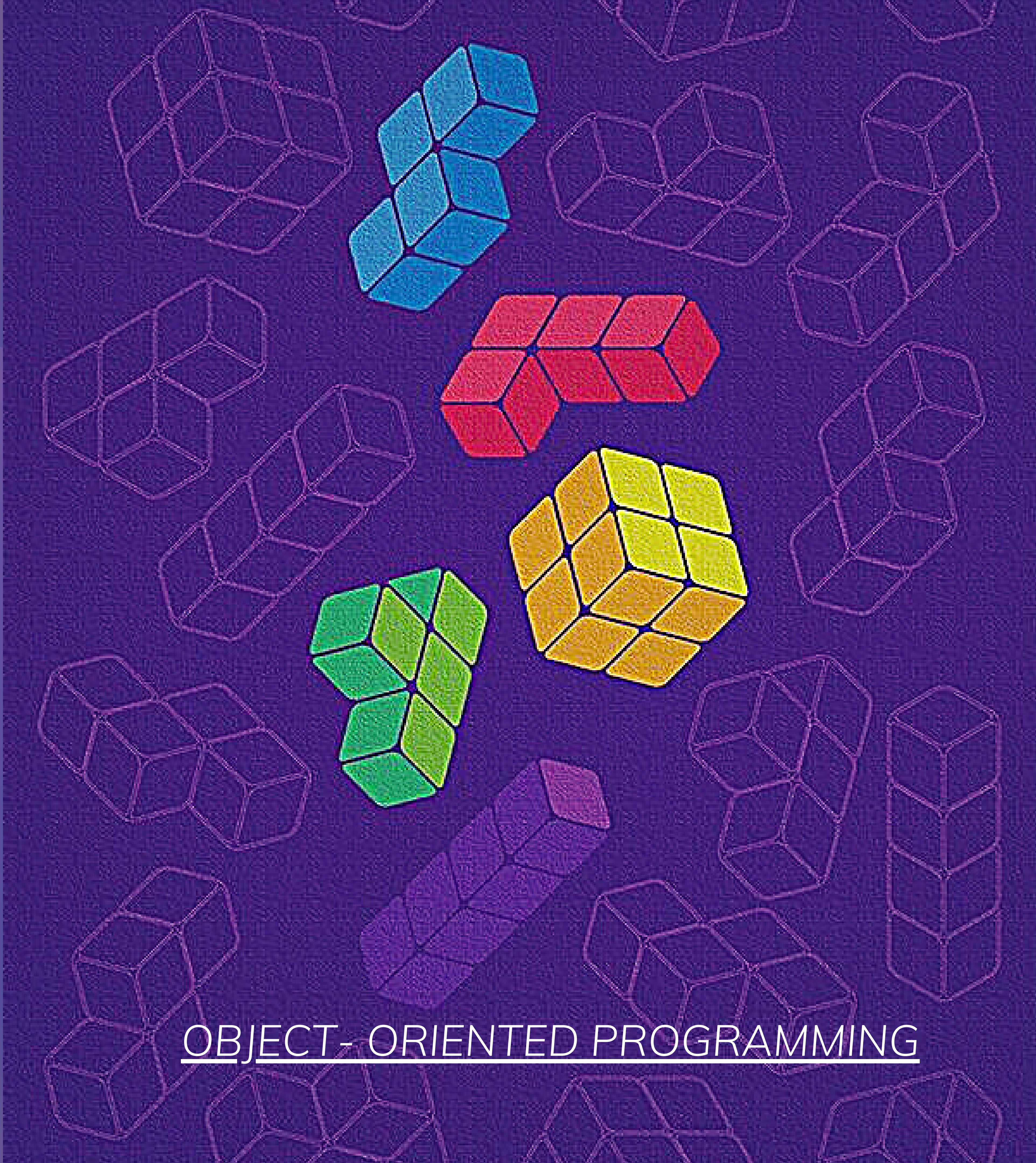
PROJECT GAME TETRIS

Members of our group:

1. Đặng Quốc Anh Duy - ITDSIU20015
2. Thái Nguyễn Minh Huân - ITITIU20207
3. Nguyễn Đức Toàn - ITCSIU21112
4. Nguyễn Song Hào Kiệt - ITDSIU20098
5. Ung Thị Hoài Thương - ITDSIU20028



