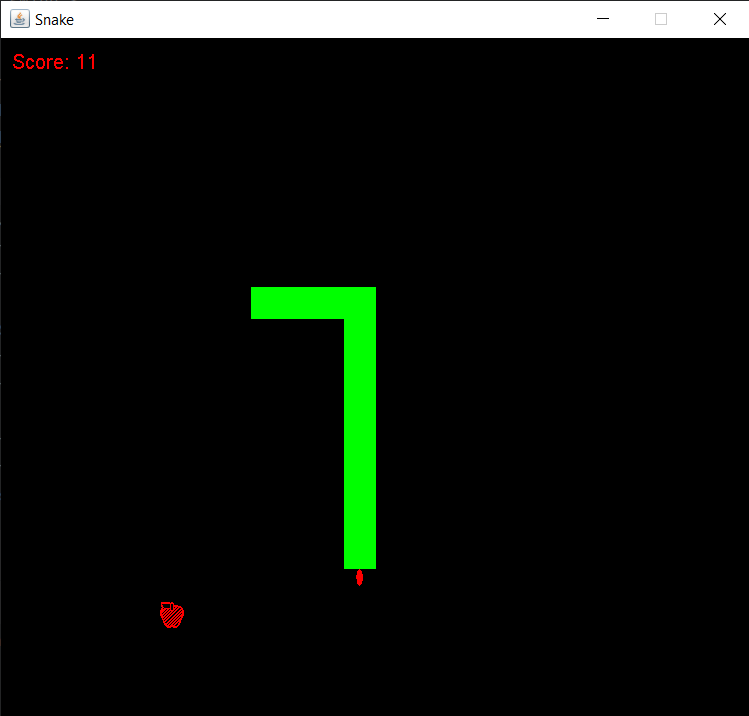
**Snake Game**

Orian Teodor Octavian

Group 30423\_2



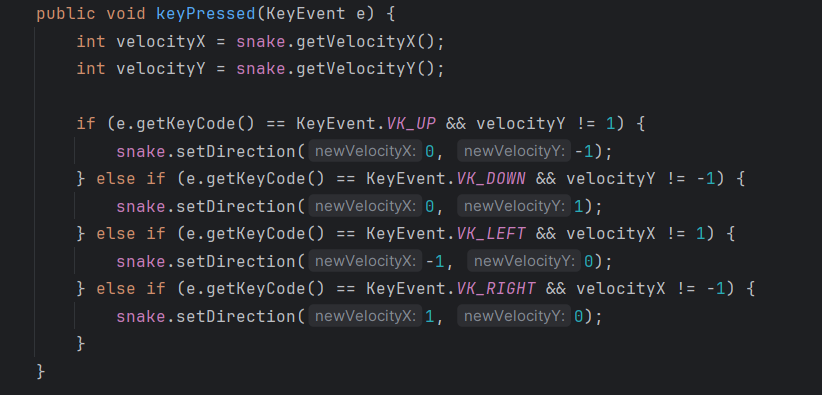
Java Swing Project

**Game description:**

The Snake game involves controlling a growing snake within a bordered area, eating food to grow longer while avoiding hitting the walls or itself. Players use controls to maneuver the snake, aiming to score points by consuming food items until they collide, ending the game.

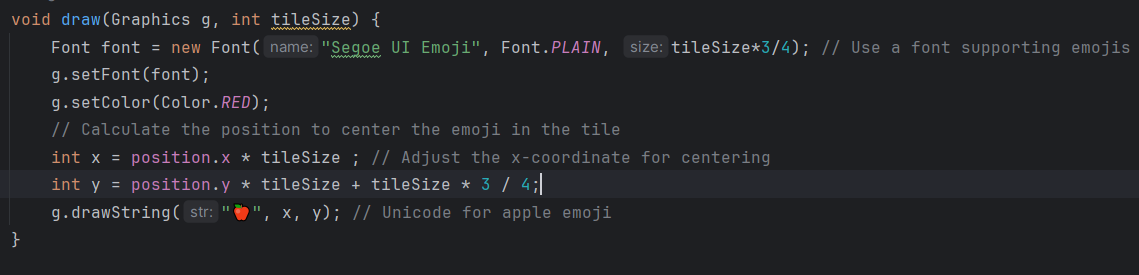
**Here's a brief description of how the game usually works:**

**1.Snake Control:** The player controls the snake using arrow keys to direct its movement.



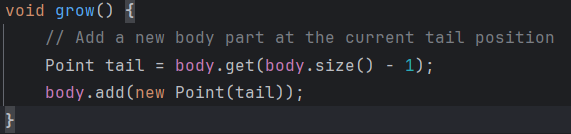
This code snippet handles keyboard input to change the snake's direction of movement. It checks the pressed key and updates the snake's direction, accordingly, preventing the snake from reversing its direction immediately to avoid self-collisions.

