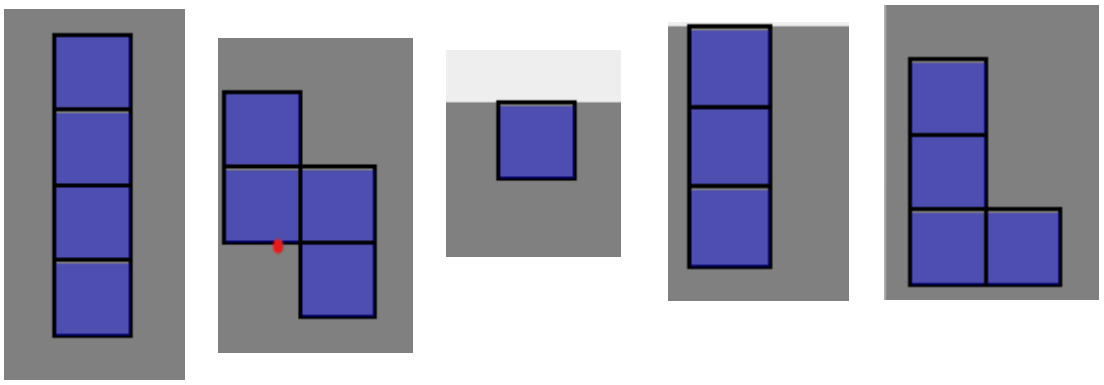
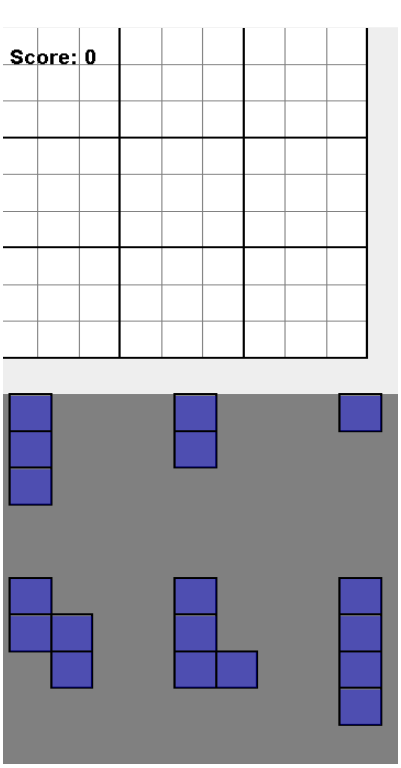
Block Puzzle Lab Assignment:

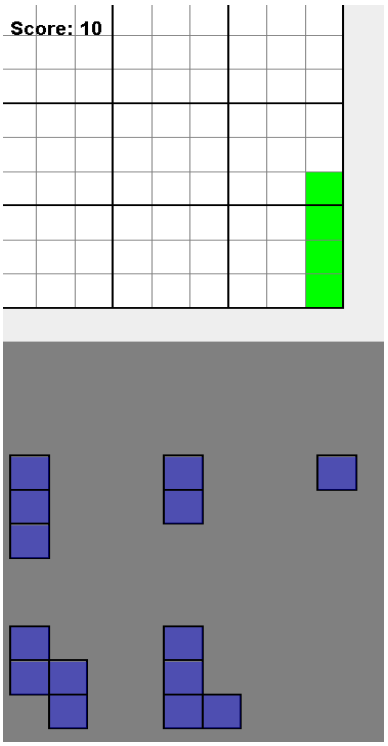
Deliverables

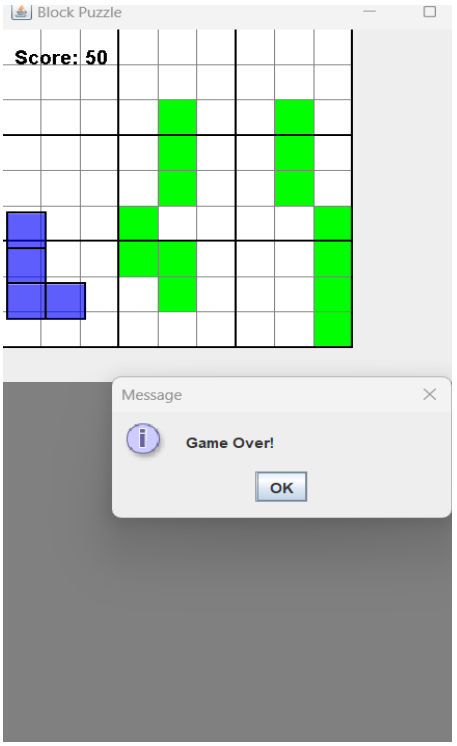
1. A set of pieces of various sizes (each piece is like a tetromino, but not limited to only 4 cells).