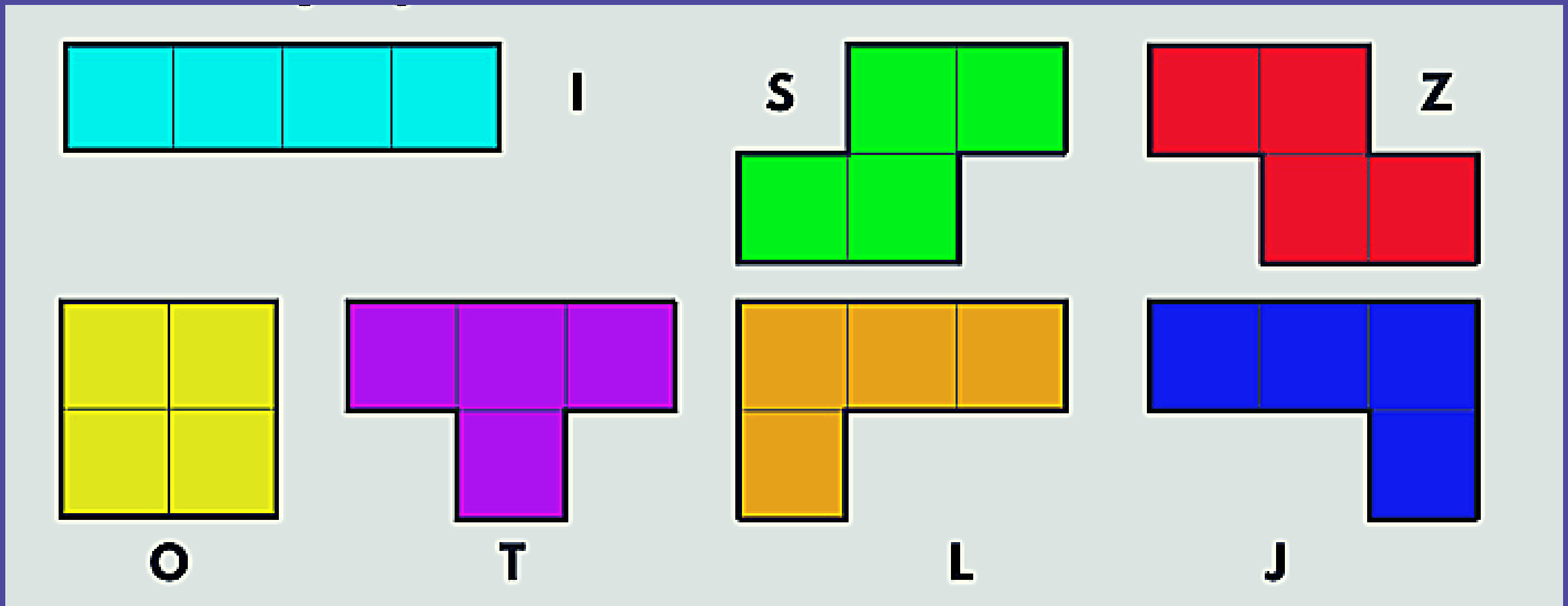
OBJECT- ORIENTED PROGRAMMING

PROJECT OUTLINE

- INTRODUCTION
- GAME RULES
- CLASS DIAGRAMS
- LIMITATION
- REFERENCE

I. INTRODUCTION

- The purpose was to build a Tetris video game system using java language on IntelliJ IDEA
- Our Tetris game is a single-player game where the computer randomly generates tetromino blocks
- Tetris is a classic puzzle game where the object of the game is to manipulate tetrominoes to stack and fit together along a horizontal line



Tetrominoes in order of I, S, Z, O, T, L, J shape

II. GAME RULES

01

How to play

02

Mechanics

03

Obstacles

04

Score pattern and 7 types of Shape

05

Pause and Refresh Button

2.1 How to play

1 Start the puzzle game

2 Move tetrominoes

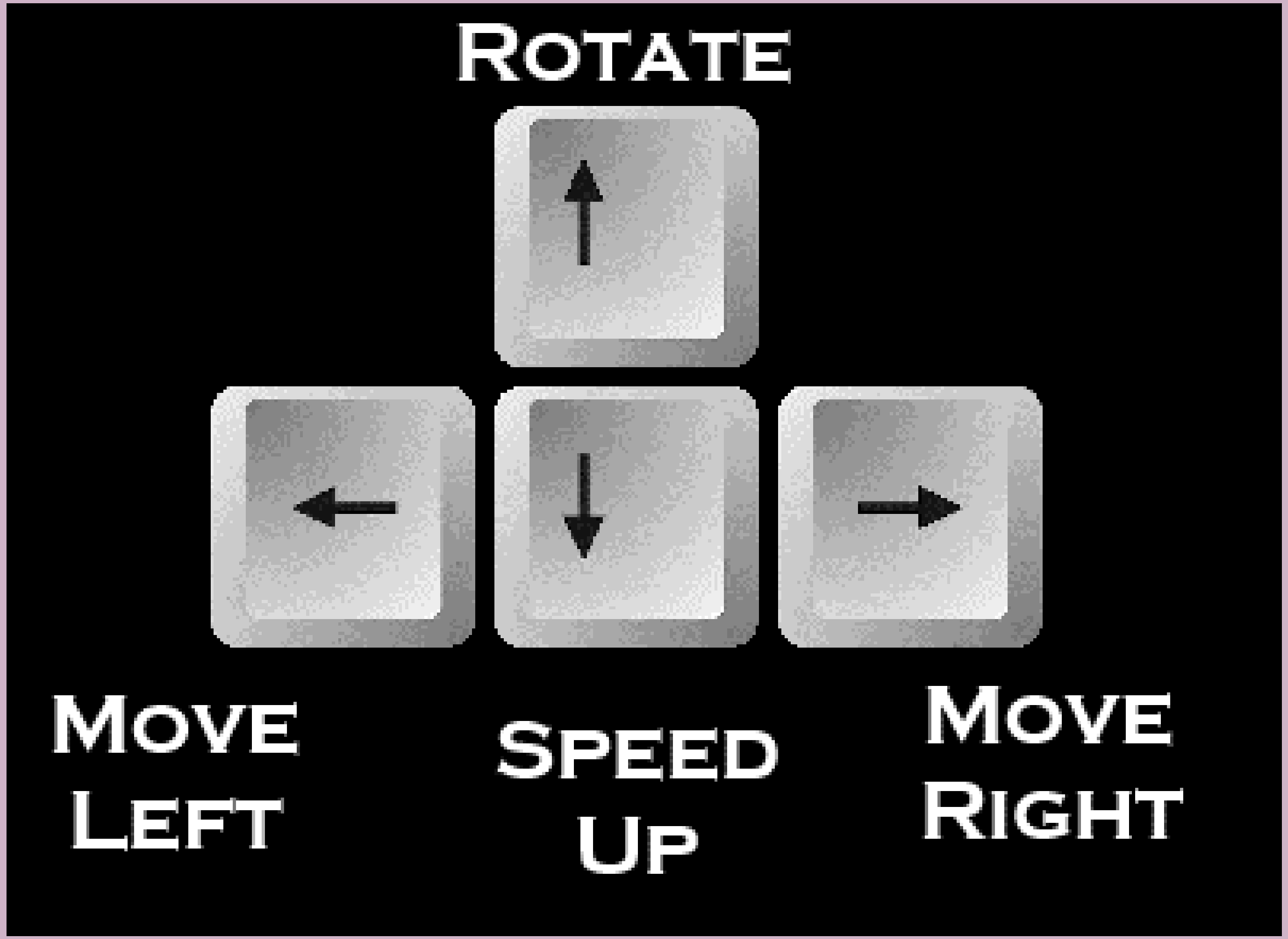
3 Understanding different types of tetrominoes