**2.Food Items:** Food items, often represented as dots or other symbols, appear randomly in the playing area. When the snake consumes a piece of food, it grows longer. In our case it is an apple.



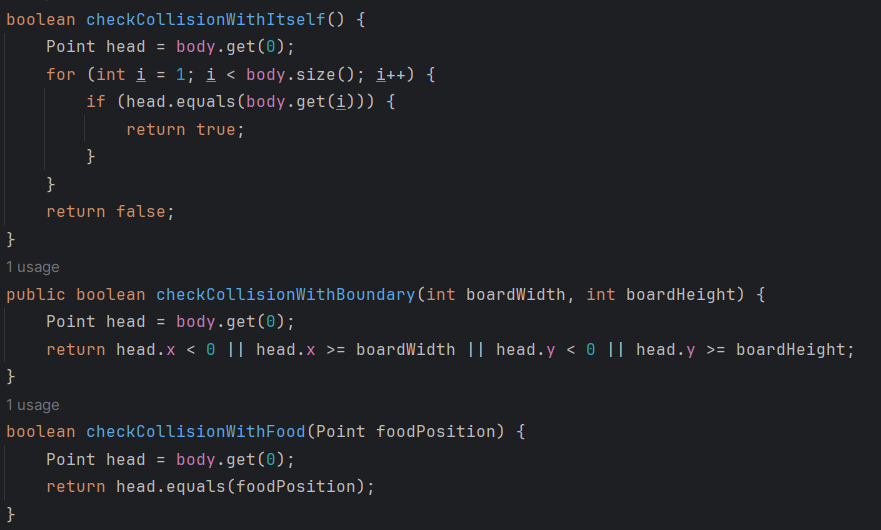
This function uses a specified graphics context (**g**) to draw a red apple emoji (**🍎**) at a calculated position on the screen based on the given position and tileSize. It ensures the emoji is centered within the tile.

**3.Growth Mechanism:** As the snake eats food, its length increases, making it more challenging to maneuver without colliding with itself or the borders of the play area.



The grow**()** function extends the snake's length by adding a new body part at the current tail position.

**4.Avoidance:** The game's challenge lies in avoiding collisions between the snake's head and its own body or the boundaries. If the snake runs into itself or the walls, it results in a game over.



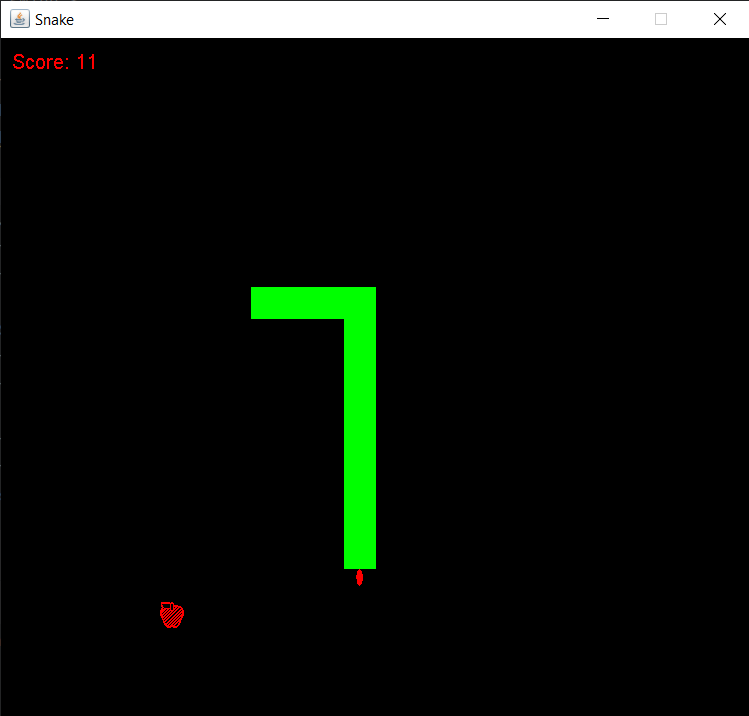
Detects if the snake's head collides with its own body parts.

