Department of Computer Science and Engineering

Course Code: CSE-3636
Course Title: Artificial Intelligence Lab
Project Title: AI based Chess Game using Python

Project Documentation

Week 1: Drawing the Chess Board

Submitted by

Lamisa Mashiat (C201249)

Sohana Tasneem (C201266)

Sadia Haque Chowdhury (C201270)

Submitted to

Subrina Akter

Assistant Professor

Dept. of CSE

18th March, 2023

The IDE in which we are working to develop our project is PyCharm Community Edition 2021.2.3. And we are using Python 3.9.7 language as an interpreter. The first stage of our project, the development of the chess board, took place over the period of the last week.

We have created a Python Package called Chess. The Package contains following list of files:

1. Images folder
2. ChessEngine.py
3. ChessMain.py

Images Folder

This folder contains 12 images of the chess set pieces. Images are named following a convention where first letter represents their color (b for black, w for white) & the last letter denotes the type of the pieces (K - king, Q - queen, R - rook, B - bishop, N - Knight, P - pawn)

ChessEngine.py

This Python file contains a class called GameState(). This class is responsible for storing all the information about the current state of a chess game. It'll also be responsible for determining the valid moves at the current state. It will also keep track of the move log.
We have created a list named 'board' to portray the design of a chess board.

- Board is an 8X8 2d list, each element of the list has 2 characters.
- First character represents the color of the piece, 'b' OR 'w'
- Second character represents the type of the piece,'K','Q','R','B','N','P'
- "--" represents an empty space

ChessMain.py

This is a Python script for a chess game GUI built with the Pygame library. The game logic is implemented separately in the "ChessEngine" module which is imported at the beginning of the script. This is our Main Driver file. It will be responsible for handling user input & displaying the current GameState object.

The script defines the following variables:

1. WIDTH: The width of the game window in pixels.
2. HEIGHT: The height of the game window in pixels.
3. DIMENSION: The number of rows and columns on the chess board (the board is square).
4. SQ_SIZE: The size of each square on the chess board (calculated by dividing HEIGHT by DIMENSION).
5. MAX_FPS: The maximum frame rate for animations.
6. IMAGES: A dictionary that maps piece names (strings) to images of those pieces (loaded from image files).

The script also defines the following methods:

- loadImages(): Loads the images for all chess pieces into the IMAGES dictionary.
- main(): The main function that initializes Pygame, creates the game window, loads the images, initializes the game state (using the GameState class from the ChessEngine module), and runs the game loop. The game loop handles user input and updates the graphics by calling drawGameState().
- drawGameState(screen, gs): Draws the current game state on the screen by calling drawBoard() and drawPieces().
- drawBoard(screen): Draws the squares of the chess board on the screen by iterating through each row and column of the board, determining the color of the square, and drawing a rectangle using Pygame's draw.rect() method.
- drawPieces(screen, board): Draws the pieces on the chess board by iterating through each row and column of the board, determining which piece is on each square, and drawing the corresponding image from the IMAGES dictionary using Pygame's blit() method.

The script also includes an if statement at the end that checks if the script is being run as the main program (rather than being imported as a module). If it is being run as the main program, the main() function is called to start the game.

The work we have done so far produces the following output: