EECS 2030 Project:  
  
Snake World

Jason Skinner, Alessa Ivascu, Dawood Choksi, Leroy Nguyen, Kristiana Papajani

Description of Code

* The snake game project consists of many classes that are blueprints for its objects.
* The project consist of the main classes are MVC (**Model**.java, **Controller.**java**,** and **View**.java), which is a software architectural pattern for implementing user interfaces on computers.
* There are other classes that take care the objects of the game which are: **AudioFilePlayer.java**, **HS.java**, **RowCol.java**, and **Snake.java**. Lastly, in order to run and execute the snake game, requires a PSVM which is found in Game.java. These classes all adhere to the MVC which embodies the objects of the Snake Game.
* The **Model** separates the logic of a program from the rest of the user interface. It manages the data and fundamental behaviours of a program. It is the data and a data-management portion of the program.
  + In our Model class, we have *constant* fields of numbers and strings which are public, private, static, and final.
  + We have an empty Mode class constructor which adheres to the MVC pattern.
  + A showHighScores() is constructed, where the Model retrieves the high scores from the HS class and then displays it them into in a pop-up message dialog window.
    - In order to implement the method, requires a for loop technique.
  + The game includes a multiplayer mode, where it introduces a second player on the left-hand side of the Graphic User Interface (GUI), whereas the first player is on the right-side. Thus the new game is set up for both players.
    - The newTwoPlayerGame() method is responsible for this.
  + The model takes care of the image of the snake as it moves across the game board for both single and multiplayer, as it moves in the game board or background of the GUI. For example, as the snake head moves in a direction forward, the body of the snake follows the previous position of what the snake head was just in. There are two types of images for the snake’s body and four types for the snakes head that must be considered depending on the DIRECTION (up, down, right, left) of the snake’s movement.
    - Thus, changeHeadToBody(Snake snake) is implemented for this behaviour.
    - The method showSnakeBody(Snake snake) takes care of the movement of the snake’s body as it moves around at any direction.
* At the very start of the game after running Game.java, the player will be immediately prompted and be asked to optionally if he/she wants to play the background music, and as well as choosing their player's usernames. Then after, the game refreshes and sets up for the player(s).
* This is evident in setView(View view) method.

* + When the snake dies in the game, the game immediately stops and resets when the user wants to try and play again.
    - So the Model calls the showSnakeGone(Snake snake) and erases the snake in the game.
  + So depending on which direction of the snake is moving, it will change it to either left, right, upward or downward direction.
    - The methods: leftPressed(String snake), rightPressed(String snake), upPressed(String snake), and downPressed(String snake) is manages the movements of the player(s) snakes.

* In the Snake class, we have used an encapsulation where we have all our fields/attributes in **private** (i.e; private int length, private RowCol head, private ArrayList<RowCol>…etc.) and getters which are being accessed by the Model class.