A white background with black text

Description automatically generated

Overview

“sky jumper” will be a basic 2d arcade game to demonstrate the overall flow of a game using pygame and simpleGE.

The premise is extremely simple: the player is the actor trying to jump the tiles and also achieve coins on the tile while the tile is moving, the higher the coin the player get the higher is score once he fall from any of the tile that’s the end of the game.

**Algorithm**

1. **Initialize the Game**:
   * Set up the game environment including background, platforms, and the main character Charlie.
2. **Game Loop**:
   * **Event Handling**: Process keyboard inputs to control character movement (left, right, jump).
   * **Update Game State**:
     + Move platforms and recycle them when they move out of view.
     + Check collisions between Charlie and platforms to allow jumping and landing.
     + Update Charlie's state (e.g., check if he is in the air or falling off the screen).
   * **Score and Game Progress**:
     + Increment the score over time and increase the platform speed.
     + Transition to instructions or end game based on player's actions or game conditions.
3. **Render Game**:
   * Update the display with the new positions of Charlie, platforms, and updated scores and time.
4. **Repeat or End**:
   * Depending on player input and game conditions, either continue the game loop or transition to an instructions screen or end the game.

**Pseudocode:**

function main()

Initialize pygame

Set keepGoing to True

Initialize score to 0

while keepGoing is True

Show Instruction Scene

If player chooses to play

Start Game Scene

Continue playing until game conditions change

Update score after game ends

Else

Set keepGoing to False

Quit pygame

class Charlie extends Sprite

Initialize Charlie (load image, set size and initial position)

function process()

Apply gravity if in the air

Check for ground collision and reset jumping state

Handle horizontal movements based on arrow keys

Handle jumping based on arrow key and in-air state

Check for falling off the screen and trigger transition

class Platform extends Sprite

Initialize Platform (load image, set size, set initial position and movement speed)

function process()

Move horizontally

Recycle position when moving out of screen

Randomly decide to make the platform visible again

class Game extends Scene

Initialize game (set background, create platforms and character, start timers)

function process()

If gameOver or transition flag is set, stop the game

Otherwise, update game state

Update timers and scores based on time elapsed

Update platform speed periodically

class Instruction extends Scene

Initialize instructions (set background, display controls and tips)

Handle user input for playing or quitting

function process()

Detect button clicks or key presses to continue or quit

function initializePygame()

Set up pygame with necessary configurations

function createSprites()

Create game objects (characters, platforms) and set their properties

function updateDisplay()

Render all sprites and text to the screen

function checkCollisions()

Detect and handle collisions between Charlie and platforms

function updateGameState()

Update the positions, states, and check game-ending conditions