Snake Class

Purpose

The Snake class represents the snake in the game, encapsulating its core properties and behaviors such as position, length, movement direction, and interactions with other game elements.

Attributes:

int body[MAX_LENGTH][2]:

- 2D array that stores the coordinates of each segment of the snake.
- Each row represents a segment: [x, y].

int length:

- Indicates the current length of the snake.

Direction direction:

- An enum type (UP, DOWN, LEFT, RIGHT) that represents the snake's movement direction.

Methods

Snake::Snake(int startX, int startY)

- Initializes the snake's starting position (startX, startY) with a default direction of RIGHT.
- Sets the initial length to 1.

void Snake::move()

- Updates the snake's position.
- Shifts each segment to the position of the one ahead of it, starting from the tail.
- Moves the head based on the current direction.

void Snake::grow()

- Increases the snake's length.
- Adds a new segment at the tail, duplicating the last segment's position.

void Snake::shrink()

- Reduces the snake's length.
- Removes the last segment, but the length cannot be less than 1.

Bool Snake::isCollision(int x, int y)

- Checks if the snake collides with the given coordinates (x, y).

- ✧ Returns true if a collision occurs, otherwise false.

Game Class

Purpose

The Game class acts as the central controller of the game, managing the snake, food, obstacles, and the overall game logic and state.

Attributes:

Snake snake:

- ✧ A Snake object that represents and manages the snake's behavior.

int score:

- ✧ Record the player's current score.

int redFood[6][2]

int blackFood[3][2]

- ✧ 2D arrays that store the coordinates of red and black food items, respectively.

int obstacles[4][2]:

- ✧ Stores the coordinates of obstacles.

bool isGameOver:

- ✧ Indicates whether the game has ended.

Methods

Game::Game()

- ✧ Creates the snake object.
- ✧ Sets the initial score to 0.
- ✧ Calls methods to generate red food, black food, and obstacles.
- ✧ Sets isGameOver to false.

`void Game::generateRedFood()`

`void Game::generateBlackFood()`

`void Game::generateObstacles()`

✧ Generate red and black food, obstacles at random positions.

`void Game::moveObstacles()`

✧ Random moving obstacle.

`void Game::update()`

✧ Updates the game logic:
✧ Moves the snake.
✧ Checks for collisions with boundaries, food, or obstacles.
✧ Updates the score.
✧ Moves obstacles.

`void Game::draw()`

✧ Draws the game elements:
✧ Renders the snake, food, obstacles, and the score.
✧ Displays "Game Over" if the game has ended.