

# Connect Four: A Python Gaming Application

Team members: Aadya Gupta, Anika Nair, Khushi Gojanur

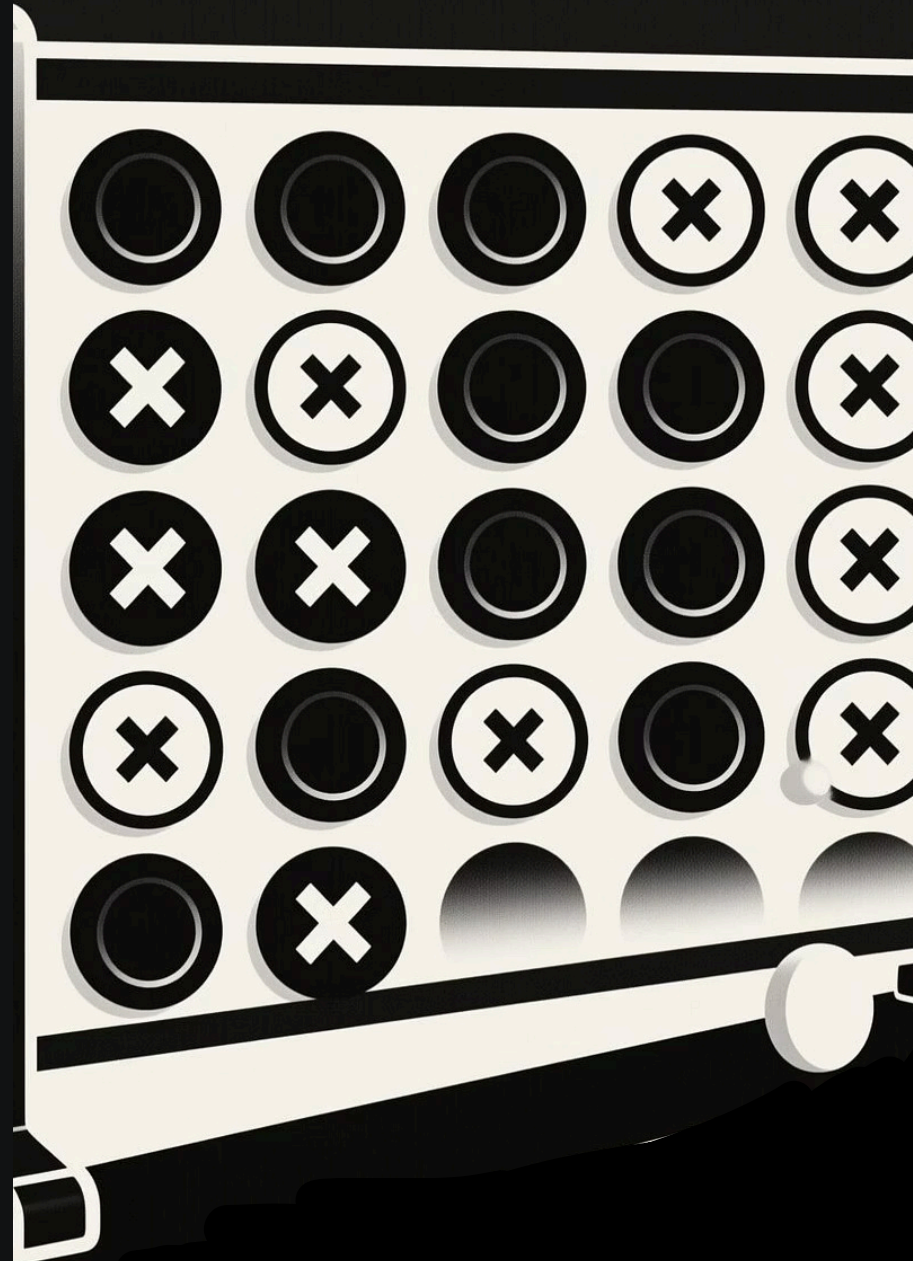
# Introduction: The Classic Reimagined

## A Timeless Strategy Game

Connect four has captivated players for decades with its elegant simplicity: drop coloured pieces, connect four in a row, and claim victory. The game demands strategic foresight and tactical execution.

