Final Project Design Document

By Drexel Zybko for CSC221 Final Project

Introduction

I created a classic Snake game using Python and the Pygame library. In this game, I control a snake that moves around a 2D grid, trying to eat food that appears randomly on the screen. Every time the snake eats, it grows longer and my score increases. The game ends if the snake runs into itself or the wall. To make the experience more immersive, I included calm background music that loops during gameplay, and a startling, funny sound effect that plays when the player loses. I also added a pause feature so players can stop and resume gameplay smoothly, and ensured the snake cannot accidentally turn directly into itself.

Project Functionality

- The snake moves in a grid of 25x25 tiles.
- Movement is controlled using the arrow keys.
- Pressing the P key pauses or resumes the game.
- Food appears in a random spot on the grid that is not already occupied by the snake.
- Eating food increases the length of the snake and the score by 1.
- If the snake runs into itself or the edge of the grid, the game ends.
- Background music plays in a loop during gameplay.
- A separate scream sound plays once when the player loses, followed by a reset of the calm music.
- The score is displayed in real-time, and a custom game over screen shows the final score and restart instructions.

Design decisions included:

- Preventing the snake from reversing direction and running into itself.
- Ensuring food never spawns inside the snake's body.
- Implementing a finite game loop that ends on collision.
- Including a pause/resume feature for improved playability.
- Enhancing player engagement with sound design and visual clarity.

Design Process

I started by setting up the basic grid and the snake. Once movement was working, I added food placement

and collision detection. I then implemented score tracking and game over logic. A key improvement was fixing the issue where the snake could turn into itself when moving too fast'this was solved by checking against the current movement direction. Another highlight was refining the sound behavior: the background music now resets cleanly after a game over, and the scream plays only once. I also added a pause function with a dedicated key and made stylistic enhancements like a dark background for contrast. This project gave me deeper experience with event handling, sound integration, and game control flow in Python.

Project Development

This project was developed in Python using the Pygame library. It was run and tested locally in a Python environment with all media files (.mp3) located in the same directory as the program. I followed an iterative development process: implement one feature at a time, test it, then move to the next. I used pygame.mixer to manage background music and sound effects, and created functions to handle input, game state updates, and drawing.

Pseudocode

Initialize Pygame

Set up screen, colors, constants

Load background music and game over sound

Define function to draw text

Define function to place food randomly (not inside the snake)

Start main game loop:

- Handle player input for direction and pause
- If paused: display pause screen and skip updates
- Move snake in current direction
- Prevent reversing into itself
- Check for collisions with self or wall
- If collision: stop music, play scream, delay, show score
- If food eaten: increase score, grow snake, place new food
- Else: move snake by removing tail
- Draw snake, food, and score

Final Project Design Document

By Drexel Zybko for CSC221 Final Project

After game ends:

- Show game over screen and final score
- Wait for player to press R (restart) or Q (quit)

Flowchart

[Start] ' [Initialize Game] ' [Enter Game Loop] ' [Handle Input / Pause] ' [Update Snake Position]

'

[Check for Collision]

' Yes ' [Stop Music + Play Scream] ' [Game Over Screen] ' [Restart/Quit]

' No ' [Check if Food is Eaten]

' Yes ' [Grow Snake + Increase Score]

' No ' [Move Snake Normally]

'

[Draw Everything] ' [Loop Back to Handle Input]

UML Diagram

(Since this project is procedural and not object-oriented, a traditional UML class diagram isn't necessary. If desired, a simple diagram could be created to illustrate functional relationships such as game(), main(), draw_text(), and random_food_position().)

Requirements

- Includes user interaction via keyboard (arrow keys and pause)
- Uses processing: loops, conditionals, function calls
- Outputs formatted score and end messages
- Implements custom functions (draw_text, random_food_position, etc.)
- Uses an external library (Pygame)

Key Features and Functional Requirements:

1. Board Size / Play Area:
   - 25x25 tile grid (each tile is 20 pixels)

Final Project Design Document

By Drexel Zybko for CSC221 Final Project

2. Snake Movement:
   - Controlled with arrow keys; continuous movement
   - Movement blocked from reversing into self
3. Snake Growth:
   - Each food item eaten grows the snake by one unit
4. Food Generation:
   - Food placed randomly in open spaces only
5. Collision Detection:
   - Game ends on collision with wall or snake body
6. Game Over and Score:
   - Score displayed live during game
   - Game Over screen shows final score and restart/quit instructions
7. Restart Option:
   - Player can press R to restart or Q to quit
8. Sound Effects and Music:
   - Calm background music loops during gameplay
   - Scream sound plays once at game over and resets after restart
9. Pause Feature:
   - Player can press P to pause or resume gameplay at any time