

```
import pygame
import random
import time

# --- Pygame Initialization ---
pygame.init()

# --- Constants and Screen Setup ---
SCREEN_WIDTH = 640
SCREEN_HEIGHT = 640
FPS = 30
TILE_SIZE = 64
BOARD_SIZE = 10
NUM_TILES = BOARD_SIZE * BOARD_SIZE

# Define colors
WHITE = (255, 255, 255)
BLACK = (0, 0, 0)
GREEN = (0, 255, 0)
RED = (255, 0, 0)
BLUE = (0, 0, 255)
LIGHT_BLUE = (173, 216, 230)
ORANGE = (255, 165, 0)
PURPLE = (128, 0, 128)
YELLOW = (255, 255, 0)

screen = pygame.display.set_mode((SCREEN_WIDTH, SCREEN_HEIGHT))
pygame.display.set_caption("Snake and Ladder")
clock = pygame.time.Clock()

# --- Game Assets ---
```

```
# You can replace these with your own images later.  
  
# For now, we'll use simple colored circles for the player.  
  
font_dice = pygame.font.Font(None, 72)  
font_message = pygame.font.Font(None, 40)  
player_token_radius = TILE_SIZE // 4  
  
# --- Board Layout Calculation ---  
  
# This function calculates the (x, y) pixel coordinates for each tile (1-100)  
  
def get_tile_position(tile_number):  
    """  
    Calculates the pixel position for a given tile number (1-100).  
    The board is a zigzag pattern, starting from the bottom-left.  
    """  
  
    if tile_number < 1 or tile_number > NUM_TILES:  
        return None  
  
    # Calculate row and column  
  
    row = (tile_number - 1) // BOARD_SIZE  
    col = (tile_number - 1) % BOARD_SIZE  
  
    # Adjust column for zigzag pattern  
  
    if row % 2 == 1:  
        col = (BOARD_SIZE - 1) - col  
  
    # Calculate pixel position  
  
    x = col * TILE_SIZE + TILE_SIZE // 2  
    y = SCREEN_HEIGHT - (row * TILE_SIZE + TILE_SIZE // 2)  
  
    return (x, y)  
  
# --- Snakes and Ladders Mapping ---
```

```
# A dictionary where keys are the start and values are the end of a snake or ladder
snakes_ladders = {

    # Ladders
    4: 14,
    9: 31,
    17: 7,
    20: 38,
    28: 84,
    36: 44,
    51: 67,
    63: 81,
    71: 91,
    80: 99,

    # Snakes
    16: 6,
    48: 26,
    49: 11,
    56: 53,
    62: 19,
    64: 60,
    87: 24,
    93: 73,
    95: 75,
    98: 78
}
```

```
# --- Game State ---
player_position = 1
dice_roll_value = None
game_over = False
rolling_dice = False
```

```

# --- Game Functions ---

def draw_board():
    """Draws the 10x10 grid of the game board."""

    for row in range(BOARD_SIZE):
        for col in range(BOARD_SIZE):
            color = WHITE if (row + col) % 2 == 0 else LIGHT_BLUE
            pygame.draw.rect(screen, color, (col * TILE_SIZE, row * TILE_SIZE, TILE_SIZE, TILE_SIZE))

    # Draw tile numbers
    tile_number = 100 - (row * 10) + (9 - col) if row % 2 == 0 else 100 - (row * 10) - (9 - col)
    text_surface = pygame.font.Font(None, 24).render(str(tile_number), True, BLACK)
    text_rect = text_surface.get_rect(center=(col * TILE_SIZE + TILE_SIZE // 2, row * TILE_SIZE + TILE_SIZE // 2))
    screen.blit(text_surface, text_rect)

def draw_snakes_ladders():
    """Draws visual representations of snakes and ladders."""

    for start, end in snakes_ladders.items():
        start_pos = get_tile_position(start)
        end_pos = get_tile_position(end)

        if start_pos and end_pos:
            color = GREEN if end > start else RED
            pygame.draw.line(screen, color, start_pos, end_pos, 5)
            # Draw a small circle at the start and end of each line
            pygame.draw.circle(screen, color, start_pos, 5)
            pygame.draw.circle(screen, color, end_pos, 5)

def draw_player(position):

```

```

"""Draws the player token on the board."""
pos = get_tile_position(position)

if pos:
    pygame.draw.circle(screen, BLUE, pos, player_token_radius)
    # Draw a black border for contrast
    pygame.draw.circle(screen, BLACK, pos, player_token_radius, 2)

def draw_dice_area():
    """Draws the 'Roll Dice' button and the dice result area."""

    dice_rect = pygame.Rect(SCREEN_WIDTH // 2 - 50, SCREEN_HEIGHT // 2 - 50, 100, 100)
    pygame.draw.rect(screen, ORANGE, dice_rect, border_radius=10)

    text = "Roll"
    if dice_roll_value is not None:
        text = str(dice_roll_value)

    text_surface = font_dice.render(text, True, BLACK)
    text_rect = text_surface.get_rect(center=dice_rect.center)
    screen.blit(text_surface, text_rect)

def check_for_snakes_ladders():
    """Checks if the player landed on a snake or ladder and moves them."""

    global player_position

    if player_position in snakes_ladders:
        end_tile = snakes_ladders[player_position]

        print(f'Landed on a {"ladder" if end_tile > player_position else "snake"} from {player_position} to {end_tile}!')

        player_position = end_tile
        # Add a short delay to make the move visible
        time.sleep(0.5)

```

```

def move_player():
    """Moves the player based on the dice roll."""
    global player_position, dice_roll_value, game_over

    new_position = player_position + dice_roll_value

    if new_position > NUM_TILES:
        # Don't move if the roll is too high
        print("Roll too high, stay put.")

    elif new_position == NUM_TILES:
        player_position = new_position
        game_over = True
        print("You win!")

    else:
        player_position = new_position
        check_for_snakes_ladders()

# --- Main Game Loop ---
running = True
while running:
    # --- Event Handling ---
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            running = False
        if event.type == pygame.MOUSEBUTTONDOWN and not game_over and not rolling_dice:
            # Check if the click is on the "Roll" button
            dice_rect = pygame.Rect(SCREEN_WIDTH // 2 - 50, SCREEN_HEIGHT // 2 - 50, 100, 100)
            if dice_rect.collidepoint(event.pos):
                rolling_dice = True

    # --- Update Game State ---

```

```
if rolling_dice:
    # Simulate rolling animation
    for _ in range(5):
        dice_roll_value = random.randint(1, 6)
        draw_board()
        draw_snakes_ladders()
        draw_dice_area()
        draw_player(player_position)
        pygame.display.flip()
        time.sleep(0.1)

    move_player()
    rolling_dice = False

# --- Drawing ---
screen.fill(BLACK)
draw_board()
draw_snakes_ladders()
draw_player(player_position)
draw_dice_area()

if game_over:
    message_text = "You Win! Final Score: " + str(player_position)
    text_surface = font_message.render(message_text, True, YELLOW)
    text_rect = text_surface.get_rect(center=(SCREEN_WIDTH // 2, SCREEN_HEIGHT // 2 + 100))
    screen.blit(text_surface, text_rect)

else:
    # Show whose turn it is
    turn_text = font_message.render("Click to Roll", True, WHITE)
    turn_rect = turn_text.get_rect(center=(SCREEN_WIDTH // 2, SCREEN_HEIGHT // 2 + 100))
    screen.blit(turn_text, turn_rect)
```

```
pygame.display.flip()

# --- Frame Rate Control ---

clock.tick(FPS)

# --- Quit Pygame ---

pygame.quit()
```