Introduction: Game

This project is an introduction to two dimensional game design using an animation timer and an event handler processing up and down keys. Although the structure of the software is simplistic it reflects the basic concepts of designing two dimensional games. Games such as Space Invaders which was released in 1978 and Pac Man released in 1980 can be implemented using the same methodology and design structure. The gaming industry has come a long way since then, and has become a multi-billion dollar industry.

A modern game like World of Warcraft required a couple of hundred engineers to create the game. The development time was approximately 5 years. The game is built using its own unique 3D engine and terrain mapping, as well as character development, and story writing. My point is that students studying game design need to manage their expectations, work in teams, and collaborate with various talents needed.

Key Design Points

- The snake is comprised of 4 circles, c1, c2, c3, and c4. Circle c1 uses a different color than the other 3 circles and it is designated as the head of the snake. When adding the four circles to the pane, c1 should be the last circle added, since the last widget added to the pane is automatically displayed on top.

  ```
  pane.getChildren().addAll(c4, c3, c2, c1);
  ```

- The screen coordinates of the pane is (0, 0) in the top left corner and (400, 400) or (n, n) in the bottom right corner. This requires the y coordinate to decrease when moving the snake to the top of the screen.
- The class MyTimer is implemented as an inner class and it extends the AnimationTimer. The AnimationTimer is an abstract class and we cannot instantiate an object directly of an abstract class. When creating the class MyTimer we need to be careful not to insert it into the body of the method start(). That is a common mistake that students sometimes make.
- The class MyTimer controls the automated movement of the enemy target (e1). The position coordinates of the enemy target are controlled using a random generator. Students should explore a more elaborate method to move the target.
- There is no boundary checking implemented in this project. Boundary checking should be implemented for the target, since the target could run out of the pane area. Once the target is outside the area it is not visible.
- The timing of events can be changed by adjusting the conditional statement. “50” should not be hardcoded in the conditional statement, instead a control variable should have been used.

  ```
  lastUpdate++;
  if (lastUpdate >= 50) { …
  ```

Instructions:

Create a project “Snake” using ECLIPSE or equivalent and add a class called Main. Delete all content in the class Main and copy the code below into the file.

The up, down, left, and right key controls the snake movement. The snake is comprised of 4 circles and the head circle is in the color red. After the Start button is pressed an enemy target appears and starts to move. The enemy target is a slightly larger circle in the color purple. The movements of the enemy target are completely random, and it is possible for the target to move out of sight since there is no border checking in place. That kind of detail is left up to students to implement. Move the Snake until the head touches the enemy object. The enemy object disappears when the head of the snake overlaps and the player earns 10 points. Shortly afterwards a new enemy target appears. The game can be stopped by pressing the Stop button. After the game is stopped the Reset button can be pressed which resets the game returning the Snake to its starting position. The Start button must be pressed to start a new game.

