Cpts 440 Project 3, Alpha-Beta Pruning Documentation

Description: A Python implementations of Connect 4 using Pygame, featuring an AI opponent

Relevant File: connect_4_playing_ai.py

Required Packages: Pygame, Numpy

Rules:
- Players alternate turns dropping pieces
- The first player to connect 4 pieces horizontally, vertically, or diagonally wins
- Player 1 (the player that drops the first piece) uses red pieces
- Player 2 uses yellow pieces

Key Function Definitions:

- d**raw_board(board)**
  Renders the game board using Pygame
  Args: board (numpy.array): ROWSxCOLS game board array
  Note: ROWS and COLS are global variables in the relevant file

- **drop_piece(board, col, piece)**
  Attempts to drop a piece in the specified column
  Args:
  board (numpy.array): Game board
  col (int): Column to drop piece (0-6)
  piece (int): Player piece (1 or 2)
  Returns: bool: True if piece was placed, False if column full

- **winning_move(board, player_piece)**
  Checks if the specified player has won
  Args:
  board (numpy.array): Game board
  player_piece (int): Player to check for win (1 or 2)
  Returns: bool: True if player won

- **minimax(board, depth, alpha, beta, maximizingPlayer)**
  AI algorithm for choosing optimal move

Args:
board (numpy.array): Current game state
depth (int): Search depth remaining
alpha (float): Alpha value for pruning
beta (float): Beta value for pruning
maximizingPlayer (bool): True if AI's turn
Returns: tuple: (column, score) of best move

- **heuristic(board, piece)**
  Evaluates board state for given player
  Args:
  board (numpy.array): Game board
  piece (int): Player piece to evaluate for
  Returns: int: Score representing board state

- **play_game()**
  Main game loop handling player input and game flow
  No args/returns, runs until game completion

Technical Implementations: The player that moves first is decided by two global variables:

- HUMAN_PIECE
- AI_PIECE

In the case where (HUMAN_PIECE == 1) and (AI_PIECE == 2), the human player will get the first move.
In the case where (HUMAN_PIECE == 2) and (AI_PIECE == 1), the AI player will get the first move.
The program is designed to handle only the two specified cases. Behavior is not guaranteed for any other scenarios

The AI's move speed and move quality is heavily dependent on the depth parameter of the minimax() function. The relevant minimax() call can be found in the play_game() function. A smaller depth value results in faster move speed but reduced move quality. A larger depth value results in slower move speed but improved move quality.

Controls: Upon running the relevant file, a window that displays the game board should appear on your screen. You can place a piece by hovering your cursor over the column you wish to place a piece in and clicking. Regardless of which row your cursor is in, the piece should appear at the lowest available column.