4 Complete the current block to get point

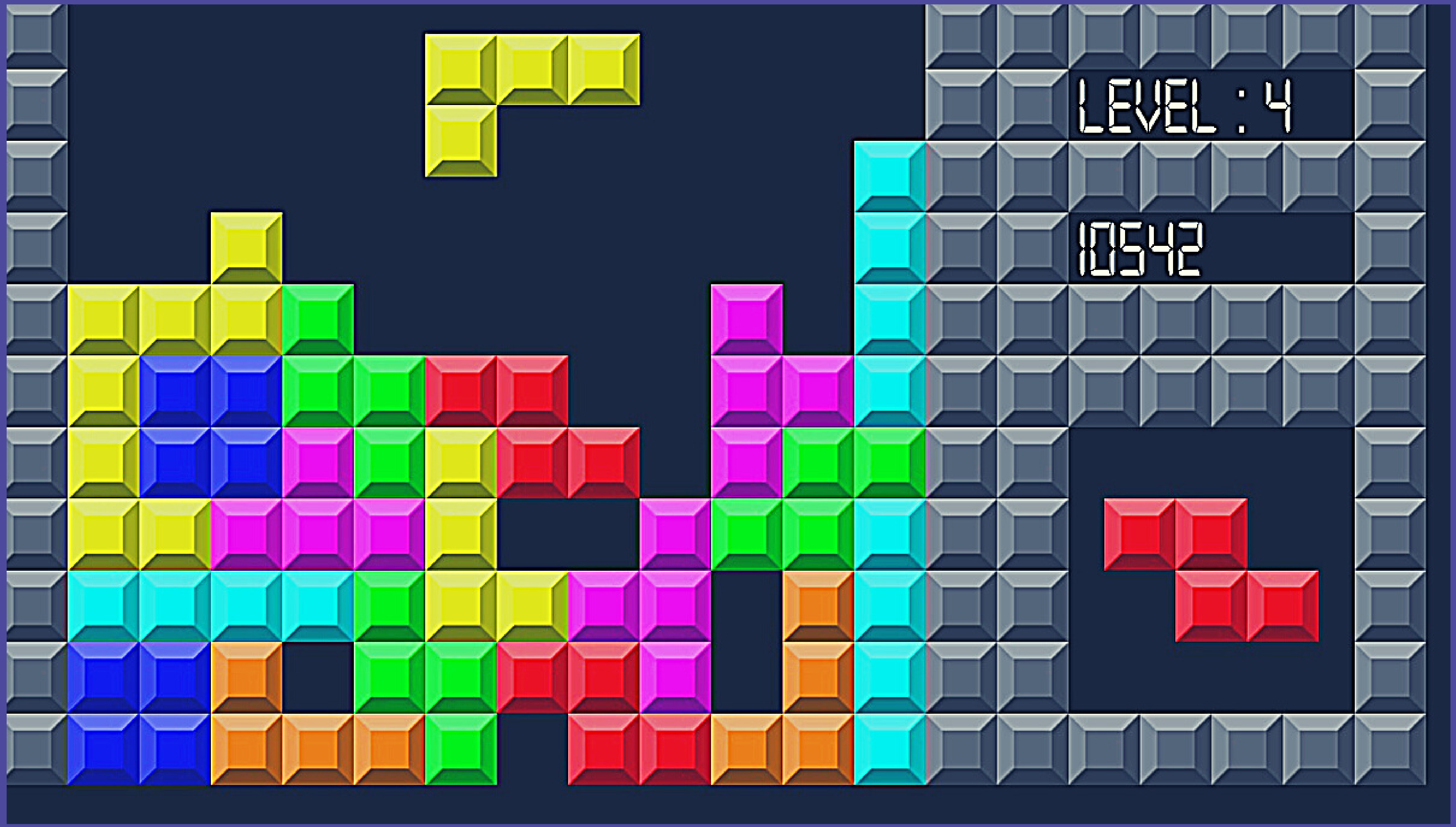
5 Check out the top right corner for the next shape

2.2 Mechanics

- A ten by twenty cell grid (10 x 20 board), no limit of time
- Left-handed version
- Blocks are randomly fallen from the top of the board with 7 different shapes
- Memorize how pieces rotate clockwise and counterclockwise.
- Press the down button to speed up the block drop