## The Digital Journey

This presentation explores how we transformed this beloved board game into a fully functional Python application. We'll examine the architecture, implementation strategies, and the elegant solutions that bring Connect four to life on screen.



# Objective & Aim:



## Create Functional Game

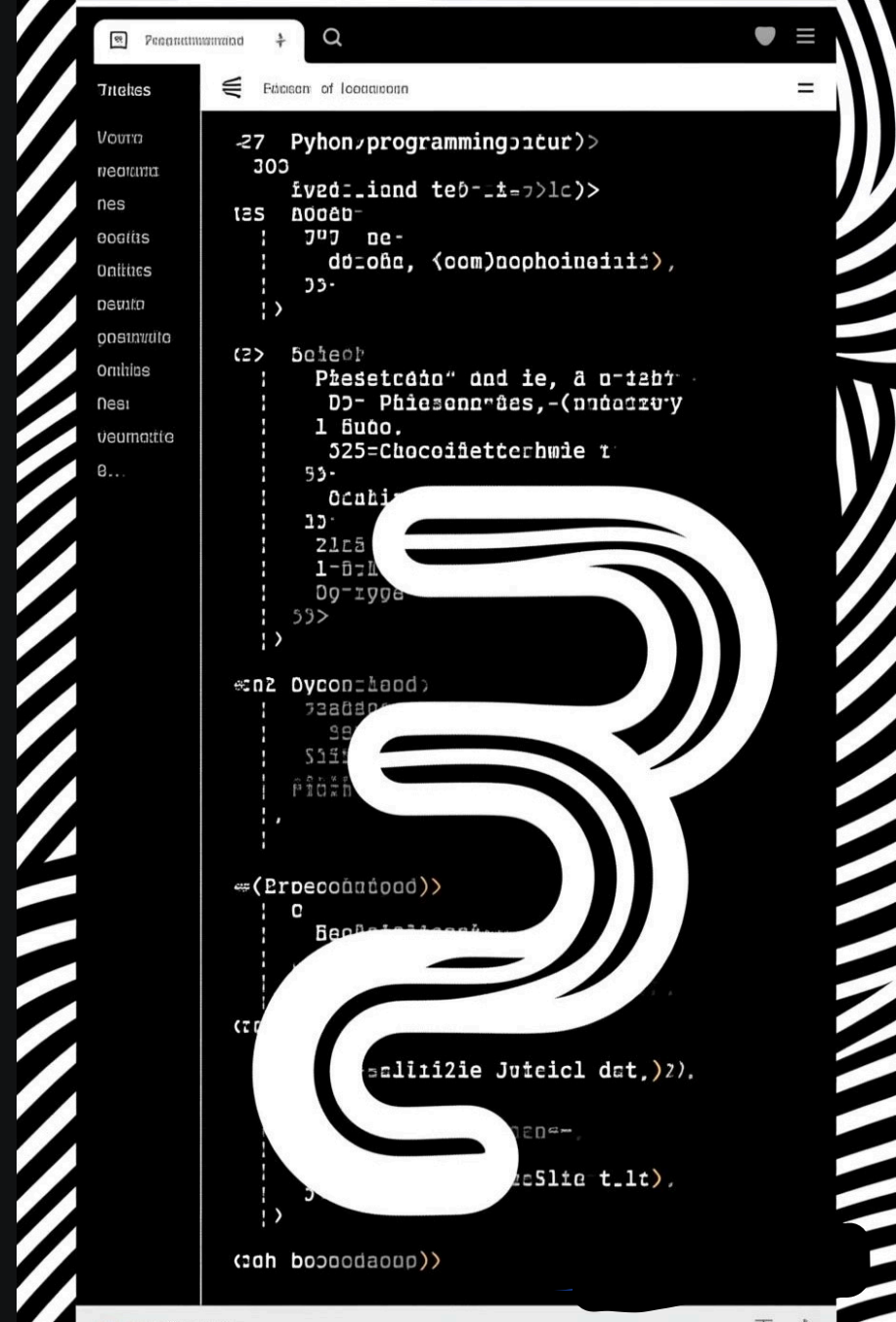
Develop a fully operational Connect four game with complete gameplay mechanics.



## Implement Game Logic

To **apply programming logic** for turn-based gameplay and win detection.

- To **enhance problem-solving and coding skills** through game development.





