1D Project

10.014 – Computational Thinking For Design

Cohort 09 Team 9 – Fox and Hounds

Members: Yee Li Ren, Harel (1005133), Sim Shang Hong (1005500), Nandini Prabakaran (1005390), Soh Zheng Rong (1005199)

Description

Summary:

This game is a modified version of Fox and Hounds, which is played on an 8x7 checker board. There are 2 players to the game, one controlling the Fox while the other controls 4 hounds. The objective of the game is for the hounds to catch the fox while the fox tries to break through the line of hounds to reach the other end of the board.
At the start of the game, four hounds are lined up at one end of the board while the fox and at the other end of the board.

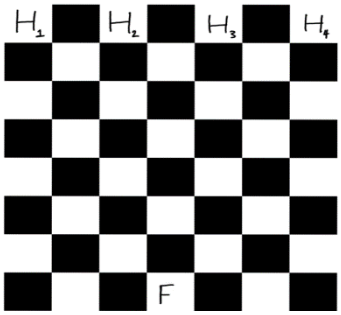

Figure 1: Initial setup of the board

Movement:

- Fox:
  - The fox is able to move diagonally on the black squares, one square each turn.
  - The fox is able to move both diagonally forwards or backwards.
  - The fox is able to consume the hounds.
- Hounds:
  - The hounds are able to move diagonally on the black squares, one square each turn.
  - However, the hounds are only able to move diagonally forwards.
  - Only one hound can be chosen by the player to be moved at one time.

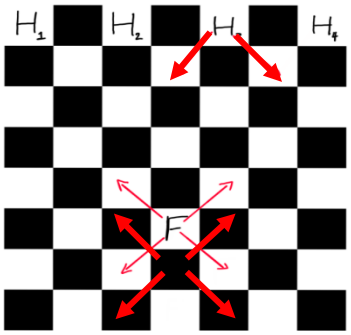

Figure 2: Moving the Fox and the Hounds

Winning Conditions:

- Fox:
  - The fox wins when it reaches the other end of the board (*Figure 3).*
- Hounds:
  - The hounds win if it captures the fox
  - The hounds win if the fox has no more valid moves:
    - Case 1 – The fox is cornered on all four sides by hounds and can no longer move (*Figure 4a*).
    - Case 2 – The fox is cornered on the side of the board and can no longer move (*Figure 4b*).

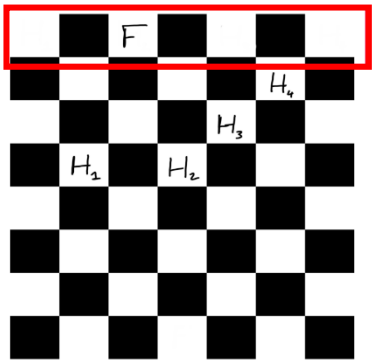

Figure 3: Fox Winning Condition

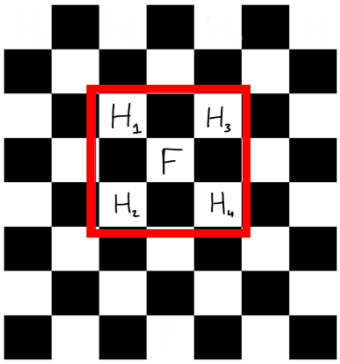

Figure 4(a): Hounds Winning Condition- Case 1

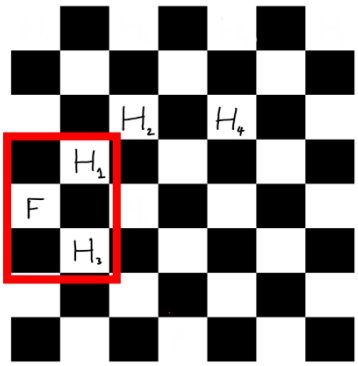

Figure 4(b): Hounds Winning Condition- Case 2

How to Play

In the beginning, the Fox (Orange tile) and the Hounds (Hound 1: Yellow Tile, Hound 2: Green Tile, Hound 3: Blue Tile, Hound 4: Purple Tile) have already been placed in their starting positions. When the code is run, the fox will begin his turn first. The rows and columns are indexed on the board and the player will need to reference the rules to know which tile is traversable for the fox or the hound. The player(Fox) will then be prompted to choose the row and the column to move his tile to. After which, the turn will switch to the player(Hound). The player(Hound) will then need to choose the minion to move and to which column forwards. The game ends when a winning condition(stated above) is met and the board can then be reset back to its initial position

Documentation

It is required for this game to use the matplot library.

display_board(board): This function takes in the board and returns a display board as an image.

win_logic(board): This function takes in the state of the board at the start of each round and returns a value of 0, 1 or 2 depending on who has won. 0 will refer to a tie, 1 will refer to a win by Hounds and 2 will refer to a win by the Fox.

user_input(player, board): This function takes in 2 parameters, the current player that is performing his turn, and asks for the player to choose the hound they would like to control (if the player is controlling hounds) and choose the position that he would like to move to. The function also takes in the board to reference the possible moves that can be made by each palyer. The function will then return coordinates and the hound minion(if it is the hound minion's turn). This function also checks if the move input by the player is valid. This will be based on the condition that:

1. There is no other piece on the grid that the player wants to move into.
2. (For Hounds) The move is in the forward direction.

update_board(player,i,j,board, hound_minion): This function updates the board to the next iteration after the move has been made by the players. It takes in the player currently performing his turn, the coordinates chosen by the user, the board and the hound minion(if it is the hound minion's turn). The function will then return a **board** for the next iteration.

switch_player(player): This function takes in one parameter **player** and switches the player to the next to allow the next player to make his turn.

main(): This is the function that runs the game. The pseudocode is as follows:

1. Import necessary libraries
2. While set to "play" in order to reset the game afterwards
3. Initialize a board with the initial fox and hounds on the board.
4. While the winning condition is not met:
    a. Display the board
    b. Request for an input from the player
    c. Update the board according to the inputs from the player
    d. Switch the player for the next turn
    e. Check for winning conditions
5. When the winning condition is met, display a message to show who has won.
6. Ask the player on whether or not to reset the board and the game.