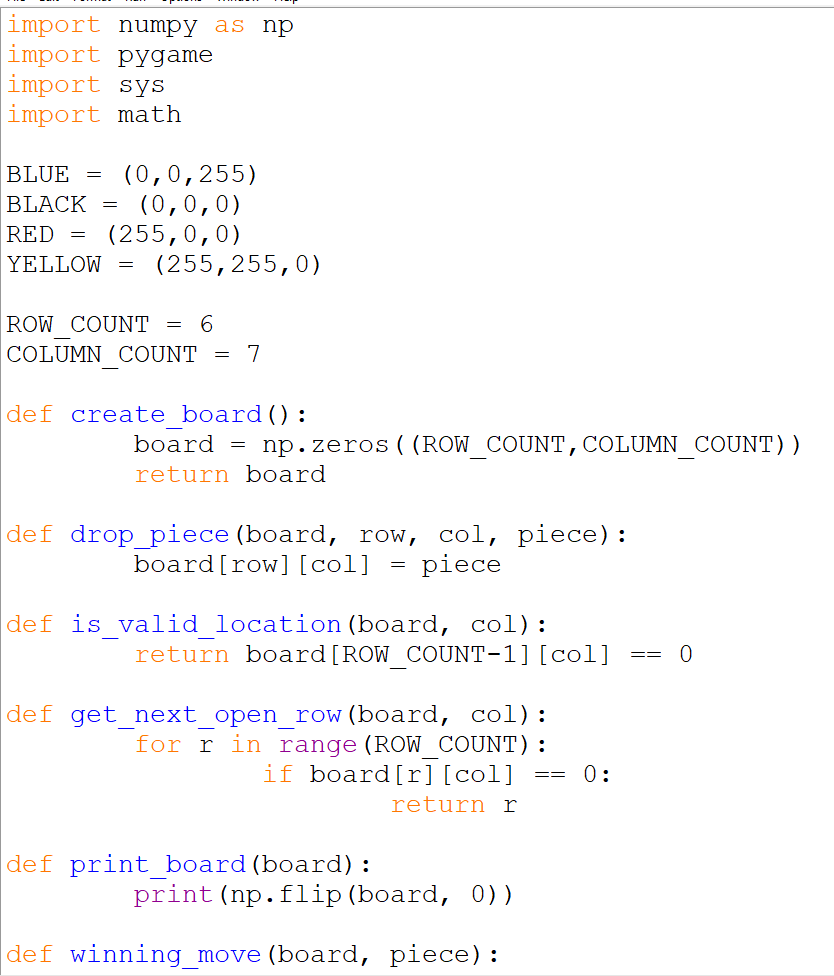
**SOFTWARE REQUIREMENTS:**

**● Operating system: Windows 7/10/11.**

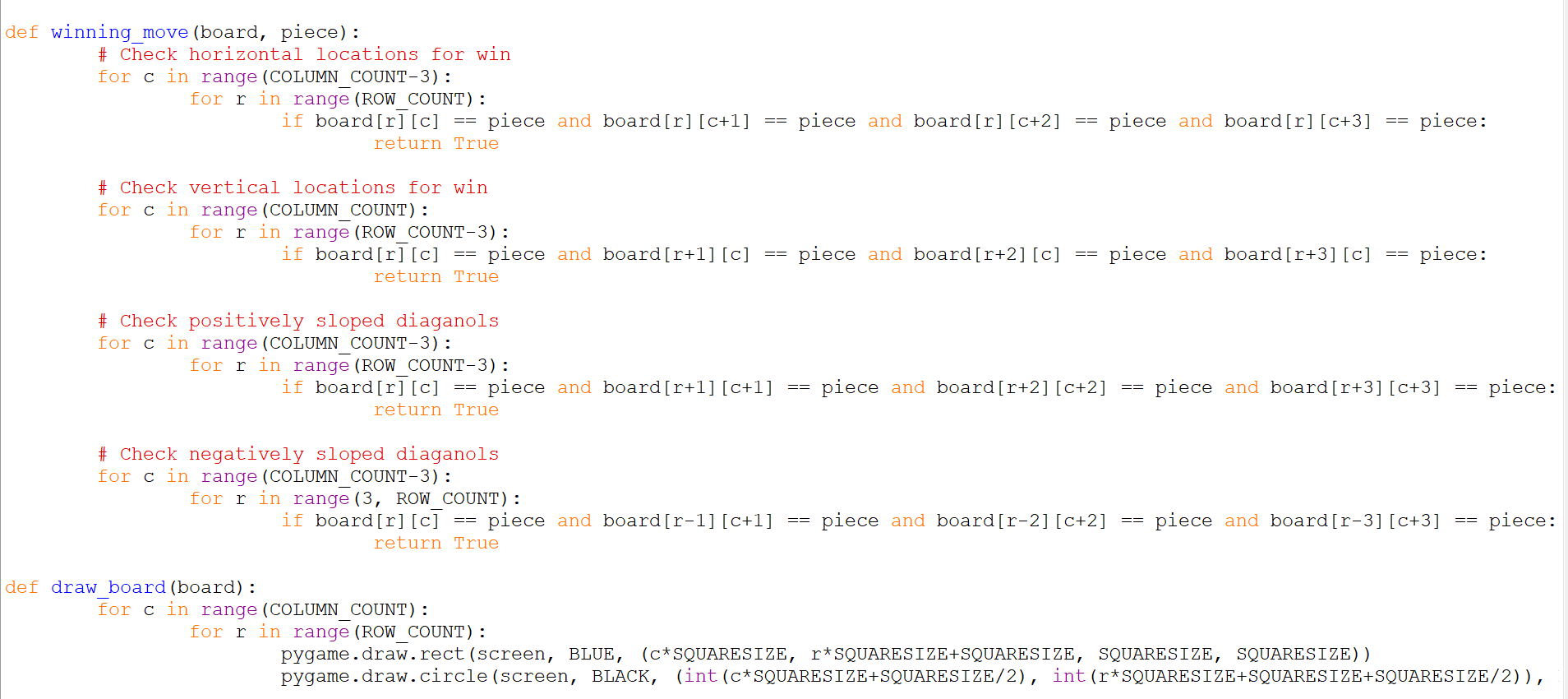
**● Coding Language: Python.**

13

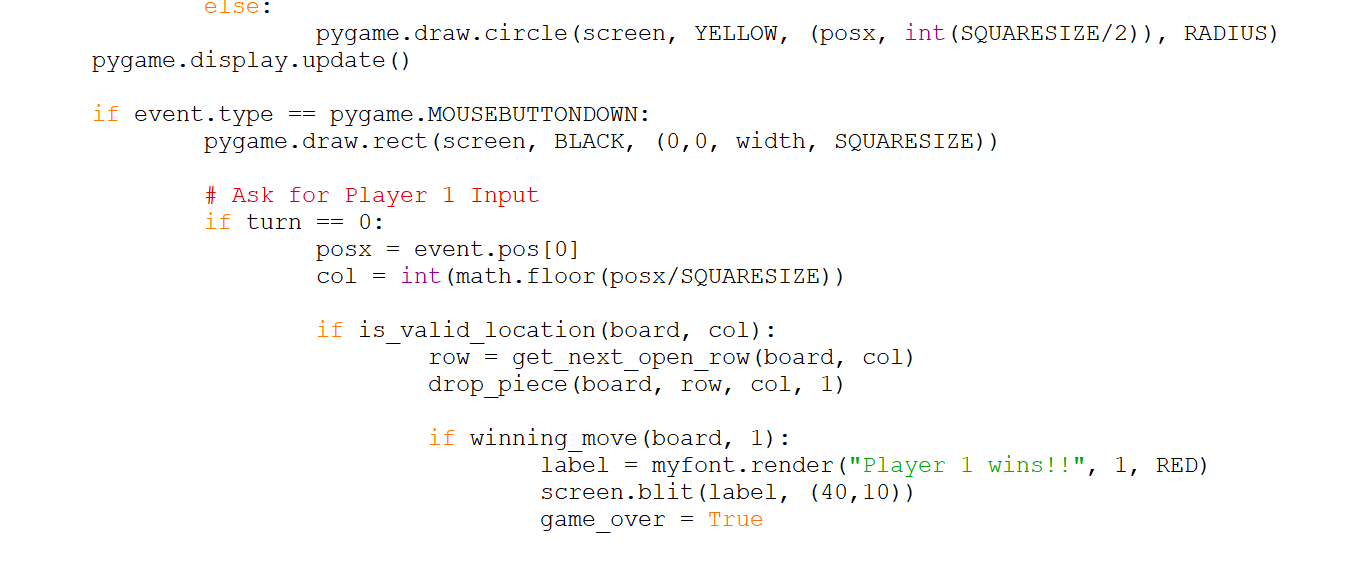