Determines if the snake's head hits the boundaries of the game board.

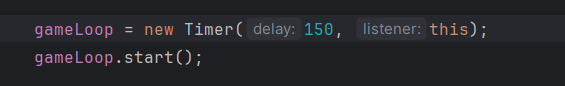
Checks if the snake's head collides with a food item on the game board.

**5.Scoring:** Players aim to achieve the highest score possible by consuming as much food as they can before ultimately ending the game due to a collision.

👇



**6.Increasing Difficulty:** Some versions of the game increase the speed of the snake as it grows longer or progressively add obstacles to increase difficulty. In our case we can modify the speed of the snake.



We can modify the delay value with the preferred speed of the snake.

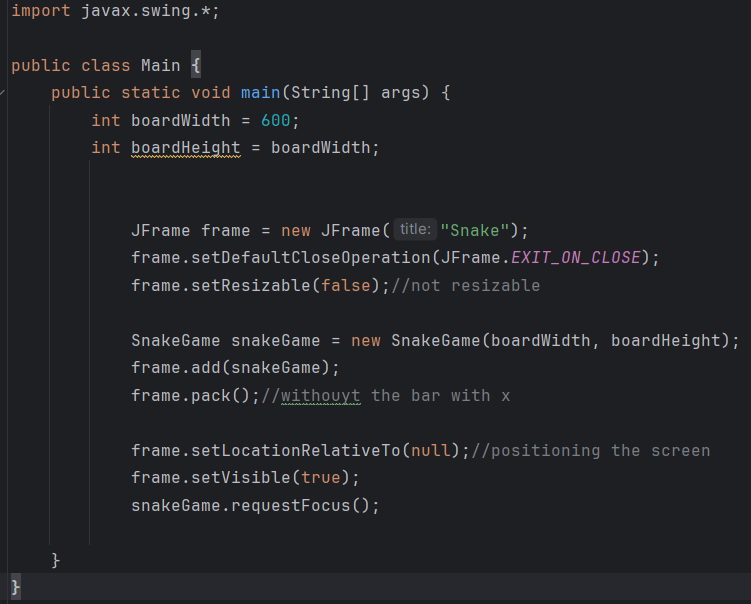
**7.Game Over:** The game ends when the snake collides with itself or the borders, displaying the player's final score. But we have a restart button so that we can play again.



**We have a total of 5 classes:**

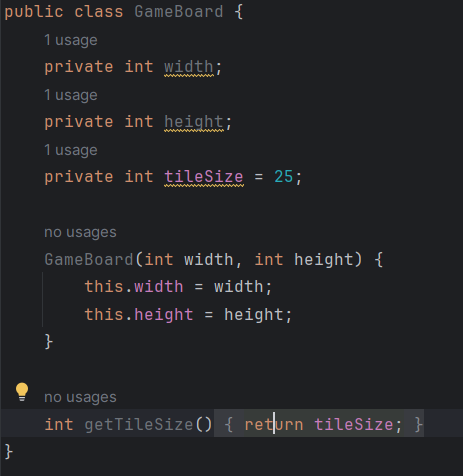
**1.Main**

This Main class sets up a non-resizable window titled "Snake" using JFrame. It initializes the SnakeGame, adds it to the window, centers the window on the screen, and makes the game window visible for user interaction.



**2.GameBoard**

The GameBoard class defines attributes for the dimensions and tile size of a game board but doesn't include specific game logic or rendering functionalities.



**3.Snake**

The Snake class manages the behavior of a game's snake entity. It includes functionalities for movement, collision detection with itself and boundaries, rendering on the game board, growth mechanics, and game over conditions.



