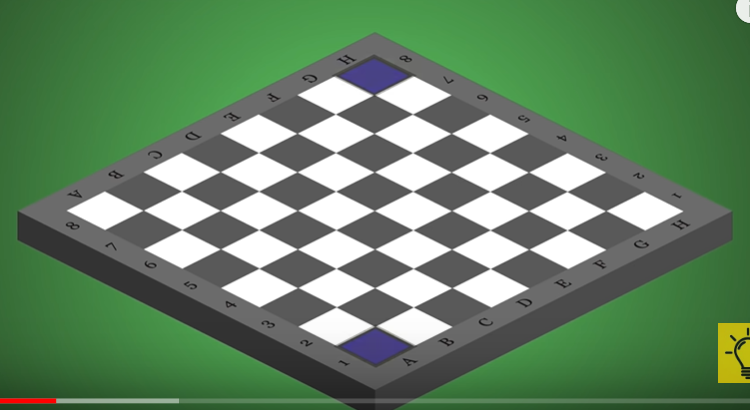
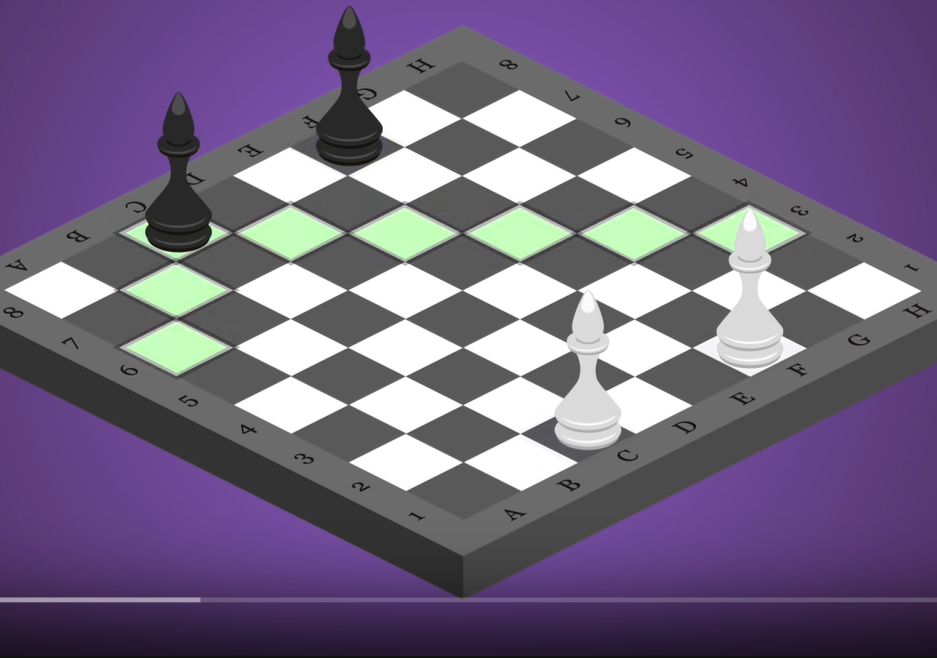
1. Themes:
   1. Polymorphism and Dynamic Binding
   2. Member access control
      1. Public
      2. Private
      3. Protected classes
   3. Pure virtual function/abstract classes
      1. NOTE the use of virtual functions to ensure that pieces that inherit are destroyed properly and accordingly. MAKE SURE that style of programming is appropriate and accepted. No choice but it appears that you definitely have to use pointers and destructors. BUT on the upside, usage is definitely limited.
   4. ADT/Templates (TBC)
2. Thoughts:
   1. Implement board
   2. Implement moves
   3. Implement positions
   4. Anything else?
   5. Note that strategies are NOT required EI implementing castling will give you additional credit
   6. Virtual function implementation is necessary to get a decent mark on it. Check out implementation as well.
3. Guide to chess: <https://www.youtube.com/watch?v=NAIQyoPcjNM&t=148s>
   1. Chess board = 64 square spaces
   2. Files = columns going up and down A-H
   3. Ranks = Horizontal rows from 1-8
   4. Position = Letter of file and number of rank
4. Chess pieces
   1. Each player should have dark square in their lower left corner
   2. 
   3. Each player has 16 chess pieces in total:
      1. 1 king
      2. 1 queen
      3. 2 rooks
      4. 2 bishops
      5. 2 knights
      6. 8 pawns
      7. Remember that the goal of the game is to checkmate the other king. Checkmate happens when the king is in a position that he CAN’T escape from capture
5. Rook
   1. Positions:
      1. Placed in corner of Board
      2. A1/H1 on one corner
      3. A8/H8 on the other corner
   2. Movement:
      1. Can move horizontally/vertically for any number of squares.
      2. Can’t move across each of the opponent pieces
      3. BUT remember that rooks CAN’T jump over pieces.
   3. Capture
      1. Move to where the opponent is seated and the piece is gone
6. Horse/Knight
   1. Positions
      1. Placed on squares B1/G1 for 1 player
      2. and B8/G8 for the other player
   2. Movement
      1. Knights CAN jump over other pieces. ONLY piece that can do so
      2. L shaped
         1. 2 squares horizontally Then 1 square vertically
         2. OR 1 square horizontally then 2 square vertically
   3. Capture
      1. Can only capture WHEN they LAND on the piece’s square
7. Bishops
   1. Position
      1. C1/F1
      2. OR C8/F8
   2. Capture
      1. Can only capture by stepping on the piece’s square
   3. Movement
      1. Can move over any number of free squares in a diagonal fashion
      2. 
8. King
   1. Position
      1. Starts at E1/E8
   2. Movement
      1. Can only move 1 square at a time in all directions
   3. Capturing