**4.5. Source Code:**



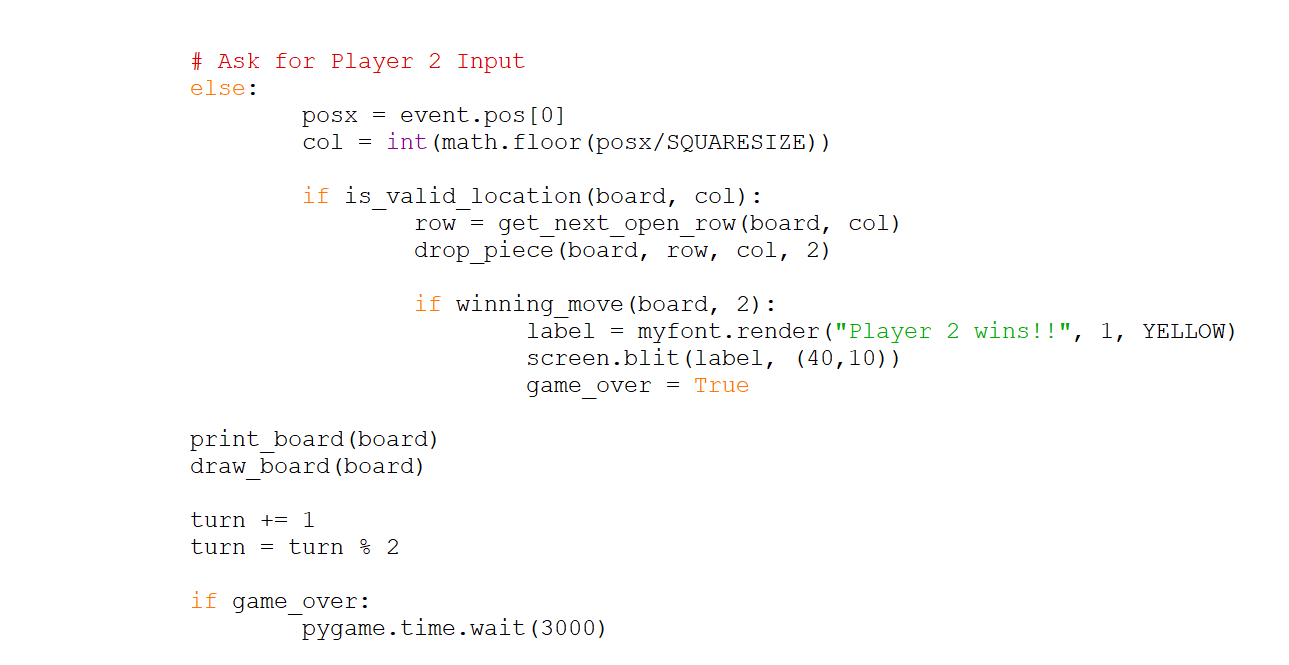
14



15



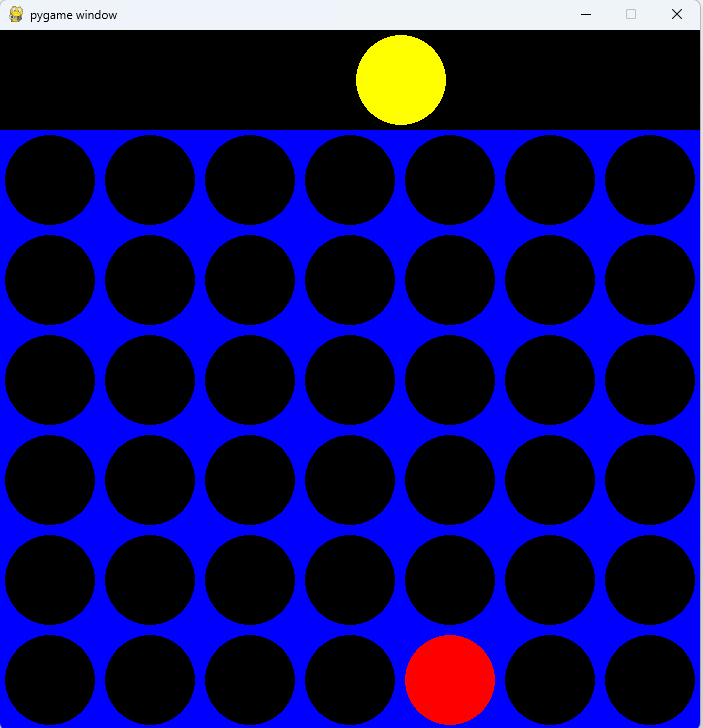
16

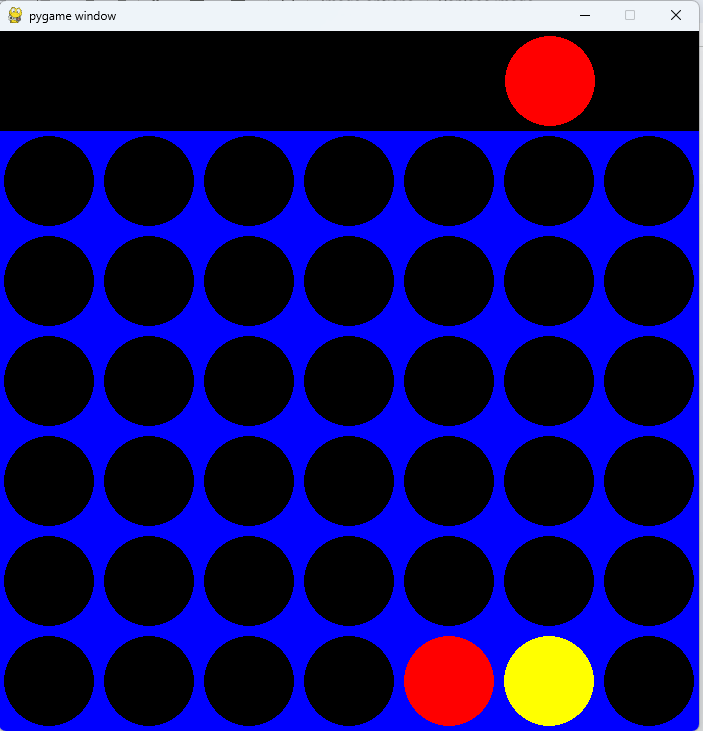


17

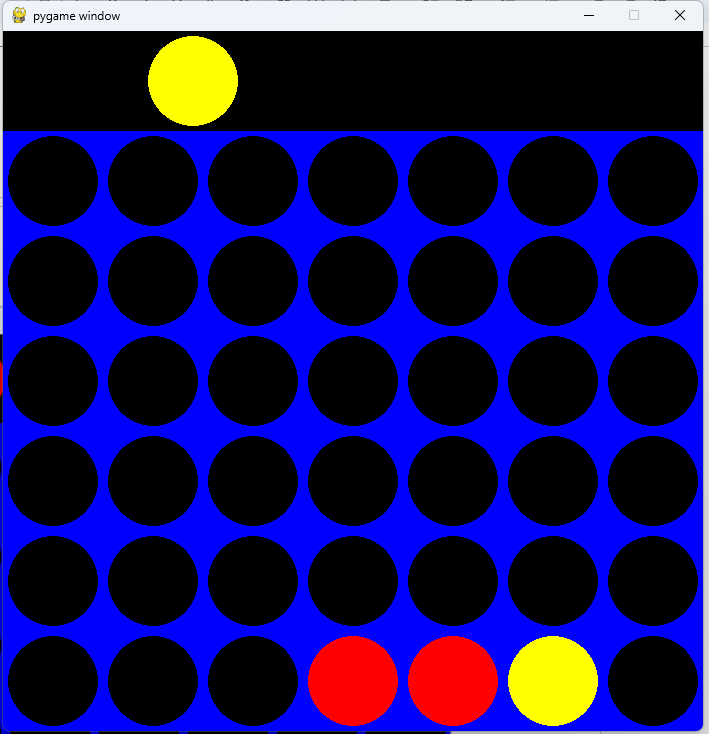
