SNAKE GAME PROJECT

Snake Game in Python – Project Report

1. Introduction

The Snake Game is a classic arcade game where the player controls a snake to eat food, growing longer with each bite. The challenge lies in avoiding collisions with the walls or the snake's own body. This project implements the game using Python and the Pygame library, focusing on simplicity, interactivity, and smooth gameplay.

2. Objectives

Develop an interactive game using Python.

Implement real-time user input handling.

Apply collision detection algorithms.

Track and display the player's score dynamically.

Enhance programming skills in game development.

3. Software & Hardware Requirements

Software:

Python 3.x

Pygame library

Code editor (VS Code / PyCharm)

Hardware:

Minimum 2 GB RAM

Dual-core processor

Keyboard for input

4. System Design

Game Components:

Snake – Represented as a list of coordinates (segments).

Food – Randomly generated on the game grid.

Scoreboard – Displays the current score.

Game Loop – Handles events, updates positions, and renders graphics.

Flow:

Initialize game window and variables.

Capture keyboard input for snake movement.

Update snake position and check for collisions.

If food is eaten, increase length and score.

End game on collision with wall or self.

5. Implementation Details

Language: Python

Library: Pygame for graphics, sound, and event handling.

Data Structures: Lists for storing snake body coordinates.

Collision Detection:

6. Sample Code Snippet

Pythonimport pygame, random

pygame.init()

width, height = 600, 400

```
win = pygame.display.set_mode((width, height))
clock = pygame.time.Clock()

snake = [(100, 50)]
snake_dir = (10, 0)
food = (200, 200)
score = 0

def draw():
    win.fill((0, 0, 0))
    for seg in snake:
        pygame.draw.rect(win, (0, 255, 0), (*seg, 10, 10))
    pygame.draw.rect(win, (255, 0, 0), (*food, 10, 10))
    pygame.display.flip()

running = True
while running:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            running = False
    keys = pygame.key.get_pressed()
    if keys[pygame.K_UP]: snake_dir = (0, -10)
    if keys[pygame.K_DOWN]: snake_dir = (0, 10)
    if keys[pygame.K_LEFT]: snake_dir = (-10, 0)
    if keys[pygame.K_RIGHT]: snake_dir = (10, 0)

    new_head = (snake[0][0] + snake_dir[0], snake[0][1] +
snake_dir[1])
    snake.insert(0, new_head)
    if new_head == food:
        food = (random.randrange(0, width, 10),
random.randrange(0, height, 10))
        score += 1
    else:
        snake.pop()

    if (new_head[0] < 0 or new_head[0] >= width or
        new_head[1] < 0 or new_head[1] >= height or
        new_head in snake[1:]):
        running = False

    draw()
    clock.tick(15)

pygame.quit()
```

7. Testing

Functional Testing: Verified snake movement, food generation, and score updates.

Boundary Testing: Checked wall collisions at all edges.

Stress Testing: Ensured smooth performance with long snake lengths.

8. Results

The game runs smoothly with responsive controls, accurate collision detection, and a functional scoring system. It successfully meets the project objectives.

9. Conclusion

This project demonstrates the use of Python and Pygame for developing an interactive game. It enhances understanding of event-driven programming, graphics rendering, and algorithmic problem-solving.

10. Future Enhancements

Add difficulty levels.

Include sound effects and background music.

THANK YOU!!!

SHAMBHAVI NIGAM

25BSA10111