**Daicey – Log File**

**09/09/2016**

* Did a preliminary design and brainstorming of ideas on potential strategies for game play
* Partially implemented Dice.h, Square.h, Board.h, Player.h, Moves.h
* Dice.h needs modification in setters based on the notion that setting one face automatically sets the opposite since the sum of opposite sides is 7. Also, a strategy is needed to derive remaining sides based on a known side in the 3D model of the dice.
* Board.h needs to integrate the soldier and king dices.
* The row/column variables in Square class should be linked with the row/column variables in Dice Class to make sure update of one updates the other.
* The moves class might need a second look in the four directional roll functions.

**09/11/2016**

* Worked primarily on the Board class and created a board with dices located in respective places in respective orientation.
* Found out about the clockwise and anticlockwise nature of dices and used that to find remaining sides of a dice on being provided with 2 sides.
* SetCoordinates, SetRow, SetColumn functions inside Dice class got some further refactoring to check the validity of input parameters.
* DrawBoard function in Board class has been left halfway until a link has been established between the dice and square class.

**09/12/2016**

* Used pointers to solve the issue of linking the squares and their corresponding dices, if any.
* Some modification within the square and dice classes to integrate pointers to setup a link.
* Completed DrawBoard and UpdateBoard function in the Board class. Also, integrated a multidimensional string array that will be utilized later for serialization.
* The functions within Board class that print results to the console should be separated in a view class.
* Having issues with coming up with a proper way to access the Board class objects from other classes without compromising on Data Encapsulation.

**09/15/2016 (2.5hrs)**

* Made the GameBoard multidimensional array in Board class private and defined proper selectors/mutator functions to achieve so.
* Created a separate BoardView class and moved the display/serialization functions for the Board
* Merged the Moves class with the Player class with the intention of forming derived classes of Human/Computer later.
* Rearranged the code to meet the Project description of required classes.

**09/16/2016 (2hrs)**

* Implemented MakeAMove() function along with its supporting KeepMovingVertically() and KeepMovingLaterally() in Player.h
* Implemented a Play function in Human.h
* None of these new additions have been tested with inputs yet.

**09/17/2016 (4hr)**

* Implemented the Play() function in the Human class.
* Fixed an error of NullPtr exception by modifying the do while conditional in KeepMovingVertically() and KeepMovingLaterally() and adding breaks in switch statements of MakeAMove().
* There are some nullptr errors coming up even while making valid 90 degree moves,
* ISSUE: Moves are being made even when blockade exists on the path; also wipes the blocking dices entirely from the board while making the illegal move.
* Solved the above issues in the evening. Multiple changes in existing functions had to be made throughout the Player class.
* The destination and path validation, dice swaps works properly at this point.
* ISSUE: Any dice can capture any dice right now. Need to make it so only the opponent’s dice can be captured.

**09/18/2016 (2hr)**

* Made the functions in Player class Protected.
* Partially implemented the Game class up to the main do-while loop.
* In the evening, completed the do-while loop. The human part can be played now.
* ISSUE: If user enters char instead of int, it leads to an infinite loop. Need to validate the int input
* NEXT: Fix the existing issues, and implement the Tournament class and ErrorDisplay class. The last thing on the list are Computer’s strategy and serialization.

**09/19/2016 (2.5hr)**

* Can only capture opponent’s dice now. A simple if statement in the IsValidDestination() did the trick.
* Error handling done for user input of co-ordinates.
* Found and fixed the bug of human getting multiple move turns due to improper transfer of controls.
* Added the functionality to allow user to pick a path in case of a 90 degree turn. If the user chosen path is invalid, the code will automatically select the next best path. (Done by adding an optional parameter to MakeAMove()).