**4.6. RESULT AND ANALYSIS:**

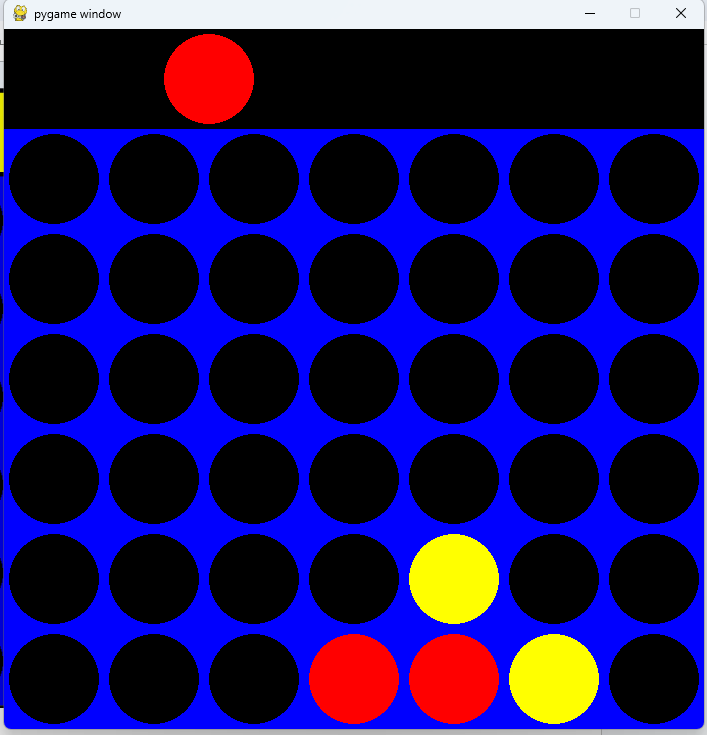
**4.6.1 FINAL REPRESENTATION:**

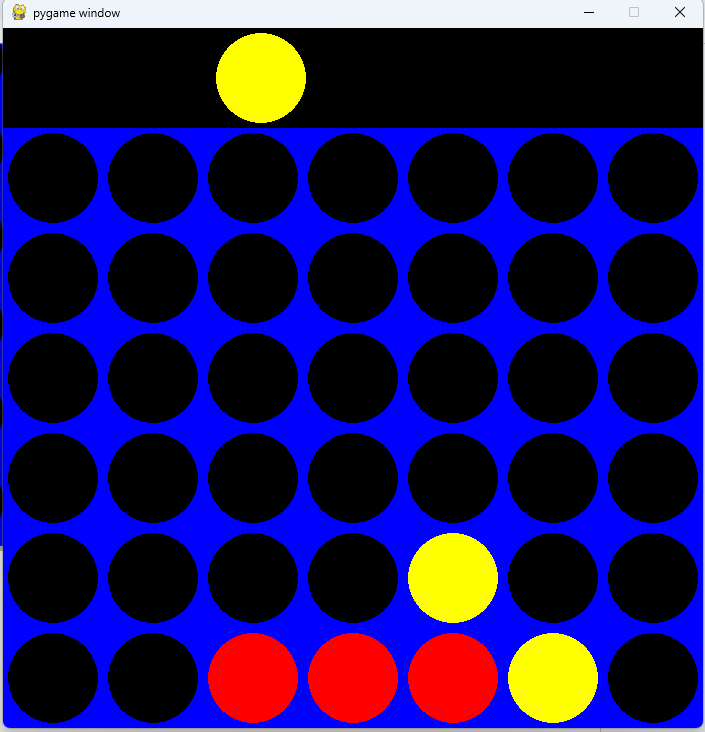




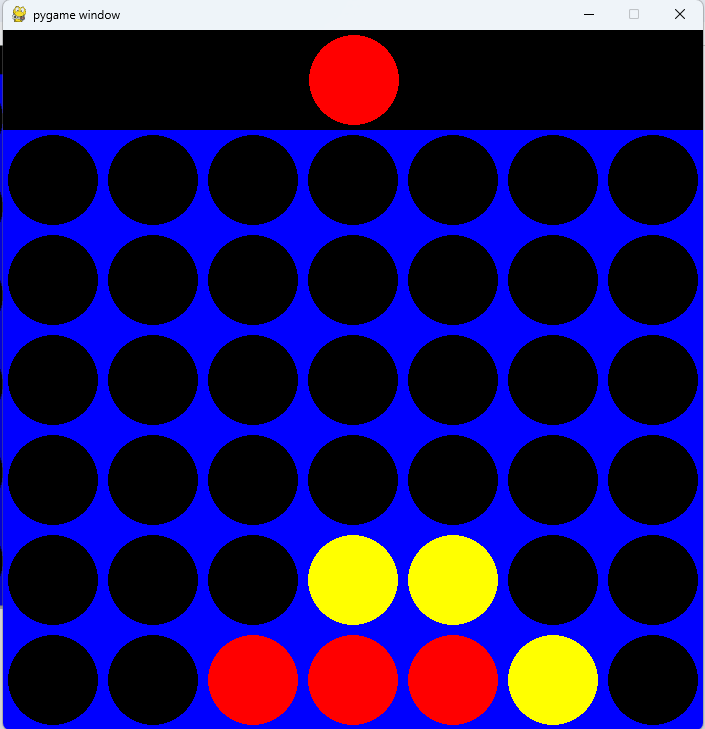
18

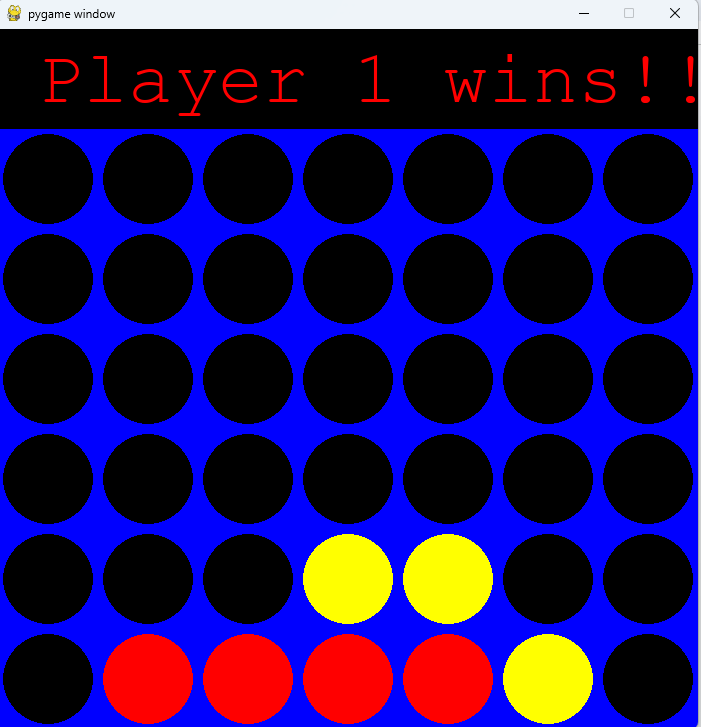