      1. Can attack ANY piece EXCEPT for the other king/queen. Can’t move close enough to initiate capture
      2. Remember that IF you lose the king, you LOSE the game
9. Queen
   1. Position
      1. D1/D8
   2. Movement (So basically inherits from the Bishop and Rook)
      1. Can do what the rook and bishop does combined. Horizontal, vertical and diagonal.
   3. Capturing (Same default function for all).
      1. Captures pieces by moving to their squares
10. Pawns
    1. Position:
       1. A1 to H1 for 1 player
       2. A7 to H7 for other player
    2. Movement
       1. ONLY FIRST MOVE: Can move either 1 or 2 square
       2. AFTER FIRST MOVE: Only 1 square at a time
    3. Capturing
       1. Downside: IF opponent piece is directly in front of you, you CAN’T move the piece forward or capture the piece
       2. ONLY way to capture: One piece is one square forward and the other is one square forward to the left or right of the piece itself
11. Checking a piece
    1. THINK CAREFULLY before making a move that might expose you to capture
    2. Checkmate means putting king in a position where he’ll be captured bc he can’t move or be protected by any other piece.
    3. Checking:
       1. One of your piece within range of capturing the king
       2. THEN you must say check out loud so that opponent can make moves
12. Have a think about the CONDITIONS for implementing the chess moves.
    1. Appears that you need a map to store the relative position of the different chess pieces
    2. CHECK also what else is needed in the process and how you would execute on this?

Questions to ask:

1. What information does the ChessBoard class require from each of its pieces to be able to validate the moves that are submitted to it?
   1. Present position
   2. Permitted moves
   3. Capture information
2. What information does a chess piece need to know in order to determine which moves it can make?
   1. Surrounding pieces
3. What conditions must be checked after each piece has moved?
4. Is the map class necessary? Map each piece and the information surrounding it to the others.
5. Inheritance/virtual functions as the other consideration here.

Implementation classes

1. Chessboard
   1. Chess piece present positions
   2. Needs to check WHETHER checkmate etc. has occurred in the various instances.
2. Chess pieces
   1. CHECK based on present positions whether it’s allowed to move or not. So some sort of function there would be helpful to check for surrounding areas in the proposed move.
3. ~~Knights~~ 
   1. ~~Capturing~~ 
      1. ~~Can ONLY capture IFF they lend on that piece’s square~~
4. ~~Bishop~~ 
   1. ~~Starting~~ 
      1. ~~C1/F1 OR C8/F8~~
   2. ~~Capturing~~ 
      1. ~~Can capture by stepping on the pieces square~~