Final Project Design Document

By Jackson Taylor for CSC221 Final Project

Introduction

Project Functionality

This game takes user input in the form of arrow keys and moves a snake around the defined bordered area. If you orient the snake to collect randomly spawning food (represented by red pixels), the snake will increase in length by one pixel. If collision occurs between the head of the snake and either itself of the border of the defined area, the game will come to an end.

Design Process

This game was created via coding on Python through many instances of trial and error and due revisions. It utilizes a 20x20 grid to define the snake's movement and also randomly spawns food across the grid. Color values were assigned to the snake and food to differentiate the two. Difficulties encountered include collision detection and movement capabilities. Accurealty defining when the game was supposed to end proved to be a difficulty as frequently it would end at the incorrect time.

Project Development

Pseudocode

```
Initialize Pygame and game display window

Set screen dimensions (width, height)
Set grid size and calculate grid dimensions
```

```
Define colors (black, white, red, green)
Set snake properties (color, speed, block size)
Load font for game messages

Define function to display messages on screen

Define main game loop:
    Initialize game state (snake position, direction, length, list, food location)

    While game is not over:
        Check for user input:
            - If QUIT event, end game
            - If arrow key pressed, change snake direction

        Update snake position based on direction

        If snake hits wall, end game

        Fill screen with background color
        Draw food at current food location

        Add new head position to snake list
        If snake length exceeded, remove the oldest segment

        Check for collision with itself:
            - If snake head touches any body segment, end game

        Draw the entire snake on the screen

        If snake head reaches food:
```

```
        - Increase snake length
        - Generate new food position at random

    Update display and control frame rate

Display "Game Over" and restart/quit message

Wait for player input:
    - If Q is pressed, quit game
    - If P is pressed, restart game loop
```

Flowchart

1. **Start the program**
    - Initialize Pygame, set screen size, grid, colors, and font.
2. **Enter main game loop**
    - While the game is active:
        - Handle user input (arrow keys to move, quit event).
        - Update the snake's direction and position.
        - Check for collisions:
            - With walls → End game.
            - With itself → End game.

        - If the snake eats the food:
            - Increase its length.
            - Spawn new food at a random location.
        - Redraw the screen with the updated snake and food.
        - Control the game speed with a clock tick.

3. **Game Over**
    - Display "Game Over" message and options to quit or play again.
    - Wait for user input:
        - Press **Q** to quit.
        - Press **P** to restart the game.

UML Diagram

Game Class

- Attributes:
    - `screen`: The game display surface
    - `clock`: Controls the game loop speed

- Methods:
  - `run()`: Starts the game loop
  - `update()`: Updates the game state (e.g., moves the snake)
  - `render()`: Draws the snake, food, and game screen

Snake Class

- Attributes:
  - `body`: List of segments that make up the snake
  - `direction`: The current movement direction of the snake
- Methods:
  - `move()`: Moves the snake in the current direction
  - `grow()`: Increases the size of the snake
  - `draw()`: Renders the snake on the screen

Food Class

- Attributes:

    - `x`: The x-coordinate of the food
    - `y`: The y-coordinate of the food

- Methods:
    - `spawn()`: Randomly places the food on the screen
    - `draw()`: Renders the food on the screen

Requirements

Board Size / Play Area – Project has a defined area with borders.

Snake Movement – Snake has the ability to move in all four directions by player control.

Snake Growth – Each food consumed results in snake length increasing by 1 unit.

Food Generation – Food randomly generates on the play area.

Collision Detection – Game ends if the snake's head collides with the body or borders.

Game Over – When collision occurs, game over prompt appears.

Restart Option – User input is received to play again or quit.