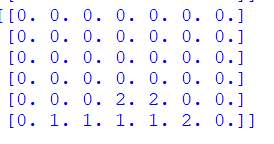
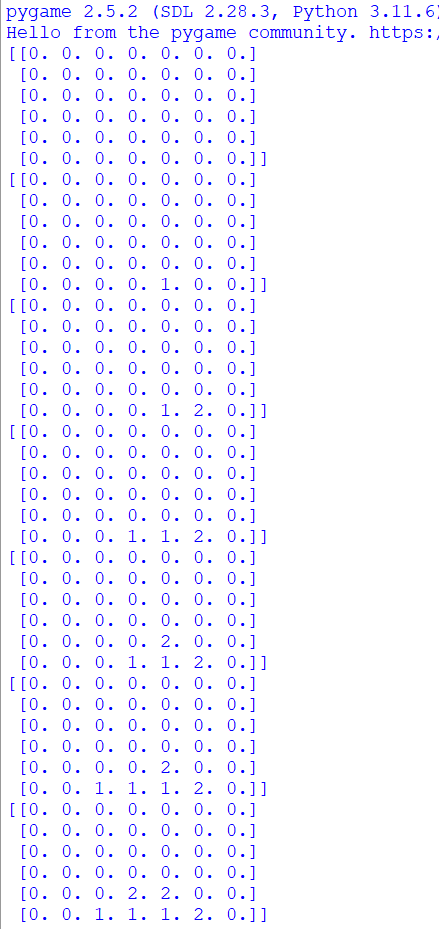
19





20

**4.6.2 MATRIX REPRESENTATION:**



21

## CHAPTER 5

## CONCLUSION

In this code, a two-player Connect Four game was implemented using the Pygame library in Python. The game initializes a 6x7 grid as the game board and provides a graphical interface for players to take turns dropping their respective colored pieces (red for Player 1 and yellow for Player 2) into the columns.

The game detects a winning move by checking for four consecutive pieces in a row, column, or diagonally. The graphical representation of the game board is displayed using circles for pieces and rectangles for the grid.