2.3 Obstacles

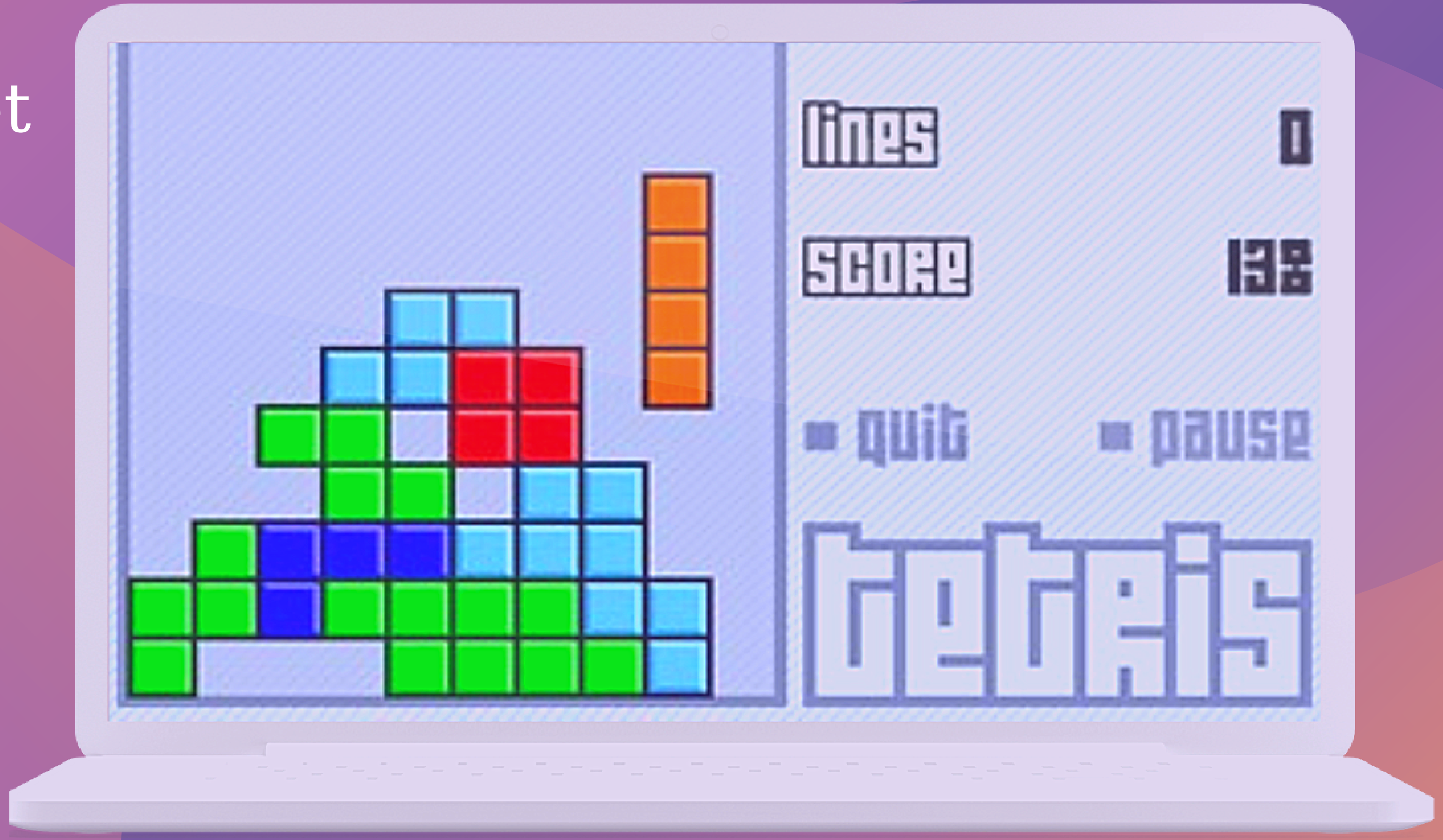


Over the last decades, Tetris has become the epitome of the 'easy to play, hard to master' gaming principle.

2.4 Score Pattern

- For each tetromino touched down you get one point

- When the tetramines are put together in a horizontal row and disappear you also get one point



2.5 Pause and Refresh Button

Refresh Button

Press and the icon above the icon to pause to play again, then the player's score will be lost and return to 0

Pause Button

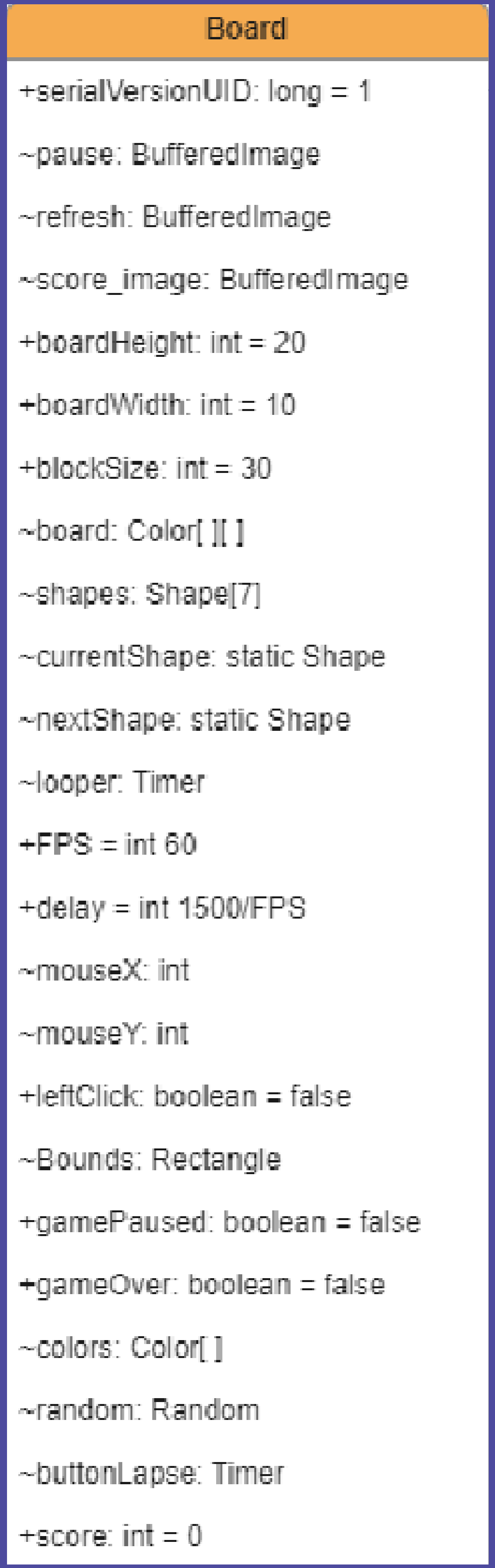
Click to pause and click again to resume playing

