Snake Game – Grid & Logic (Complete Guide)

This document explains the Snake game from first principles: grid structure, 1D vs 2D representation, movement logic, food, growth, scoring, and restart flow.

1. Grid Representation

The Snake game grid is visually two-dimensional but internally represented as a one-dimensional array. A 20×20 grid contains 400 indexed cells.

```
Index layout example:
0 1 2 ... 19
20 21 22 ... 39
40 41 42 ... 59
```

2. Index ↔ Coordinate Mapping

To convert between 1D index and 2D grid coordinates:

```
row = floor(index / COLS)
col = index % COLS
Reverse mapping:
index = row × COLS + col
```

3. Snake Data Structure

The snake is stored as an ordered list of cell indexes. The first index is the head, and the last index is the tail.

```
Example:
snake = [210, 209, 208]
```

4. Movement Logic

Movement is performed using index arithmetic:

```
RIGHT → index + 1
LEFT → index - 1
UP → index - COLS
DOWN → index + COLS
```

5. Collision Detection

Wall collision occurs when the snake attempts to move outside grid boundaries. Self collision occurs when the head moves into its own body.

6. Food Logic

Food is a randomly generated cell index that never overlaps the snake body.

7. Growth Mechanism

When food is eaten, the snake grows by not removing the tail during movement.

```
Normal move:
[newHead, ...snake.slice(0, -1)]
```

```
Growth move:
[newHead, ...snake]
```

8. Scoring System

Each food eaten increases the score by one.

9. Restart Logic

Restarting the game resets snake position, direction, food, score, and game-over state.

10. Game Loop Order

- Calculate new head
- Check wall collision
- Check self collision
- Check food collision
- Move or grow snake
- Update score

Conclusion

Snake is fundamentally a grid-math problem. Understanding index mapping and movement logic enables implementation of many grid-based games and algorithms.