Cairo university

Faculty of computers and information

CS213
Object-oriented-programming
Assignment#3
2023

Ahmed Hossam

Mohamed Shebl

Mostafa Tarek

Dr Mohammed El-Ramly

m.elramly@fci-edu.eg

Program description :

This program is a menu-driven console application for different board games. It allows users to select a game from a menu and provides options to play against another player or against a computer AI for certain games. The selected game is instantiated along with player objects, and the GameManager class manages the flow of the game.

Based on the user's input (c), the program enters different branches (if conditions) to handle each game choice (1, 2, 3, or 4).

For each game choice:

Two player objects (players[2]) are created. The first player is always initialized with the symbol 'x' and the second with the symbol 'o'.

The program prompts the user to choose between playing against another player (2) or against a computer (1).

Player 2 is instantiated accordingly, either as a human player or a computer player.

A specific game board object (pyramid_XO, Four_in_a_row_Board, five_X_O_Board, X_O_Board) is instantiated based on the user's choice.

Game Initialization and Execution

Once the game board and player objects are set up according to the user's selections, a GameManager object (games) is created.

The GameManager object is initialized with the game board and player objects.

The run() method of the GameManager object is called to start the game.

The Idea Behind Each Function in the inherited ” Board “ class for Pyramid Tic Tac Toe Game :

Constructor (pyramid_XO()):

Initializes the n_rows, n_cols, and n_moves member variables.

Dynamically allocates memory for the 2D array board to represent the game board.

Initializes each element of the board to a space character (' ').

update_board(int x, int y, char mark):

Updates the game board with the player's move at position (x, y) with the specified mark.

Checks various conditions to ensure the move is valid, such as the position being within bounds and the cell being empty.

Returns true if the move is valid and updates the board; otherwise, returns false.

display_board():

Displays the current state of the game board in a visually formatted way.

Uses a nested loop to iterate through each row and column, printing the corresponding board elements.

Special formatting is applied for certain positions to create the pyramid structure.

is_winner():

Checks for winning conditions in the current state of the game board.

Checks for three in a row horizontally, vertically, and diagonally.

is_draw():

Checks if the game is a draw by verifying that all nine moves have been made and there is no win

The Idea Behind Each Function in the inherited ” Board “ class for Four in a row Tic Tac Toe Game :

Constructor Four_in_a_row_Board::Four_in_a_row_Board():

Initializes the game board for a "Four in a Row" game.

Allocates memory for the game board, sets the number of rows and columns, and initializes each cell of the board to an empty state (represented by the value 0).

bool Four_in_a_row_Board::update_board(int x, int y, char mark):

Updates the game board with a player's move.

Checks if the move is valid by verifying if the selected column (y) is within the board boundaries and if the selected column has an empty space. If the move is valid, the function updates the board with the player's mark in the lowest available row in the chosen column.

void Four_in_a_row_Board::display_board()

Displays the current state of the game board.

Prints out the game board along with the coordinates and marks (player symbols) present in each cell. Visualizes the board structure and the current game state for players.

bool Four_in_a_row_Board::is_winner()

Checks if there's a winner based on the game rules.

Checks the board horizontally, vertically, and diagonally to identify if there's any sequence of four consecutive marks (either 'X' or 'O') indicating a win. Returns true if there's a winner, otherwise false.

bool Four_in_a_row_Board::is_draw()

Determines if the game is a draw.

Checks if the game has reached its maximum possible moves (42 moves in this case) without a winner. If there's no winner and the maximum moves are reached, it considers the game a draw and returns true.

bool Four_in_a_row_Board::game_is_over()

Checks if the game is over.

Determines if the game is over by checking if the number of moves made is equal to or greater than the maximum possible moves (42 in this case). Returns true if the game is over, otherwise false.

Each function serves a specific role in managing the game board, updating its state, checking for a winner or draw, and determining the game's status. Together, they facilitate the logic and functionality of the "Four in a Row" game.

The Idea Behind Each Function in the inherited ” Board “ class for 5x5 Tic Tac Toe Game :

Constructor five_X_O_Board::five_X_O_Board():

Initializes the board for a "Five X O" game.

Sets up the board with dimensions 5x5 and initializes each cell to contain empty spaces initially, denoted by the character ' '.

bool five_X_O_Board::is_winner():

Checks if there is a winner according to the game rules.

Evaluates the board horizontally, vertically, and diagonally to determine if either 'X' or 'O' has three consecutive marks (indicating a win). The function counts the number of winning patterns for both 'X' and 'O' and determines the winner based on the count.

bool five_X_O_Board::update_board(int x, int y, char mark):

Updates the board with a player's move.

Checks if the provided coordinates are within the board boundaries and if the cell is empty. If valid, places the player's mark ('X' or 'O') in the specified location and increments the number of moves made.

bool five_X_O_Board::game_is_over():

Determines if the game is over.

Checks if the number of moves made reaches 24 (indicating the end of the game in this implementation).

bool five_X_O_Board::is_draw():

Checks if the game ends in a draw.

Verifies if the total number of moves reaches 24 without declaring a winner, signifying a draw.

void five_X_O_Board::display_board():

Displays the current state of the game board.

Prints out the game board along with the coordinates and player marks ('X' or 'O') present in each cell. Provides a visual representation of the board and its contents.

Each function is dedicated to managing the game board, validating moves, checking for a winner or a draw, and displaying the board's state. Together, they contribute to the logic and functionality of the "Five X O" game.

Work break down table

Mostafa Tarek	Ahmed Hossam	Mohamed Shebl
5 x 5 Tic Tac Toe	Pyramic Tic-Tac-Toe	Four-in-a-row
We all worked on the menu together		

Github screenshot