III. CLASS DIAGRAMS

1. Board
2. Shape
3. WindowGame
4. Title
5. ImageLoader

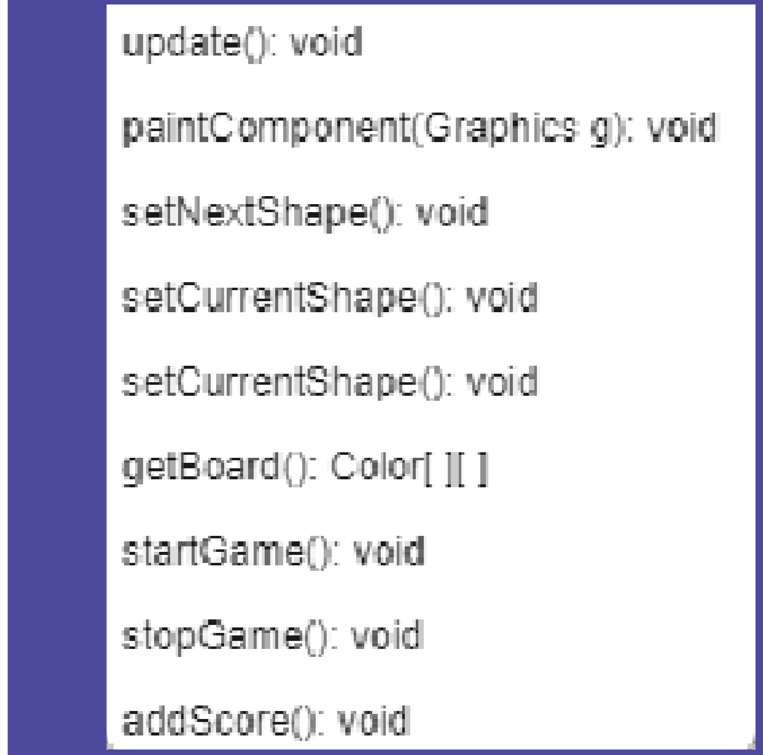
3.1 Board

- Generate 7 shape in matrix
- Timer
- Paint Component
- Function: setNextShape, setCurrentShape
- Use Key Event to link with the key binding
- StartGame, stopGame method



Board

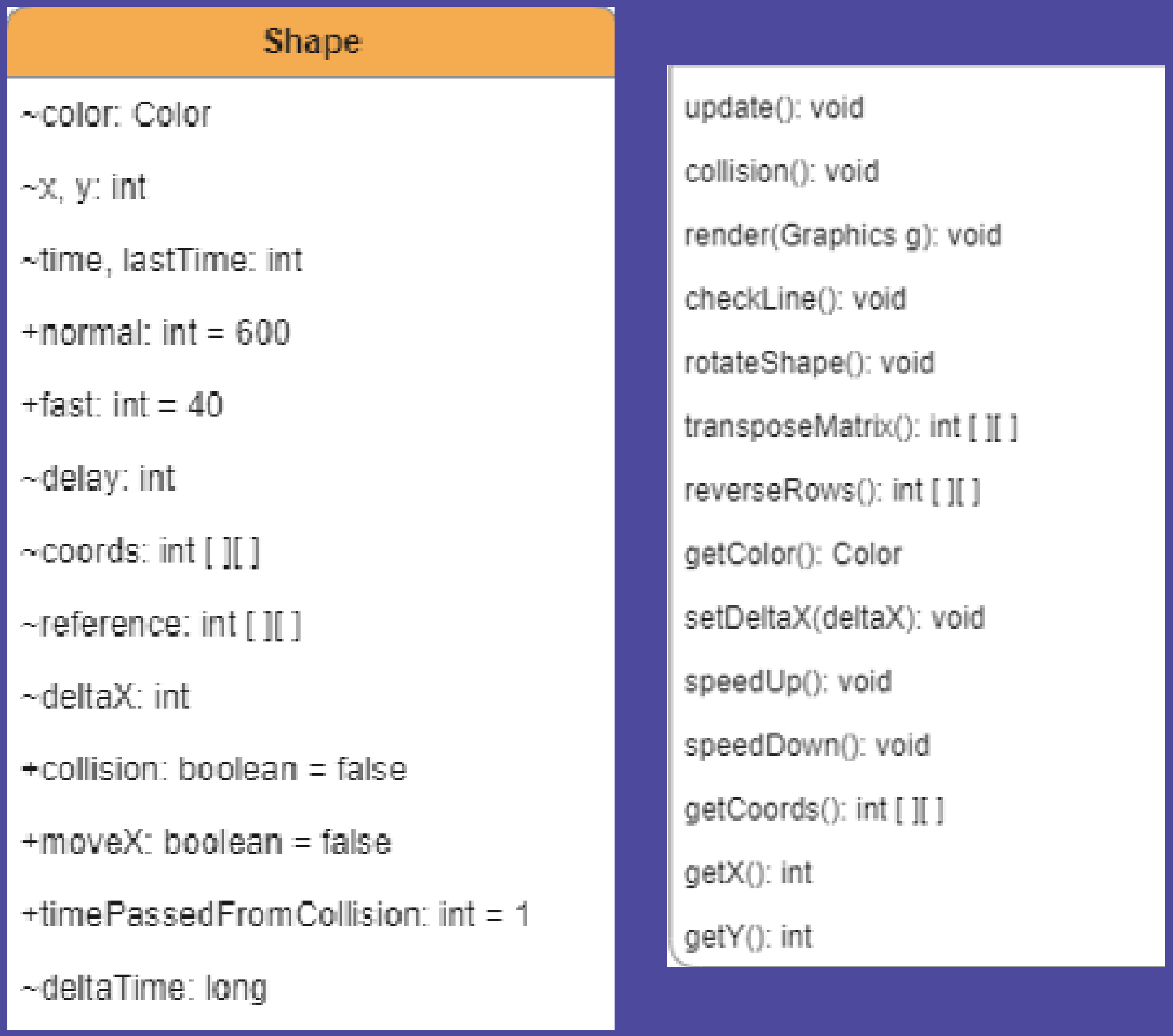
+serialVersionUID: long = 1
~pause: BufferedImage
~refresh: BufferedImage
~score_image: BufferedImage
+boardHeight: int = 20
+boardWidth: int = 10
+blockSize: int = 30
~board: Color[ ][ ]
~shapes: Shape[7]
~currentShape: static Shape
~nextShape: static Shape
~looper: Timer
+FPS = int 60
+delay = int 1500/FPS
~mouseX: int
~mouseY: int
+leftClick: boolean = false
~Bounds: Rectangle
+gamePaused: boolean = false
+gameOver: boolean = false
~colors: Color[ ]
~random: Random
~buttonLapse: Timer
+score: int = 0



update(): void
paintComponent(Graphics g): void
setNextShape(): void
setCurrentShape(): void
setCurrentShape(): void
getBoard(): Color[ ][ ]
startGame(): void
stopGame(): void
addScore(): void