# This specific code uses only Python's built-in features

**Lists** → for creating and managing the game board.

**Loops (for, while)** → for gameplay logic.

**Input/Output** → `input()` and `print()` for user interaction.

**Functions** → to organize the code (e.g., `create_board`, `drop_piece`, `winning_move`).

# Working of the App

- The game starts with an **empty 6×7 board**.
- **Player 1 and Player 2** take turns choosing a column.
- The chosen piece **drops to the lowest empty spot** in that column.
- After each move, the program **checks for a win** or a **draw**.
- If a player connects **four pieces in a row**, they **win the game**.
- If the board is full and no one wins, it's a **draw**.
- The game then **ends and displays the result**.



# The Gameplay Loop:

**Initialise Board**  
Create empty game state

**Switch Turns**  
Continue playing

**Check Win**  
Detect victory

**Display Board**  
Render current state

**Get Player Input**  
Column selection

**Drop Piece**  
Gravity logic



The game continues this loop until a player aligns four pieces or the board reaches capacity.

# Output: Gameplay Demonstration

```
Welcome to Connect Four!
Player 1 = X | Player 2 = O
```

```
  0 1 2 3 4 5 6
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
-----
```

```
Player 1's turn (X)
Choose a column (0-6):
```

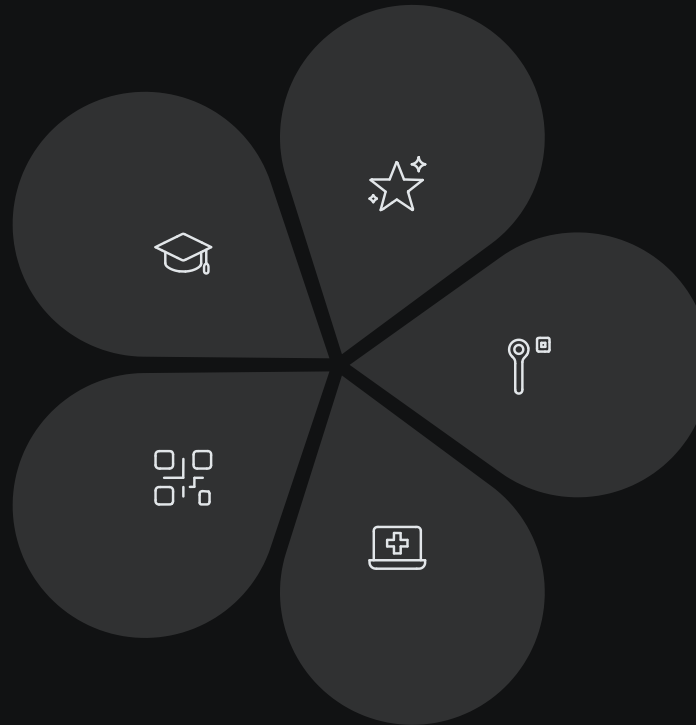
```
  0 1 2 3 4 5 6
| | | | | | |
| | | | | | |
| | | | | | |
| | | | |X| |
| | |X|X|O| |
| |X|X|O|O|O|
-----
```

```
Player 2's turn (O)
Choose a column (0-6): 6
```

# Key Benefits

**Rule-based logic** – follows the real Connect Four rules (4 in a row to win).

**Automatic win detection** – checks horizontal, vertical, and diagonal lines.



## **Interactive Entertainment**

Delivers genuine gameplay enjoyment with responsive controls and engaging visual experience.

**Console-based interface** – easy to play without extra installations or graphics.

## **Cross-Platform Access**

Runs seamlessly on Windows, macOS, and Linux with minimal dependencies.





# Future Improvements

- Introduce a **single-player mode** with an **AI opponent** using algorithms like *Minimax*.
- Include **sound effects and animations** to make gameplay more engaging.
- Add a **graphical interface (GUI)** using *Pygame* or *Tkinter* for better visuals.
- Implement a **restart or reset option** without restarting the program.



## Conclusion: Game On!

- The Connect Four game successfully demonstrates **Python programming concepts** in an interactive way.
- It shows the use of **loops, functions, lists, and conditions** to build a complete working application.
- The program provides a **fun and logical gaming experience** through a simple console interface.
- Overall, it is a **simple yet powerful example** of how Python can be used to create classic games.