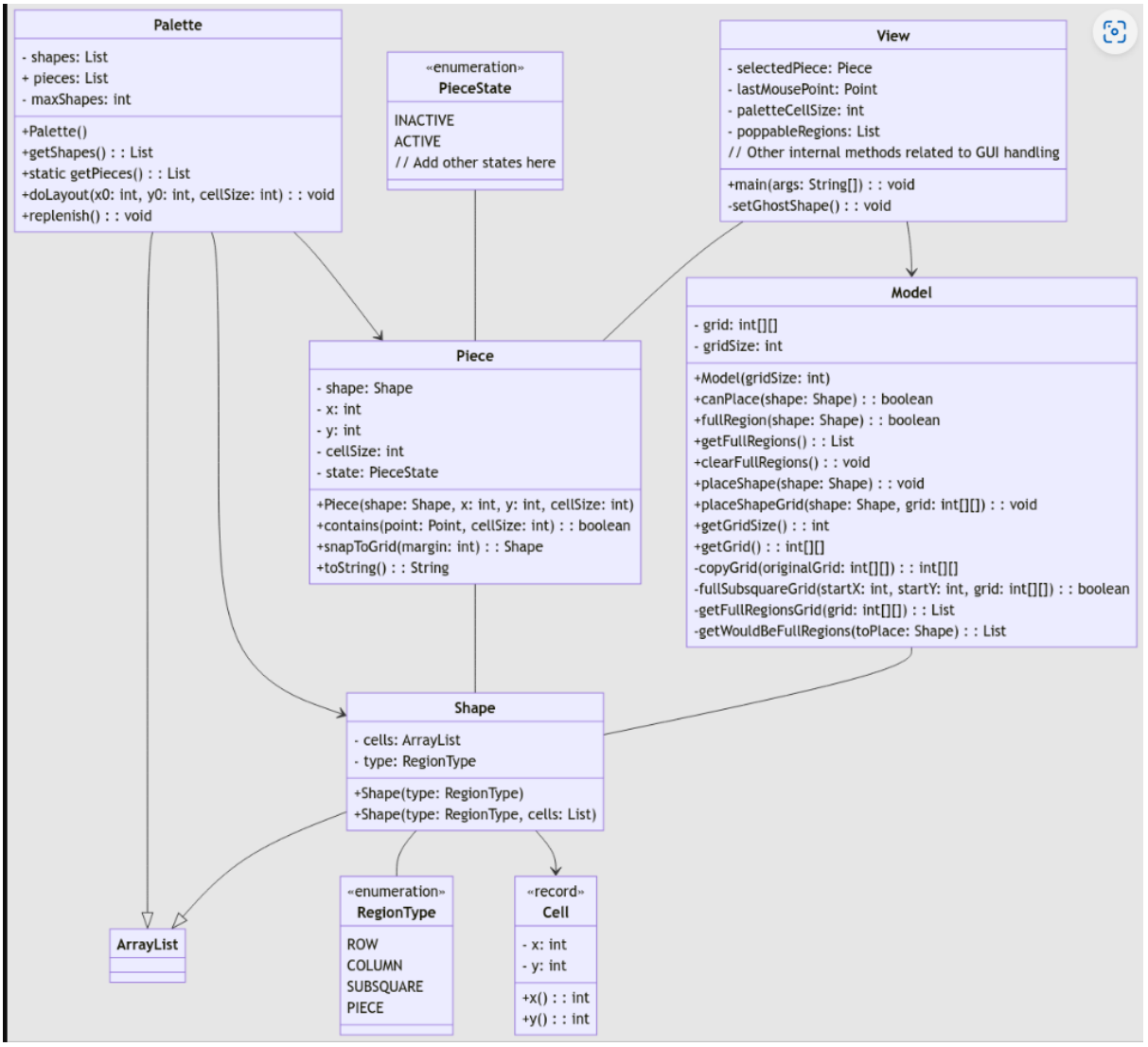
2. Puzzle model / logic (see below) & A “piece-buffer” showing the currently available pieces (it’s up to you whether any one of them can be selected by the player, or if they have to be used in a fixed order)







3. Scoring : The score increases by 10 when the player adds a piece to the grid. If the player hasn't cleared any regions with the given pieces and runs out of pieces on the palette then it is Game Over.
4. UML Class Diagram.



5. Brief Description:

View:
Manages the visual representation of the game using Swing components.
Handles user input via mouse events for dragging and placing pieces.
Model: Represents the game's logic, maintaining the grid state, handling placement of pieces, checking for full regions, and updating the score. Utilizes a grid-based approach to manage the game state.

Piece: Represents a puzzle piece with a shape, position, and state. Handles snapping pieces to the grid and contains methods for checking containment and conversion to grid coordinates.
Palette: Manages the collection and layout of pieces available for selection in the game.
ShapeSet: Generates different shapes used for the puzzle pieces and populates the Palette with these shapes.
Shape: Defines the structure of a shape, composed of cells, and includes methods to manipulate and retrieve information about the shape.
Object-Oriented Principles:

Polymorphism: Instances of Shape are used polymorphically to represent various puzzle shapes.
Composition: The View class composes the game's visual elements using Swing components.
Design Patterns:
Model-View-Controller (MVC): Divides the code into View (presentation logic), Model (game logic), and potential separation of control logic, ensuring separation of concerns.
Factory Method Pattern: The ShapeSet class incorporates a factory method pattern by providing a method getShapes() that creates and returns different shapes based on predefined rules. This promotes flexibility in creating shapes without exposing their creation logic.