**Comp2522 A#2c**

**Purpose**: Testing the flexibility of your A2a/b code design with the addition of new features.

**Description**: You are to change the chess game to be a 3D chess game. Use the following guidelines:

3 boards displayed – either side by side or another view but all 3 boards must be visible at the same time. Pieces start on the bottom board.

Movements – regular chess movements remain the same however pieces can move in their appropriate direction(s) up/down levels (boards) but limited to 3 positions (e.g. bishop can move diagonally up the 3 boards crossing 3 spaces, but cannot move straight up). Think of walking up stairs. The direction the stairs are (these are virtual stairs) defines the direction passed to the Piece to determine validity. Stepping up and forward would move a piece to the next board once “space”. Stepping up and forward again would move a piece to the next board higher and again counts as one “space”. Thus a Pawn on it’s first move could move to the top and would be in the same position on the top board as if it had moved 2 steps only on the top board. Thus all pieces that can move more than 1 space at a time are limited to a max distance of 2 if they move between boards. Max between board movement is only available if they are on the top or bottom board. Pieces can move up or down but only one direction at a time (as normal) thus they cannot move up then down in the same move. The boards do allow for warp around (meaning going beyond the top or bottom board is not allowed).

Code reuse is the key here. As an example this was the structure I used:

public abstract class Board{

public abstract void init();

public abstract boolean isDiagonal(Square a, Square b);

//etc

}

public ChessBoard extends Board{…….}

public ChessBoard3D extends Board{

Board[] board = new ChessBoard[3];

//etc

}

I created a new ChessBoard3D from Board class. With a simple change to the instantiation of the Board from ChessBoard to ChessBoard3D no further code was modified. I then created an array of ChessBoard objects and used their methods as part of the ChessBoard3D methods

e.g.

public boolean isDiagonalPathClear(Square start, Square finish){

if (onSameBoard) return board[0].isDiagonalPathClear(start,finish))

else

//check diagonalpath across boards is clear return answer

}

**Marking Guide**

**Function Mark**

3D Board display 5

chess movements (queen, pawn) 20

Class hierarchy 5

Code reuse, OOP techniques 20

**Assignment due November 26 midnight.**