Snake Game

This code implements a Snake Game using **Python's turtle library**. The game features a moving snake, food for the snake to eat, and a scoreboard that tracks the current score and the highest scoreGame Features

Screen Setup:

A 600x600 pixel screen is created with a pink background.
The title of the game is "Snake Game."

Snake Head:

The snake's head is represented by a yellow square.
It starts at the center of the screen (0, 0) and remains stationary until a key is pressed.

Food:

Food is represented by a black circle, initially positioned at (150, -200).
When the snake's head touches the food, it is repositioned randomly on the screen.

Scoreboard:

Displays the current score and the highest score in the top-left corner of the screen.
Updates dynamically as the game progresses.

Snake Movement:

The snake can move in four directions: up, down, left, and right, using the arrow keys.

Movement is achieved by updating the snake's position incrementally in 20-pixel steps.

Snake Growth:

When the snake eats the food, a new red square is added to its body.
The snake grows longer as additional food is eaten.
Collision Handling:

Wall Collision:

The snake's head wraps around the screen edges (appears on the opposite side).

Self-Collision:

If the snake's head collides with its body, the game resets:
The snake's head returns to the center.
The snake's body is cleared.
The score resets to 0.

Key Functionalities

Event Handling:

The **onkey()** method maps the arrow keys to movement functions (moveup, movedown, moveleft, moveright).
These functions ensure the snake cannot reverse direction directly (e.g., from "up" to "down").

Movement Logic:

The **move()** function updates the snake's position based on its current direction.
Snake Growth and Speed:

Each time the snake eats food:

A new square is added to the snake's body.
The score increases by 10 points.
The game speed increases by reducing the delay.

Scoreboard Updates:

The scoreboard dynamically tracks and displays the current score and the highest score.

Game Loop:

The **while True loop** continuously updates the screen and processes collisions, movement, and game mechanics.
A small delay ensures smooth gameplay.