Class Structure

Controller Classes

The class structure begins with the GameLauncher class that is started from the main class. This GameLauncher class prompts user input to select a game to begin playing or quit which ends the program. If one of the three games is selected then it begins the game controller that implements the game interface for that specific game. The three games that can be played are Tic tac Toe, Order and Chaos, and Connect Four, each of which have their own game controller that instantiates the classes and dictates the flow of the game.

Game Classes

Each current game and possible future games consist of Three Main classes that Take the form of abstract classes. These are the Board, Player, and Piece Classes. For the Board and Player classes, each game uses their own concrete subclasses which have their additional unique methods and members. The Board class also used the Cell class to act as a container to hold Pieces within it. The only piece needed for these three games was a Checker that is a subclass of the Piece class.

Input Validation

The InputValidation class is a special class that is simply used to collect valid user input without halting the program or throwing an error. Each Method within it has a different use case and will continue to prompt the user for valid inputs.

Scalability & Extendibility

This design lends itself to create other turn based games as no tweaking of the existing structure is needed to add a new game to the program. A new Game just needs its own specific Board and Player subclasses to determine win conditions and game specific attributes. Additionally it needs to add any unique pieces the game has. Finally a new controller is needed to dictate the flow of the game. None of the structure needs to be changed however and each game can use pieces from other games or implement the InputValidation class. This design Works best for turn based games with boards as that is what the class structure is geared towards.

No design alterations needed to be made to allow for another turn based variant