3.2 Shape

- Update, collision, checkline method
  - Set up speed for the block
- Block: rotateShape, transposeMatrix, reverseRows
  - Generate getter and setter

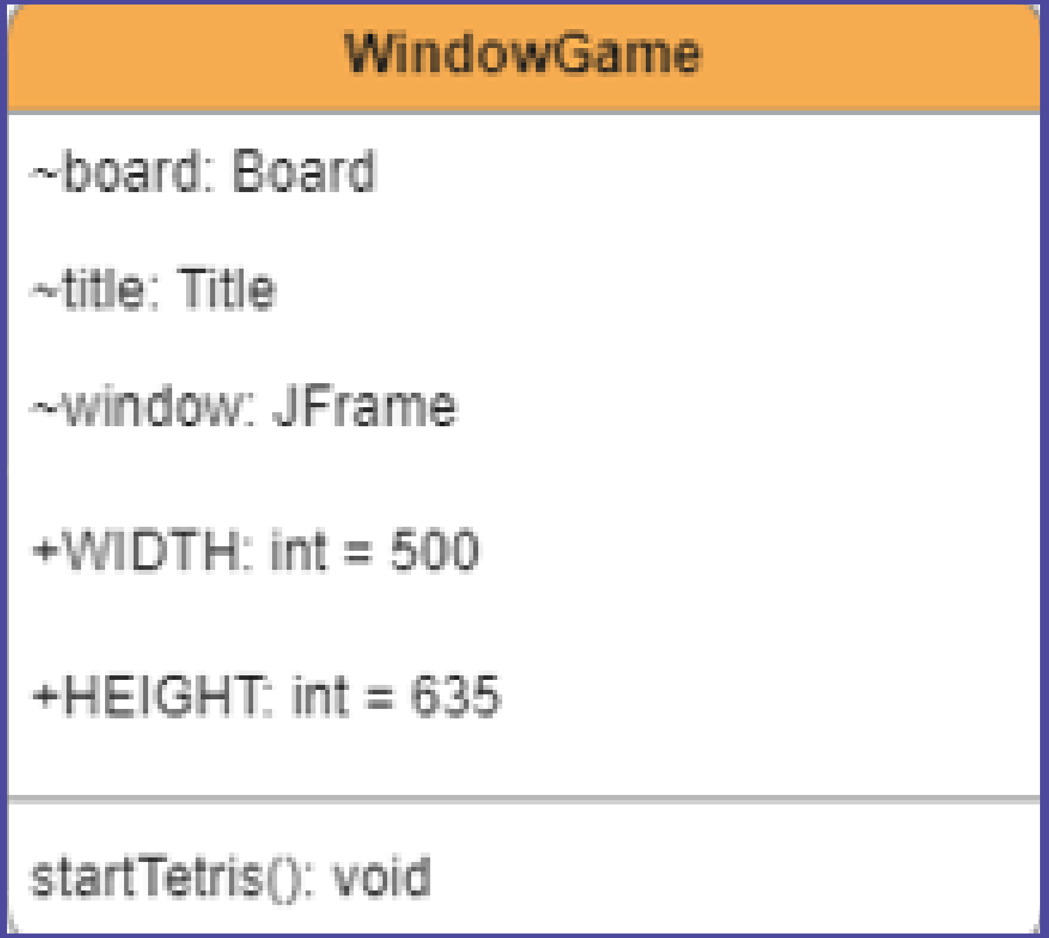


Shape

~color: Color
~x, y: int
~time, lastTime: int
+normal: int = 600
+fast: int = 40
~delay: int
~coords: int [ ][ ]
~reference: int [ ][ ]
~deltaX: int
+collision: boolean = false
+moveX: boolean = false
+timePassedFromCollision: int = 1
~deltaTime: long

update(): void
collision(): void
render(Graphics g): void
checkLine(): void
rotateShape(): void
transposeMatrix(): int [ ][ ]
reverseRows(): int [ ][ ]
getColor(): Color
setDeltaX(deltaX): void
speedUp(): void
speedDown(): void
getCoords(): int [ ][ ]
getX(): int
getY(): int

