Project: Chess
Group Members: j29yoo, sy2baek, h23yoo

Overview

1. Design of the game
    a. Explore features of the game and the specifications
    b. Consider design the pattern
    c. Possible bonus features (if decide to add)
    d. UML
2. Breakdown of time management and each group members' roles
3. Project development
4. Testing
5. Debugging
6. Final markup

Plan of Attack

Complete by July 15 2016 (Due Date 1):

- UML for the program
- Plan of Attack
- Work load separation

The first week will be focused on the design of the UML, and then splitting the workload depending on the difficulty and significance of completion order. UML is expected to be finished by Thursday alongside with the plan of attack document.

Complete by July 25 2016 (Due Date 2):

- Development: .h files and corresponding .cc files
- Merging code
- Testing
- Debugging
- Update UML
- Include bonus features if time is allowed
- Final design document

All group members would be involved with the development:

July 15 – 17, 2016

- **h23yoo**: Create a Board class and implement methods for notify, update, etc. Also Player class for human.
- **sy2baek**: Create class for all pieces, making move methods and all other simple methods.
- **j29yoo**: Create Controller Class; implement methods for initialization, setup, printscore, etc.

July 18 - 19, 2016

- **h23yoo + sy2baek**: Implement A.I level 1 and 2
- **j29yoo:** Create classes for View: TextDisplay and GraphicalDisplay

July 20 – 22, 2016

- **h23yoo + sy2baek**: Implement A.I level 3 and possible 4 if time is available. Finish rule features if not finished (en pasant, pawn promotion, castling, checkmate)
- **j29yoo**: Add scoring system and bonus features if time available

July 23 – 24, 2016

- **All**: testing, debugging, finalizing design document with the final UML

Q & A

Question 1: Chess programs usually come with a book of standard opening move sequences, which list accepted opening moves and responses to opponents' moves, for the _rst dozen or so moves of the game. Although you are not required to support this, discuss how you would implement a book of standard openings if required.

We could implement by using a tree, where each node is a state of board. Then, given the current state of the board, appropriate moves would be thoroughly searched and these would be stored as a string in the corresponding node of the tree. Therefore, when a move is made, the active move will be found by looping through the tree until the corresponding move is found. Then, the move will be made if it is found; otherwise, the move will be made without the standard openings.

Question 2: How would you implement a feature that would allow a player to undo his/her last move? What about an unlimited number of undos?

We would store all commands in a Vector of strings from the beginning of the game, so including the setup command if existed. Then, if an undo command has been performed, we would clear the current Board and create a new Board by performing the commands stored in the Vector. Thus, unlimited undo command will continuously perform this process. If a player is restricted with only one undo, there will be a boolean value to track if an undo command has already been performed.

Question 3: Variations on chess abound. For example, four-handed chess is a variant that is played by four players (search for it!). Outline the changes that would be necessary to make your program into a four-handed chess game.

1. The board must be changed/updated to fit the requirements of the four-handed chess. So, there would be additional 24 squares (8 x 3) to each side of the board where the staring pieces would be placed.
2. As a result, the movement of the chess pieces would be updated, where the boundary has been changed.
3. Additional colours would be necessary for four players. Therefore, text display system must be changed as uppercase and lowercase would not be enough to distinguish between the four players. A possible solution would be combination of two-strings where one player would have all uppercase letters, all lowercase letters, first lowercase and second uppercase, or first uppercase and second lowercase.
4. Pawn promotion must be modified as possibilities of reaching the end of the board has been increased to three from one.

5. Finally, the win condition is changed depending on the play-style of the game, whether by team battle (2v2) or independent battle (1v1v1v1). For team battle, if one team's both kings is checkmated. For independent battle, the last standing Player's king is the winner, so other three kings are captured or checkmated.