### Project Report: Tetris Game Visualization in Java

\*\*Project Title:\*\* Tetris Game Visualization

\*\*Project Description:\*\*

This project implements a graphical representation of Tetris shapes using Java's Swing library. The application demonstrates basic graphics handling and user interface design in Java, showcasing the ability to render various Tetris shapes with randomized colors on a JFrame.

\*\*Key Components:\*\*

1. \*\*TetrisComp.java\*\*:

- \*\*Purpose:\*\* This class extends `JComponent` and is responsible for rendering Tetris shapes on the screen.

- \*\*Functionality:\*\*

- \*\*Shapes:\*\* The component can draw various Tetris shapes including straight lines, L-shapes, and other tetromino configurations.

- \*\*Graphics:\*\* Utilizes `Graphics2D` for drawing rectangles that represent Tetris blocks. Shapes are drawn with random colors and positions.

- \*\*Methods:\*\*

- `paintComponent(Graphics g)`: Handles the drawing of shapes. Different shapes are drawn based on the `shape` attribute.

- `setTetrimino()`: Randomly selects a Tetris shape and assigns a random color for rendering.

2. \*\*TetrisView.java\*\*:

- \*\*Purpose:\*\* This class creates and displays the main application window for the Tetris visualization.

- \*\*Functionality:\*\*

- \*\*Frame Setup:\*\* Configures a `JFrame` with a specified size and title.

- \*\*Integration:\*\* Instantiates `TetrisComp`, sets a Tetris shape, and adds it to the frame.

- \*\*Visibility:\*\* Makes the JFrame visible to the user.

\*\*Technical Details:\*\*

- \*\*Graphics Handling:\*\* The `TetrisComp` class uses Java's `Graphics2D` class to render shapes on the screen. It employs basic drawing methods such as `fill()` and `draw()` to display filled rectangles representing Tetris blocks.

- \*\*Randomization:\*\* Shapes and colors are generated randomly to create a dynamic visual representation.

- \*\*Swing Framework:\*\* The project leverages the Swing framework to create a graphical user interface, focusing on customizing the `JComponent` class for specific rendering tasks.

\*\*Challenges and Solutions:\*\*

- \*\*Shape Rendering:\*\* Correctly positioning and drawing different shapes required careful handling of coordinates and rectangle sizes.

- \*\*Color Randomization:\*\* Implementing random colors involved generating RGB values within the acceptable range.

\*\*Outcome:\*\*

This project demonstrates proficiency in Java programming, particularly in using Swing for graphical applications. It showcases the ability to create interactive and visually engaging components, providing a foundation for more complex graphical user interface projects.

\*\*Skills Demonstrated:\*\*

- Java Swing and AWT

- Graphics and Drawing in Java

- Object-Oriented Programming

- GUI Design and Implementation

This project reflects the ability to develop and visualize graphical elements in Java, making it a strong addition to a resume for internships or positions that require skills in software development and graphical programming.