Object Oriented Programming Course (CS213)
Under Supervision Eng. Rana Abdelkader

Group: S21

Assignment 3

Name	ID	Tasks
Ghassan Tarek Elgendy	20220239	Four-in-a-row Game
Rawan Ehab Ammar	20220133	Pyramic Tic-Tac-Toe
Jana Mohamed Ramadan	20220470	5 x 5 Tic Tac Toe

Pyramic Tic-Tac-Toe:

1. **Constructor:**
   - The constructor initializes an empty 3*5 array to make the basic dimensions of the pyramid.
   - All elements in the array are initialized to the ascii of 0 which is a **blank space.**
2. **Display_board():**
   - Function that prints the content of the array in the relevant parts of the pyramid.
   - The pyramic shape is made using the equation 2*i+1.
   - If the content of the elemnt is a blank space the index is printed, else the content of the cell is printed.
3. **Update_board():**
   - Takes the user input and updates the elements of the array if there are no other elements already in that cell and it is a valid cell.
   - Increments the number of moves counter to keep track.

4. **Is_winner():**
   - Function to check after every move if there is a winner or not.
   - It iterates over each cell if there are three consecutive elements of the same label In any direction , then there is a winner.
5. **Is_draw():**
   - Function to check if the game is a draw
   - If 9 moves have been made and there is no winner yet, then it’s a draw.
6. **Game_is_over():**
   - Returns true when the game is finished

7. **initPyramicTicTacToe():**
   - Used to initialize the pyramic Tic-Tac-Toe game in source file.

Four-in-a-row:

1. **Constructor ConnectFourBoard:**
    - Initializes a Connect Four board with dimensions 7x7.
    - Populates the board with zeros and fills the bottom row with '1's.
2. **update_board:**
    - Checks if the move is valid and places the mark on the board if valid.
    - Returns true if the move is successful, false otherwise.
3. **display_board:**
    - Displays the current state of the Connect Four board, including cell coordinates.
4. **is_winner:**
    - Checks for a winning condition by examining rows, columns, and diagonals.
    - Returns true if there's a winner, false otherwise.
5. **is_draw:**
    - Determines if the game has ended in a draw.
    - Returns true if the maximum moves (42) are reached without a winner.
6. **game_is_over:**
    - Checks if the game is over (all moves are completed or there's a winner).
    - Returns true if the game is finished.
7. **get_board_value and set_board_value:**
    - Retrieve and set values at specific positions on the board, respectively.

GameManager Class:

- **Constructor GameManager(Board* bPtr, Player* playerPtr[2]):**
    - Initializes the game manager with a board and two players.
- **run:**
    - Controls the game flow:
        - Displays the initial board state.
        - Loops through player moves, checks for a winner or draw, and ends the game accordingly.

AiPlayer Class:

- **Constructor AiPlayer(char symbol):**
    - Initializes an AI player with a given symbol.
- **get_move :**
    - Implements AI logic to make moves based on the current board state.
    - Attempts to win if possible, otherwise blocks the opponent or makes a random move.
- **get_board :**
    - Retrieves the board pointer for AI decision-making.

5 x 5 Tic Tac Toe:

1. Board5x5 Class
   - **getPlayers:** Assigns players to the game board for a 5x5 game to use in the win cases.
   - **Board5x5:** Constructor for the 5x5 game board, initializing empty 5x5 grid.
   - **update_board:** Updates the board with a player's move if it's a valid move.
   - **display_board:** Displays the current state of the 5x5 game board.
   - **is_winner:** Checks if there's a winner based on rows, columns, diagonals, and reverse diagonals and count every three in a row.
   - **game_is_over:** Checks if the game is finished, considering the maximum allowed moves, 24 moves
2. RandomPlayer5x5 Class
   - **RandomPlayer5x5:** Constructor for a random player in the 5x5 game.
   - **get_move:** Generates a random valid move for the player on the 5x5 board.
3. GameManager5x5 Class
   - **GameManager5x5:** Constructor for the 5x5 game manager, managing the flow of the game.
   - **run**: Controls the game execution, allowing players to make moves and updating the display until the game is over.