3.3 WindowGame



WindowGame

~board: Board

~title: Title

~window: JFrame

+WIDTH: int = 500

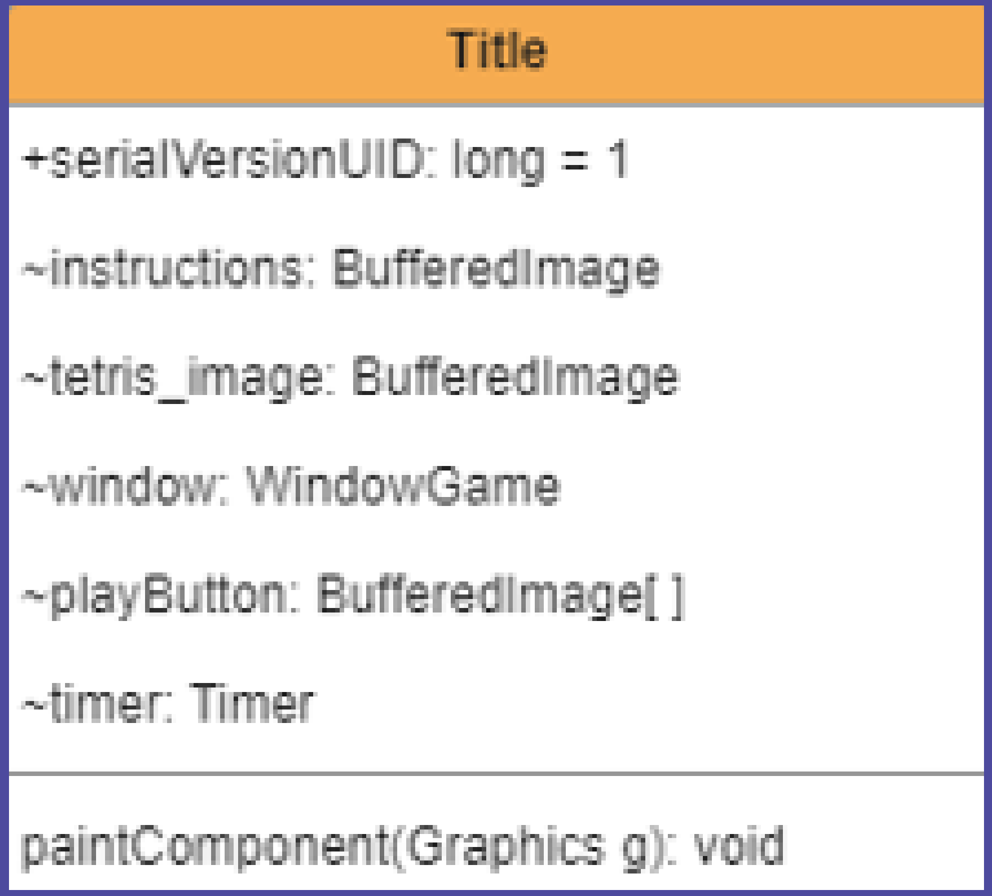
+HEIGHT: int = 635

startTetris(): void

Generate the JFrame with some other settings

StartTetris method

3.4 Title



Title

+serialVersionUID: long = 1

~instructions: BufferedImage

~tetris_image: BufferedImage

~window: WindowGame

~playButton: BufferedImage[ ]

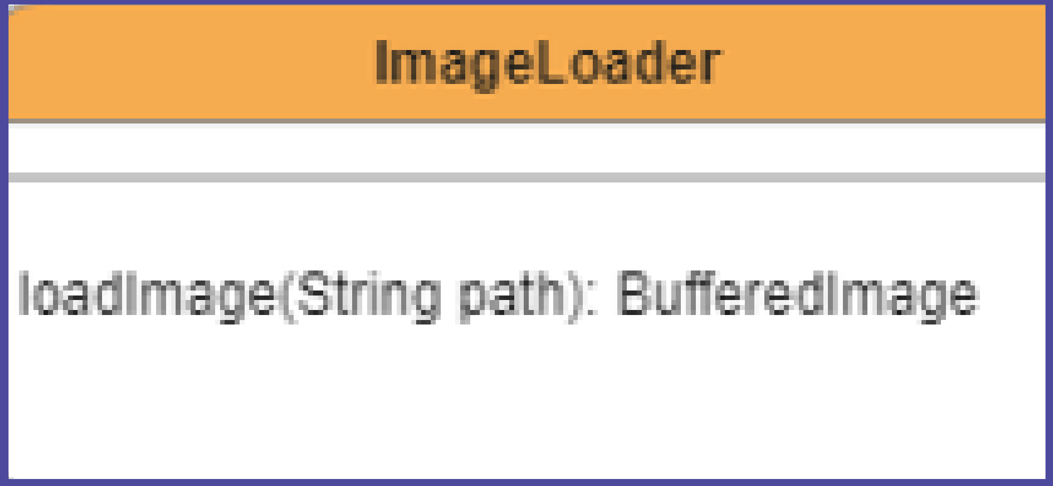
~timer: Timer

paintComponent(Graphics g): void

Import images by BufferedImage

Draw images and announce the startGame method.

