Keyboard Input

Due Date: 11:59 PM, September 18, 2019

Objectives

This assignment demonstrates how to:

- Use a timer to produce animation.
- Incorporate images and sounds in a program.
- Generate and use pseudo-random numbers.

You will add the ability to accept keyboard input in a Java graphical environment.

Background

Snake is a popular updated version of the classic computer game Nibbler. It became popular in the late 1990s when it was included free on cell phones. In the game, the user uses (arrow) keys to change the direction of the snakes movement in order to guide the snake around obstacles, directing it toward food objects on the screen. The game is over if the snake collides with either the walls of the playing area, one of the barriers or any part of the snakes body. Once the snakes head reaches a food object, the object is consumed and the snakes body grows longer by some specified length (one segment).

Applicaton

In this assignment, you will add the code to support keyboard input to a simplified version of this game. Many free, online versions of the game exist and can be found via internet search for your reference. The snake will start with some initial length (3 segments long) and move in some initial direction until the user presses an arrow key to direct the snake to turn either left or right 90 degrees. A timer is used to animate the snake. Some features of this game include:

- Pressing an arrow key to a direction corresponding to the direction of movement of the head of the snake (either forward or backward) should not have any effect on the snake. Use a `KeyListener` and the defined constants `KeyEvent.VK_UP`, `KeyEvent.VK_DOWN`, `KeyEvent.VK_LEFT` and `KeyEvent.VK_RIGHT`

- If the snake attempts to move outside the bounds on the drawing panel of the game, it loops to the opposite side of the screen and continue its movement in the same direction.
- This version does not use any barriers.
- When a food item is consumed, an appropriate sound will play (e.g. munching or chomping).
- Once a food item is consumed, the snake grows by one body segment.
- You do not need to check for collisions between the head of the snake and its body segments. If the snake overlaps itself, game play continues. The game ends when the user gets bored and closes the application.
- The food item is placed at a location determined by obtaining x and y coordinates using the pseudo-random number generator. Each new food item is placed in a location corresponding to the possible x and y coordinates of the snake to simplify the detection of when a food item is consumed.

Your Task

In order to get some experience with accepting keyboard input for upcoming project deliverables, you will complete the code necessary to respond to all four arrow keys on the keyboard. The head of the snake implements the `Mover` interface. So, it responds to the `moveUp()`, `moveDown()`, `moveLeft()`, and `moveRight()` messages. The `KeyInteractor` class was included with the code for this assignment. A private inner class that extends the `KeyInteractor` class has been created to respond appropriately to the up arrow key. You must add similar classes to respond to the left, right, and down keys.

Grading

You are free to get whatever help you need in lab class to complete the assignment correctly. But, you must type in the code yourself. If you are unable to turn the work in during the lab class, you must complete it and submit it by 11:59PM that evening.