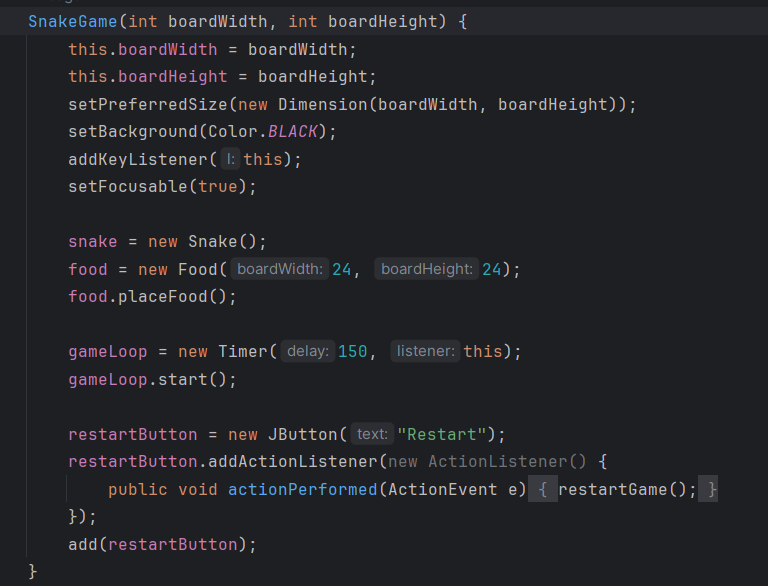
**4.Food**

The Food class manages the behavior of a food item in a game. It handles the placement and drawing of the food item on the game board while ensuring it doesn't overlap with the snake's body.



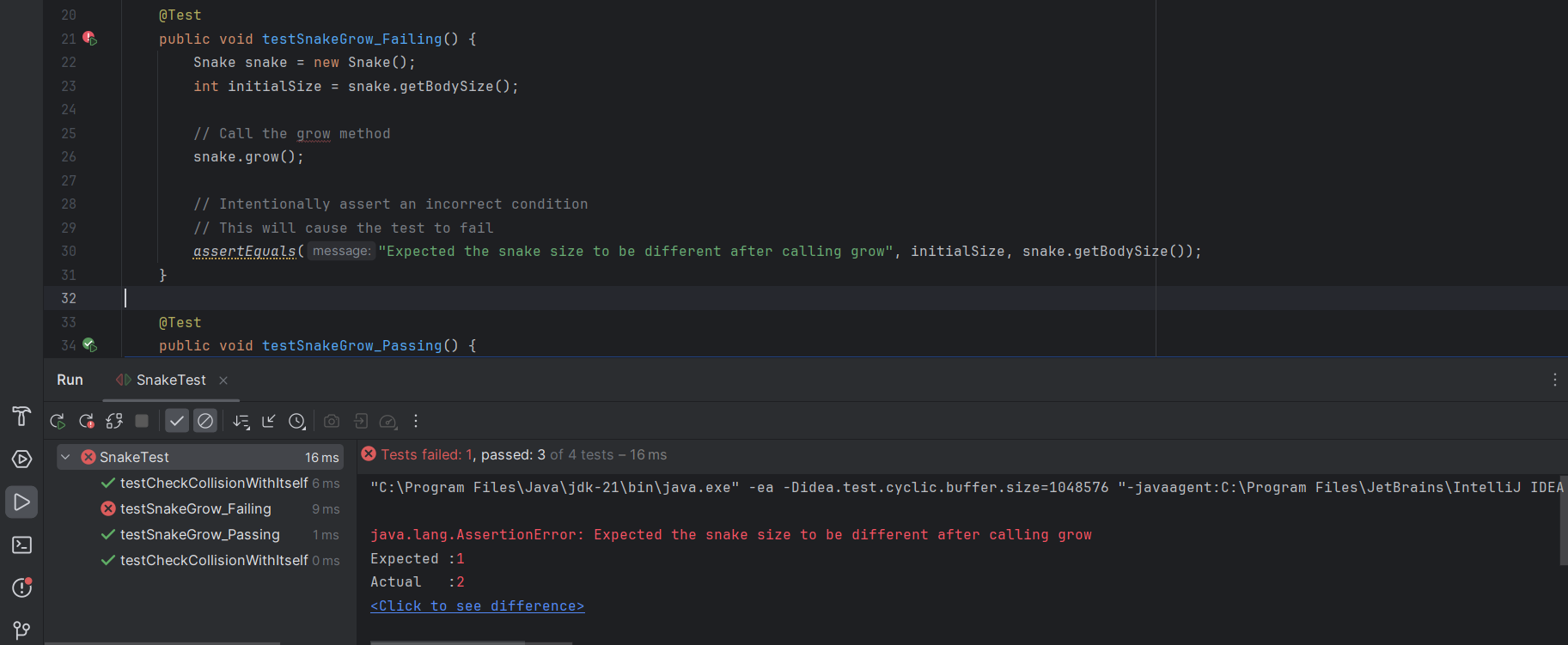
**5.SnakeGame**

The SnakeGame class manages the game logic, rendering, and user interactions for a Snake game. It handles the game loop, rendering the snake and food, checking game over conditions, and allowing the player to restart the game.