3.5 Image Loader



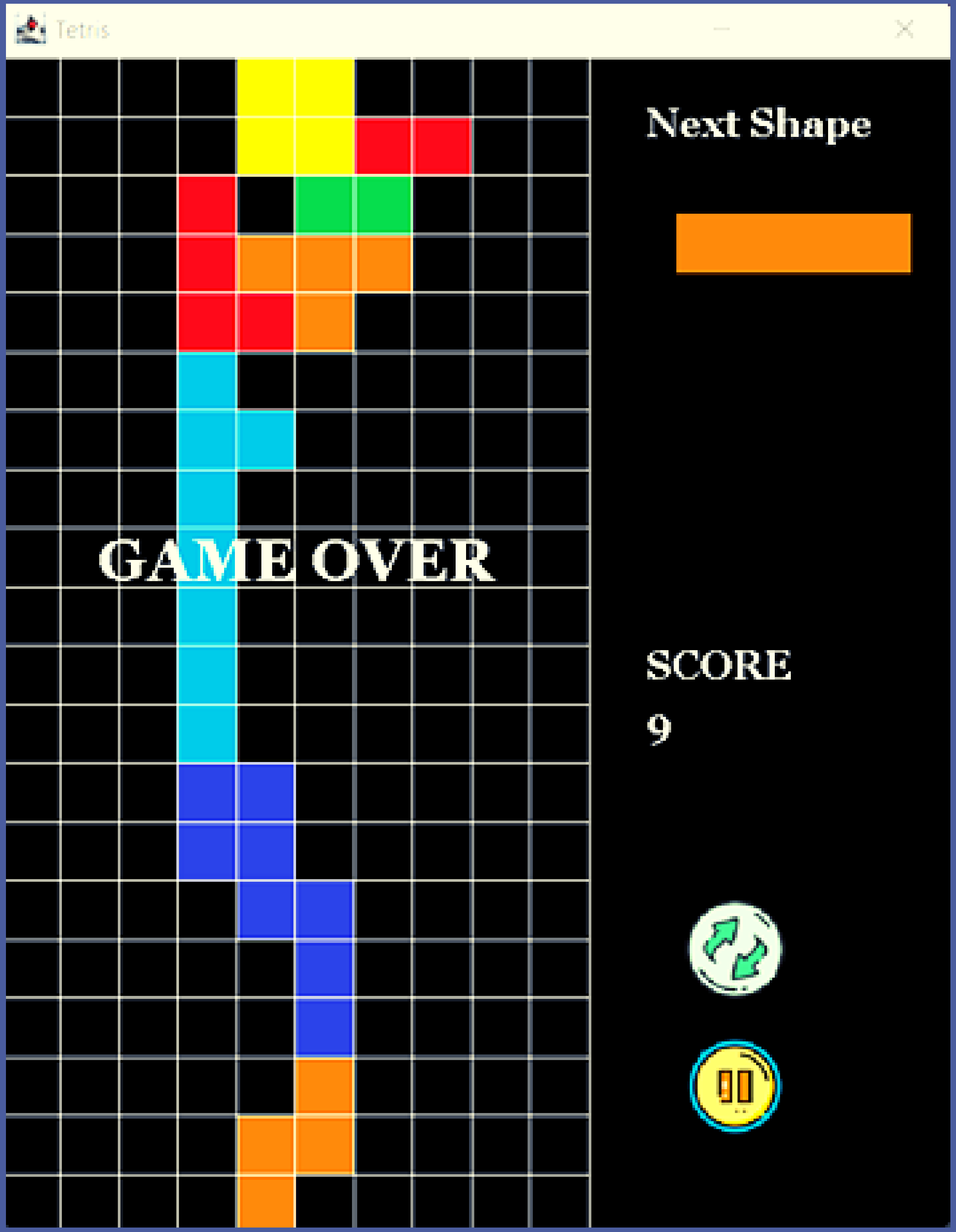
ImageLoader

loadImage(String path): BufferedImage

Loading image class

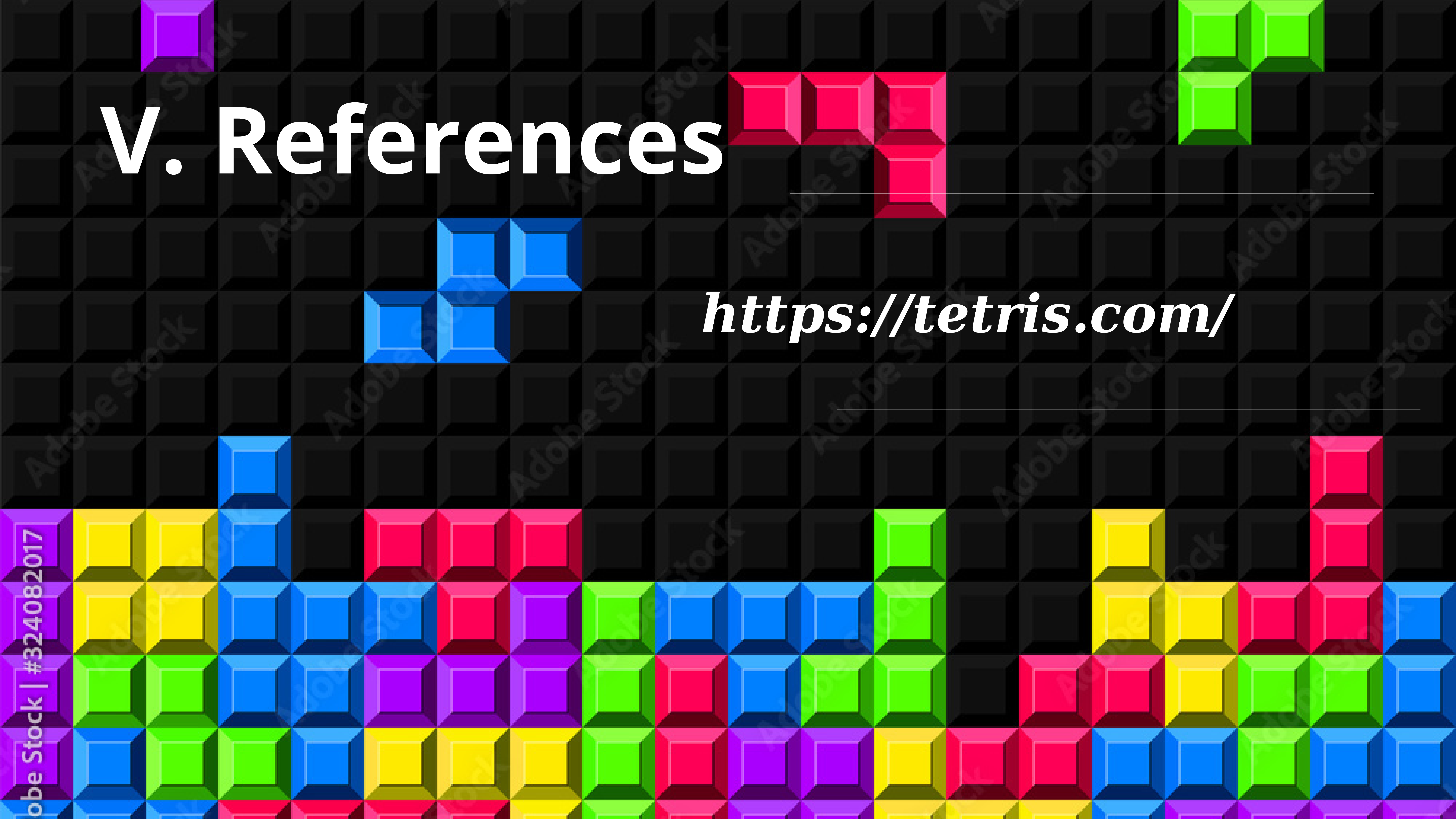
IV. Limitation

- Not using Git/GitHub (GG drive instead)
- Blocks sometimes overlap each other when we increase the falling speed
- The scoring system is not optimal



V. References

https://tetris.com/



Thank you!