Our project utilized some of the Behavior Patterns like the state behavioral design pattern. This design pattern is used when an object alters it behavior when an internal variable, that serves as the state of the object, changes.

In our case this was used in our Tetris game to change states depending the player input. When the program is started it presents a start screen for the player. If the player decides to continue with the game and start it then the code changes state, transitioning from the start state into the game state where the actual game of Tetris is played. This design pattern is also implemented in the pong game too. During the game the code will run the code to move the ball around and allow the user to manipulate the paddles. However, when one player wins then the code will change state and enter the game over state where it takes control of the paddles from the players and displays the game over screen where it declares that a player one.

Another design pattern is used in Tetris. This time it is not a behavioral design patter, instead it is a creational design pattern and more specifically it is the Builder design pattern. This type of pattern separates the construction of a complex object from its representation, allowing the same construction process to create various representations (Definition from the slides). This is utilized in the creation of shapes in Tetris. There is a base algorithm that allows many shapes to be made while only using this one algorithm, hence the builder design pattern.