Connect Four - Python Implementation

Author: Semyon Tsyrenov

Description:

This project is a Python-based implementation of the classic game **Connect Four**. It allows two players to take turns placing tokens in a 7-column, 6-row grid, with the goal of connecting four of their tokens in a row, column, or diagonally.

How to Play

- The game is played between **two players**.
- Players take turns placing their tokens (Player 1 uses `1`, Player 2 uses `2`).
- Players select a column (0-6) to drop their token.
- The first player to connect **four tokens vertically, horizontally, or diagonally** wins.
- If all 42 spaces are filled without a winner, the game ends in a **draw**.

How the Code Works

Main Functions

1. **`setup()`** - Initializes the game board as a 6x7 grid filled with zeros (empty slots).
2. **`present(b)`** - Displays the board in the console.
3. **`colheight(b, c)`** - Returns the current height of tokens in column `c`.
4. **`pt(b, p, c)`** - Places player `p`'s token in column `c` at the lowest available row.
5. **`vc(b, p, c)`** - Checks if player `p` has won **vertically** in column `c`.
6. **`hc(b, p, c)`** - Checks if player `p` has won **horizontally** after placing a token in column `c`.
7. **`d2(b, p)`** - Checks for a **diagonal win** (↘ pattern).
8. **`d4(b, p)`** - Checks for a **diagonal win** (↙ pattern).
9. **`playgame(b)`** - Runs the game loop:
   - Asks players to enter column choices.
   - Places tokens and checks for a winning condition.
   - Displays the updated board.
   - Ends the game if a player wins or if the board is full.

Features

✔ Two-player turn-based gameplay
✔ Real-time board updates
✔ Win detection for vertical, horizontal, and diagonal victories
✔ Input validation for correct column selection
✔ Game ends in a draw if no player wins within 42 turns

How to Run the Game

1. Install Python (if not already installed).
2. Save the script as `connect_four.py`.
3. Open a terminal or command prompt and navigate to the script's directory.

Run the game with:
bash
CopyEdit

```
python connect_four.py
```

4.
5. Follow on-screen prompts to play.

Possible Enhancements

- Implement an **AI opponent** for solo play.
- Add a **graphical user interface (GUI)** using `Tkinter` or `Pygame`.
- Improve **win detection efficiency** with a more optimized algorithm.

Why This Project Matters?

This project demonstrates proficiency in:

- **Python programming** (loops, conditionals, functions)
- **Data structures** (2D lists)

- **Algorithm design** (turn-based logic, win conditions)
- **Problem-solving skills** (validating user input, handling game states)

This code is a great example of implementing **game logic** and **handling user interaction** in a structured manner.