Grayson Rechsteiner

Computer Science Culminating Journal

Nov 22: 1.5 hours

- Found & cropped all images
- Got the board displayed on the window with a background
- Made the function for determining which square (in relation to 2d list) is being pressed based on mouse input

Nov 23: 2 hours

- Got the individual pieces loaded onto the screen in their starting positions
- Made the piece class (this took a lot of thinking into how the best way to do this was, but I think I ended up picking the best way to handle it (including the board 2d list))
- https://stackoverflow.com/questions/3289601/referring-to-the-null-object-in-python
    - Needed python equivalent of null to store in 2d list slot when a piece (created in a class) wasn't there (empty spot on board)

Nov 24: 1.5 hours

- When I click on one spot on the board, followed by another spot on the board, the pieces will switch locations in the 2d list and be displayed in their new spots
- Had to work through a couple bugs when doing this

Nov 25: 1.5 hours

- Got new transparent images in (this took a long time to crop all the pieces)
- Made a function to calculate current score (based on piece points)
- Fairly easy to do this, just time consuming

Nov 26: 2 hours

- When a piece of one colour moved to a spot with a piece of a different colour, it would capture the piece and take it off the board
- Got the captured pieces displayed on the side of the screen
- Made it so you can only move a piece to a empty square or a square with a different coloured piece (the other team)

Overall feeling really good this week, I've been able to get a lot done and it's coming together really well. Although the most challenging parts are ahead: only allowing valid moves from pieces, handling check, actually running the game, etc

Nov 27: 1.5 hours

- Made it so pieces could only move to valid spaces (bishop has to go on diagonal)
- Made it so potential moves based on piece selected are highlighted with green circle
- Still have to sort out pieces not being allowed to pass pieces if they are blocked
- This stuff was really straightforward, just had to draw a bit out on paper, mostly just time consuming

Dec 6: 2 hours

- Made it so pieces couldn't travel through other pieces (in valid moves)
- Made it so that pawns can take pieces diagonally when in correct position
- This was actually a lot easier than I thought it would be

Dec 7: 2 hours

- Started working on when a pawn reaches the other side of the board. Made a screen where they click the piece they want, and then the pawn changes to that piece
- This has taken a while, and I'm still working on it, more complicated than I initially thought it would be
- Need to create the new image with the possible pieces to choose from (right now it only has rook, bishop, queen, and not the knight), will do this tomorrow

Dec 8: 1.5 hours

- Made it so when pawns get to end proper images are displayed for promotion options
- Touched up an error when pawns can "take" to a spot off the board
- Touched up some minor things (possible move circles still displaying after pawn gets to end and before player has chosen piece to promote to)

Dec 9: 2 hours

- Made a function to check if either king is in check
- Made it so that if a player is in check, they are only allowed to move out of check
- When checkmate is achieved, prints who wins (for now just prints who wins, later it will bring you to a new screen or something like that)
- These things were easier than I thought it would be, although it did take a lot of thinking and experimenting. It felt great when I finally got it working

Dec 10: 1.5 hours

- Touched up some stuff from previous day (made a function to avoid duplication and made another part of my code simpler)
- Made it so that a player cannot move into a spot where they will be in check
- All of check is now done. Easier than I thought it would be.
- When a player is in check, a message is displayed on the screen
- When a player has won the game, a message is displayed on the screen

Dec 20: 2 hours & 45 mins

- Incorporated turns (each player can make 1 move before turn getting switched to other player)
- Displayed whoevers turn it is on the screen
- Converted entire program to game class so game could be played again when someone wins (this took a very very long time, harder than I initially thought it would be)

Dec 21: 1.5 hours

- Converted stuff from game class into board class.
- Lots of error messages, but eventually got it working
- Made some optimizations (took out unnecessary code/simplified it)

Dec 22: 45 mins

- Finished converting things from game class into board class. Fairly easy as I already had a hand of how to do it from the day before.

Dec 23: 2 hours

- Got castling working (including all special cases/rules of castling). This took longer than expected because of some of the special rules. I had to change how I approached it.
- Got en passant working. This was a lot easier than I thought it would be.

