Final Project: Snake Xenzia

Repository Link: https://github.com/jsvoo/csci5003-project-snake-xenzia.git

Possible Future Implementations

1. Sound Effects Integration

Description:
Add sound effects to enhance the game's immersion:

- **Meal Appearance/Disappearance:** Play a sound when meals appear, disappear, or are consumed by the snake.
- **Snake Death:** Play a specific sound when the snake dies, adding an emotional impact to the event.

Design Accommodation:

- A SoundManager class can be introduced to manage all sound effects in the game.
- The applyEffect methods in the meal classes can trigger the relevant sounds when meals are consumed.
- The Snake class can trigger the death sound when the game-over condition is met. \

2. Time-Limited Power-Ups

Description:
Introduce time-limited power-ups that grant the snake special abilities for a short duration. Examples include:

- **Speed Boost:** Temporarily increases the snake's movement speed.
- **Shield:** Grants temporary immunity from collisions with walls or its own body.
- **Reverse Controls:** Reverses the snake's controls for a short time, adding a layer of difficulty.

Design Accommodation:

- The ApplyEffect interface can be extended to include a duration parameter, allowing effects to be time-bound.
- The game timer can handle the activation and expiration of these power-ups. When a power-up is consumed, the game could start a countdown, after which the effect is reversed or removed.

3. New Meal Types

Description:
Introduce new meal types with unique effects:

- **Teleport Meal:** Teleports the snake to a random position on the board.
- **Shrink Meal:** Temporarily reduces the size of the snake, making it easier to navigate tight spaces.

Design Accommodation:

- The existing meal classes (e.g., BonusMeal, DangerMeal, HealthPack) already implement the ApplyEffect interface. New meal types would follow the same pattern, with specific logic in their applyEffect methods.
- The MealFactory class can be expanded to randomly generate these new meal types during gameplay.

4. Multi-Player Mode

Description:
Add a competitive multi-player mode where two snakes compete for meals on the same board. The game could end when one snake collides with the other, or the player with the highest score after a time limit wins.

Design Accommodation:

- The Snake class can be extended to manage multiple snakes. Each player's snake would be managed independently with its own controls, direction, and collision checks.
- The GamePanelGUI class would be updated to handle multiple snakes' drawing, movement, and collisions.

5. Advanced AI Opponent

Description:
Introduce an AI-controlled snake that competes against the player. The AI would try to collect meals while avoiding collisions, adding a challenging single-player experience.

Design Accommodation:

- A new AISnake class can inherit from the Snake class but with an AI algorithm to determine movement and decisions.
- The AI logic can use pathfinding algorithms like A* or simple heuristics to navigate the board.