Snake Game using Python

Indian Institute of Information Technology, Ranchi

Name: Hariom Nagar
Roll No: 2023UG3024
Submitted to: Shivang Tripathi

November 2024

Contents

1 Introduction

The Snake Game is a classic arcade game where the player controls a snake that grows in size as it consumes food, with the challenge being to avoid running into the walls or itself. This project implements a simple version of the game using **Pygame**, a Python library designed for writing video games.

This report provides an overview of the development process, key components, and functionality of the game.

2 Objective

The objective of this project is:

1. To implement a functional Snake Game using the **Pygame** library.
2. To learn game development principles like rendering graphics, handling input, and managing game loops.
3. To develop programming skills by using Pygame's event handling and drawing tools.

3 Development Environment

- **Programming Language:** Python
- **Libraries Used:** Pygame, Time, Random
- **Editor:** Any Python IDE (e.g., PyCharm, Visual Studio Code)
- **System Requirements:** Python 3.6 or higher, Pygame library installed

4 Implementation

4.1 Pygame Initialization

The game begins by initializing the Pygame library:

```
pygame.init()
```

4.2 Game Display

The display is set up with a resolution of 800x600 pixels:

```
width = 800
height = 600
display = pygame.display.set_mode((width, height))
pygame.display.set_caption('Snake Game')
```

4.3 Core Components

4.3.1 Snake Drawing

The snake is drawn as a series of black rectangles:

```
def our_snake(snake_block, snake_list):
    for x in snake_list:
        pygame.draw.rect(display, black, [x[0], x[1], snake_block
           , snake_block])
```

4.3.2 Food Placement

Random food positions are generated within the bounds of the screen:

```
foodx = round(random.randrange(0, width - snake_block) / 10.0) *
   10.0
foody = round(random.randrange(0, height - snake_block) / 10.0) *
    10.0
```

4.4 Game Loop

The game loop manages the game's state, processes user input, updates game elements, and renders frames.

4.4.1 Event Handling

The game responds to keyboard events to control the snake's direction:

```
if event.key == pygame.K_LEFT:
    x1_change = -snake_block
    y1_change = 0
```

4.4.2 Collision Detection

The game ends if the snake hits the screen boundaries or collides with itself. When the snake eats food, it grows in length.

5 Challenges

- **Boundary Collision:** Ensuring the game ends when the snake collides with the screen edges.
- **Smooth Movement:** Managing the speed and directional changes of the snake.
- **Snake Growth:** Implementing logic to grow the snake correctly without visual glitches.

6 Conclusion

This project successfully implements a functional Snake Game using Pygame. The game is interactive, visually appealing, and demonstrates key programming concepts like event handling, loops, and collision detection.

6.1 Future Enhancements

- **Improved Graphics:** Add animations and textures.
- **Scoring System:** Display the player's score on the screen.
- **Levels and Speed:** Introduce progressive difficulty levels.
- **Sound Effects:** Add sound effects for actions like eating and losing.