Dec 24: 2 hours

- Optimized/simplified stuff and removed unnecessary code, made variable and functions names better
- Created new functions to remove repeated code
- Implemented the code for stalemate. Fairly easy as I already had the functions needed to do this.

Dec 26: 2.5 hours

- Got a really basic starting menu
- Finished all cleanup/organization/simplification
- Needed to know how to add each individual item of a list to an already existing list, without iterating through each item
    - https://www.analyticsvidhya.com/blog/2021/06/15-functions-you-should-know-to-master-lists-in-python/
- Finally ready to start working on AI
- Feels really good to have all the basic stuff done and organized very well

Dec 29: 2 hours

- Got a really simple AI working, just chooses the move that will result in it having the highest score on the board. If all moves result in the same score, the AI moves a piece that hasn't been moved yet
- Made a home button that returns you to the menu (when within a game)

Dec 30: 1.5 hours

- Worked on the recursion needed for AI, got it working although it is fairly slow and it doesn't take into account pawn promotion
- Actually a lot easier than I thought it would be, I thought it would be a lot more difficult. At the same time, it's a pretty simple AI

Dec 31: 45 mins

- Made it so that you could deselect a piece if you clicked on it
- Inserted a sound that is played when a piece is moved

Jan 2: 3 hours

- Made it so sound effects could be toggled on and off by clicking a image on screen, mute screen would be displayed when sound is off
- Feeling really good with it as it looks really sharp with all the extra features
- Made it so the AI favoured attacking/advancing on the board
- Did some work on optimizing the AI so that it can check move possibilities quicker. So that I'll be able to make the AI do deeper checks
- https://en.wikipedia.org/wiki/Alpha%E2%80%93beta_pruning
- Made easy, medium, and hard modes for playing the AI. Based on how many moves the computer looks into the future. Although when playing hard mode it takes the computer a really long time right now (working on this), it can take up to a minute for one move.
- Made it so that pawn promotion works with the AI

Jan 5: 2 hours

- Made a hint button, follows same logic as AI with 2 depth, highlights piece that should be moved and where it should be moved to
- Went through entire program and added necessary comments, better organized code (spaces, comments, etc)
- Found a problem with the AI in regards to checkmate, will look at it tomorrow

Jan 6: 4 hours

- Worked on menu and instructions screen
- Made buttons appear as 3d
- Really happy with how some of the graphics are looking right now
- Made it so when game is won/tied, you have to click “Done” to continue
- Made a function for displaying when game is won/tied (got rid of repeated code)
- https://www.pygame.org/docs/ref/font.html#pygame.font.SysFont (for italics)

Jan 7: 2 hours

- Made selected piece displayed (box around square)
- Changed how hints are displayed to mirror how selected pieces are displayed
- Changed “DONE” to “CONTINUE” on win/tie screen
- Worked on instructions/how to play screen

Jan 9: 1.5 hours

- Did some optimizations that really helped the AI, now hard mode with 3 depth works in under 10 seconds (and medium down to under a second in most cases)
- Really, really happy with this (hard difficulty working)

Jan 16: 3 hours

- Did some minor changes to how possible moves, selected piece, and hints are displayed
- https://stackoverflow.com/questions/23852917/antialiasing-shapes-in-pygame (display outside of circles cleaner)
- Worked on speeding up the AI by sorting possible moves (helps alpha beta) and by optimizing other aspects of code
- https://docs.python.org/3/howto/sorting.html
- https://www.programiz.com/python-programming/shallow-deep-copy
- If certain moves yielded the same best output score, AI picks a random move from those moves
- https://stackoverflow.com/questions/4230000/creating-a-2d-matrix-in-python

Jan 17: 2.5 hours

- Handled other stalemate conditions (insufficient material)
- Went through entire program and better commented things, removed unneeded code
- Added two lines to instructions page
- Slight alterations to displays

Jan 18: 45 mins

- Refined insufficient material stalemate code
- Final look through of program, tested out various scenarios, played a couple games myself