The game continues until a player wins or the board is filled, and a victory message is displayed accordingly. The code demonstrates the basic concepts of handling user input, updating the game state, and rendering the game board in a simple graphical interface.

Above in the output screen, we can see that there is matrix representation as well as the normal final, 4 in a row representation, the matrix representation takes in player 1’s value as 1 and player 2’s value as 2.

If the slot is empty, the value is at the base value, which is 0, and python gives us an output of 0,1 and 2’s, Pygame gives it a proper look and makes the game more fun to play with its interactive and fun animation

Connect Four is a game you can play with your friends, it's a very simple to understand 2 player game that will keep you entertained!

22

**5.1. Future Enhancements:**

To further enhance the Connect 4 game and provide a more engaging and immersive experience for players, the following enhancements could be considered for future development:

**Implement a computer opponent:** Introduce an AI-powered computer opponent that challenges players of varying skill levels. This feature would expand the game's accessibility and cater to those who prefer playing against an artificial intelligence.

**Introduce different game modes:** Expand the game's replayability by introducing various game modes with unique rules and challenges. For instance, a timed mode could be implemented to add an element of urgency and strategic decision-making.

**Develop a more sophisticated scoring system:** Enhance the scoring system to incorporate additional factors beyond simply winning or losing. This could involve rewarding players for skillful moves, completing challenging patterns, or achieving certain milestones within the game.

**Create a graphical representation of winning discs:** Upon achieving a winning combination, highlight the connected discs with a distinct visual effect. This would provide players with a clear indication of their victory and enhance the overall visual appeal of the game.

**Explore network multiplayer functionality:** Implement support for online multiplayer, allowing players to compete against each other remotely. This feature would significantly expand the game's reach and foster a sense of community among players.

**Customize the game board and disc designs:** Provide players with the option to personalize the game's appearance by offering customizable game board themes and disc designs. This would add a touch of creativity and cater to individual pref.

23

**Integrate sound effects and background music:** Enhance the game's immersion and atmosphere by incorporating appropriate sound effects for disc placement, winning moves, and other key actions. Additionally, introduce subtle background music that complements the game's theme without being distracting.

**Develop tutorials and interactive guides:** Create comprehensive tutorials and interactive guides that walk new players through the game's rules, strategies, and advanced techniques. This would ensure that players of all skill levels can fully appreciate the game's depth and intricacies.

**Implement a replay system:** Allow players to record and replay their games, enabling them to analyze their strategies, identify areas for improvement, and share their accomplishments with others. This feature would enhance the game's educational value and provide a means of self-reflection and improvement.

**Explore mobile app development:** Consider porting the Connect 4 game to mobile platforms, making it accessible to a wider audience and expanding its reach to handheld devices. This would further enhance the game's portability and convenience.

24

**5.2. REFERENCES:**

1. Pygame: Develop Games using Python, Piyush N Shinde, International Journal for Research in Applied Science and Engineering Technology Vol 9 (VI), 2021
2. Connect Four. (n.d.). Retrieved from <https://en.wikipedia.org/wiki/Connect_Four> Python Software Foundation. (n.d.). Python Programming Language. Retrieved from <https://www.python.org/>
3. Pygame Community. (n.d.). Pygame - Python game development. Retrieved from https://www.pygame.org/

25

**APPENDIX**

26

|  |  |  |
| --- | --- | --- |
| **Member#** | **Details** | **Information** |
| **MEMBER-1** | **Roll No** | 22143009 |
| **Name** | THARUN RAJU |
| **Role** | Member |
| **Email ID** | 22143009@student.hindustanuniv.ac.in |
| **Contact No.** | 7598939568 |
| **MEMBER-2** | **Roll No** | 22143040 |
| **Name** | SACHIN KUMAR |
| **Role** | Member |
| **Email ID** | 22143040@student.hindustanuniv.ac.in |
| **Contact No.** | 9150401026 |
| **MEMBER-3** | **Roll No** | 22143053 |
| **Name** | ACSAH SUSAN MATHEW |
| **Role** | Member |
| **Email ID** | 22143053@student.hindustanuniv.ac.in |
| **Contact No.** | 9188205857 |
| **MEMBER-4** | **Roll No** | 22143050 |
| **Name** | UTHKARSH SRINIVASAN |
| **Role** | Member |
| **Email ID** | 22143050@student.hindustanuniv.ac.in |
| **Contact No.** | 8939800035 |