CS246 Chess Documentation and Guide

**Overview of Chess**

**Program Architecture**

The program overall is comprised of a View, Controller, Game, Move, Player and Piece classes with minimizing coupling and cohesion as a first priority and the use of design patterns as a second priority.

Here is a quick rundown of the classes and its role in the program:

View: Controls all IO with the player, has a general input method that takes commands from users and multiple output methods to output to the user.

Controller: Responsible for the game’s runtime. This class contains the programs main function that contains the logic which dictates the flow of the game. This class is the interface that calls the game’s setup, the start/end of the game and also controls the games turns and match making.

Game: Holds the games data, such as the board and piece locations and has the majority of the logic for the games rules. This class interprets move commands checks the validity of the command (checks if it is a legal move in chess, this includes things like checks and piece moves) and then performs the move which alters the board.

Move: An object which serves as data block that is passed between the player and the game. Has the data for a player’s move in chess.

Player: Comprised of a Human and AI subclass. The Human class allows users to play the game and the AI subclass holds logic for the computer to play the game. In addition the AI subclass has 4 unique levels of behavior.

Piece: Contains the Pawn, Bishop, Queen, King, Rook and Knight subclasses that define the behavior of each type of piece. Each Piece contains a method to check if a move is within its move set and capture set and data regarding the type of piece it is, the colour of the piece and whether or not the piece has moved or not.

**Design Patterns**

The primarily implements two design patterns; the MVC and Template pattern.

**MVC Design Pattern:** The program uses an MVC design pattern, which is evident in the class designs.

In an MVC design pattern there are three distinct layers: The Model, the View and the Controller. In this particular program each layer of the MVC design pattern is comprised of the following classes.

Model: the Player, Piece classes (and also the board field in Game) serve as the model in the program. These are classes that are manipulated by game and controller as have very little access and control over the rest of the program.

View: the View class is distinctly responsible for this layer, with all views (Input/Output) being handled by this class.

Controller: Game and Controller classes manipulate the game and directly control the legality of game moves, turn taking and match making. These two classes also do all of the calls to view in order to get Input/Output.

**Template Pattern**: the Piece class is implemented with a Template Design Pattern. The Piece class by itself is an abstract superclass and the Pawn, Bishop, Rook, Queen, King, Knight subclasses override some aspects of this superclass.

**Coupling & Cohesion**

**Flexibility of Program**

**Game Rules**

**Program Runtime at a Glance**

**Implementation Differences**

**Known Bugs & Issues**

**Questions**