Project Plan for Haskell Block Breaker Game

By Daniel Morales-Garcia, John Tyler, Darrion Romero

Game Description:
We will develop our interpretation of the classic Block Breaker game in Haskell. Players will move a paddle horizontally at the bottom of the screen to bounce a ball upwards to break blocks arranged in various patterns at the top of the screen. The game ends when all blocks are broken or the player loses their last ball.

Core Features:

- Paddle Control: Players can move the paddle left and right using keyboard inputs.
- Ball Mechanics: The ball will have a consistent movement algorithm, bouncing off the paddle, walls, and blocks.
- Block Mechanics: Blocks will disappear when hit by the ball, and the score will be awarded based on block type.
- Game Progression: The game will start with easier levels, increasing in difficulty as the player progresses. For demonstration, have the ability to skip to harder levels
- User Interface: A simple GUI will display the score, the number of lives left, and end-of-game messages.

Technology Stack:

- Gloss for rendering game graphics.
- Haskell's Random Library for ball movement randomness.
- Simple DirectMedia Layer (SDL): Potential consideration for advanced graphics

Development Strategy:

Phase 1 - Setup:

- Establish Haskell project structure.
- Set up Gloss for 2D rendering.

Phase 2 - Core Gameplay:

- Implement paddle and ball mechanics. Create the first level with a single pattern of blocks.

Phase 3 - Game Dynamics:

- Add collision detection and response between the ball, paddle, and blocks.
- Implement a scoring system based on block destruction.

Phase 4 - Level Design:

- Design additional levels with unique block patterns.
- Introduce special blocks (multi-hit, explosive, etc..).

Phase 5 - UI:

- Develop the user interface elements.

Phase 6 - Testing and Polishing:

- Playtest and iterate on game design.
- Optimize performance and finalize the game.