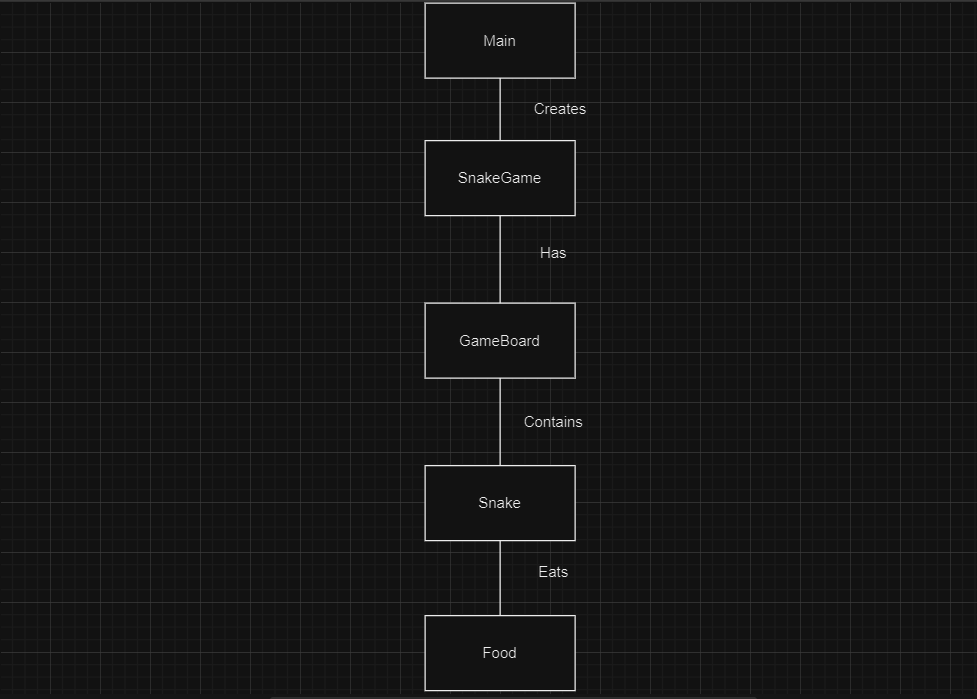


**Tests**

* testCheckCollisionWithItself**\_**NoCollisionMethod**:**
  + **Objective:** This test checks the checkCollisionWithItself method of the Snake class when there is no collision.
  + **Description:**
    - It creates an instance of the Snake class.
    - It asserts that the checkCollisionWithItself method returns false when there is no collision with the snake's own body.
* testCheckCollisionWithItself\_Collision Method:
  + **Objective:** This test checks the checkCollisionWithItself method of the **Snake** class when there is a collision.
  + **Description:**
    - It creates an instance of the Snake class.
    - It grows the snake by adding a body part using the grow method.
    - It asserts that the checkCollisionWithItself method returns true when there is a collision with the snake's own body.
* testSnakeGrow**\_**FailingMethod**:**
  + **Objective:** This test checks the grow method of the Snake class with an intentional failure.
  + **Description:**
    - It creates an instance of the Snake class.
    - It records the initial size of the snake using the getBodySize method.
    - It calls the grow method to increase the size of the snake.
    - It intentionally asserts an incorrect condition (expecting the size not to change), causing the test to fail.
* testSnakeGrow**\_**PassingMethod**:**
  + **Objective:** This test checks the grow method of the Snake class with the correct behavior.
  + **Description:**
    - It creates an instance of the Snake class.
    - It records the initial size of the snake using the getBodySize method.
    - It calls the grow method to increase the size of the snake.
    - It asserts that the size of the snake has increased by checking the getBodySize method.



**Class Diagram**



**Bibliography**

[**https://www.javatpoint.com/java-swing**](https://www.javatpoint.com/java-swing)

[**https://htmlcolorcodes.com/**](https://htmlcolorcodes.com/)

[**https://coderanch.com/t/633503/java/Tiled-Based-Map-Game-Java**](https://coderanch.com/t/633503/java/Tiled-Based-Map-Game-Java)

[**https://stackoverflow.com/questions/21879243/how-to-create-on-click-event-for-buttons-in-swing**](https://stackoverflow.com/questions/21879243/how-to-create-on-click-event-for-buttons-in-swing)

[**https://stackoverflow.com/questions/22386477/how-does-addnotify-and-requestfocus-work-in-java-with-jpanel**](https://stackoverflow.com/questions/22386477/how-does-addnotify-and-requestfocus-work-in-java-with-jpanel)

[**https://chat.openai.com/**](https://chat.openai.com/)