**Report: Simple Shooter Game Implementation**

**Introduction:** This report presents a simple implementation of a shooter game coded in C. The game involves a shooter at the bottom of the screen that can move horizontally, aiming to shoot falling squares of different colors. The shooter can move left or right based on user input. If the shooter misses three squares, the game ends. The report provides an overview of the code structure, key functions, and game logic.

**Code Structure:** The code is structured into several functions to handle different aspects of the game:

1. **write\_pixel**: Sets a single pixel on the screen with a specified color.
2. **clear\_screen**: Clears the entire screen.
3. **clear\_screen2**: Clears both the pixel data and the text area on the screen.
4. **clear\_square**: Clears the area where a square was previously drawn.
5. **draw\_shooter**: Draws the shooter at its current position.
6. **update\_shooter\_position**: Updates the position of the shooter based on user input.
7. **delay**: Introduces a delay in the game loop.
8. **write\_char**: Writes a character to the text area on the screen.
9. **check\_condition**: Checks if two objects intersect.

The **game** function contains the main game loop, where squares fall randomly from the top of the screen. The shooter's position is updated based on user input, and squares are drawn and cleared accordingly. The game continues until the player misses three squares.

The **end** function handles the end of the game, displaying a "GAME OVER" message and instructions to restart. It waits for the user to press a specific button to restart the game.

**Game Logic:**

* The game initializes by seeding the random number generator and entering a continuous loop.
* Squares of random colors fall from the top of the screen.
* The shooter's position is updated based on user input from switches.
* If the shooter intersects with a falling square (except red), the player scores a point.
* If the shooter misses three squares, the game ends, displaying a "GAME OVER" message.

**Conclusion:** This report presents a simple shooter game implemented in C. The game features a shooter controlled by the player, shooting falling squares. The code demonstrates basic game mechanics, including user input handling, graphics rendering, and game logic. The game provides an entertaining experience for players, challenging them to aim accurately and react quickly to falling squares.