Code:

```
import javafx.application.Application;
import javafx.animation.AnimationTimer;
import javafx.event.*;
import javafx.geometry.Insets;
import javafx.scene.Scene;
import javafx.scene.control.*;
import javafx.scene.input.KeyEvent;
import javafx.scene.layout.*;
import javafx.scene.paint.Color;
import javafx.scene.shape.Circle;
import javafx.stage.Stage;

import java.util.Random;

public class Main extends Application {

        public Pane pane;
        public Circle c1;
        public Circle c2;
        public Circle c3;
        public Circle c4;
        public Circle e1;
        public Label lbl;

        Random rnd;
        int dist;
        int score;
        boolean enemyFlag;
```

```
public static void main(String[] args)
{
    launch(args);
}

public void start(Stage stage)
{
    stage.setTitle("Snake");
    AnimationTimer t = new MyTimer();
    rnd = new Random();

    dist = 18;
    score = 0;
    enemyFlag = false;

    Color strokeColor = Color.rgb(190, 190, 190);
    Color snakeBodyColor = Color.rgb(100, 100, 200);

    c1 = new Circle(200, 350, 7);
    c1.setStroke(strokeColor);
    c1.setFill(Color.rgb(220, 100, 100));

    c2 = new Circle(200, 368, 7);
    c2.setStroke(strokeColor);
    c2.setFill(snakeBodyColor);

    c3 = new Circle(200, 386, 7);
    c3.setStroke(strokeColor);
    c3.setFill(snakeBodyColor);

    c4 = new Circle(200, 404, 7);
    c4.setStroke(strokeColor);
    c4.setFill(snakeBodyColor);

    e1 = new Circle(200, 404, 10);
    e1.setStroke(strokeColor);
    e1.setFill(Color.rgb(200, 100, 200));

    lbl = new Label("Score = 0");

    //*********************************************************
    // Create buttons
    //*********************************************************
    Button StartBtn = new Button("Start");
    StartBtn.setOnAction((ActionEvent e) -> {
```

```
    t.start();
});

Button StopBtn = new Button("Stop");
StopBtn.setOnAction((ActionEvent e) -> {
    t.stop();
});

Button ResetBtn = new Button("Reset");
ResetBtn.setOnAction((ActionEvent e) -> {
    c1.setCenterX(200);
    c1.setCenterY(350);
    c2.setCenterX(200);
    c2.setCenterY(368);
    c3.setCenterX(200);
    c3.setCenterY(386);
    c4.setCenterX(200);
    c4.setCenterY(404);

    enemyFlag = false;
    score  = 0;
    lbl.setText("score = 0");
    removeEnemy();
});


//*******************************************************
// Create pane, hbox, vbox, scene
//*******************************************************
pane = new Pane();
pane.setMinSize(450, 450);
removeEnemy();

HBox HB = new HBox();
HB.setPadding(new Insets(10,10,10,10));
HB.setSpacing(7);
HB.getChildren().addAll(StartBtn, StopBtn, ResetBtn);

VBox VB = new VBox();
VB.setPadding(new Insets(10,10,10,10));
VB.setSpacing(7);
VB.getChildren().addAll(pane, HB, lbl);

Scene scene = new Scene(VB);
stage.setScene(scene);
stage.show();
```

```
//*********************************************************
// Processing Key events
//*********************************************************
scene.setOnKeyPressed(new EventHandler<KeyEvent>()
{
    @Override
    public void handle(KeyEvent event)
    {
            switch (event.getCode())
            {
            case UP:        connectSnakeBody();
                            c1.setCenterY(c1.getCenterY() - 18);
                             break;
            case DOWN:   connectSnakeBody();
                            c1.setCenterY(c1.getCenterY() + 18);
                            break;
            case RIGHT:  connectSnakeBody();
                            c1.setCenterX(c1.getCenterX() + 18);
                            break;
            case LEFT:    connectSnakeBody();
                            c1.setCenterX(c1.getCenterX() - 18);
                            break;
            defaul          // No default
                            break;
            } // end switch

            checkProximity();
        } // end handle
   }); // end Onkeyressed
} // end start

//*********************************************************
// Membership Functions
//*********************************************************
public void removeEnemy()
{
   pane.getChildren().clear();
   pane.getChildren().addAll(c4, c3, c2, c1);
}

public void connectSnakeBody()
{
   c4.setCenterY(c3.getCenterY());
   c4.setCenterX(c3.getCenterX());
   c3.setCenterY(c2.getCenterY());
```

```
    c3.setCenterX(c2.getCenterX());
    c2.setCenterY(c1.getCenterY());
    c2.setCenterX(c1.getCenterX());
}

public void checkProximity()
{
    int killzoneX = (int) (c1.getCenterX() - e1.getCenterX());
    int killzoneY = (int) (c1.getCenterY() - e1.getCenterY());
    killzoneX = Math.abs(killzoneX);
    killzoneY = Math.abs(killzoneY);

    if ( killzoneX <= 10 && killzoneY <= 10)
    {  // enemy is killed
        removeEnemy();
        score += 10;
        lbl.setText("score = " + score) ;
        enemyFlag = false;
    }
} // End checkProximity


//*********************************************************
// Inner Class MyTimer
//*********************************************************
public class MyTimer extends AnimationTimer
{
    long lastUpdate = 0;

    public void handle()
    {
    }

    @Override
    public void handle(long now)
    {
        lastUpdate++;
        if (lastUpdate >= 50)
        {
            lastUpdate = 0;
            if (enemyFlag == false)
            {
                e1.setCenterX(rnd.nextInt(300));
                e1.setCenterY(rnd.nextInt(300));
                pane.getChildren().add(e1);
                enemyFlag = true;
```

```
                    }
                    else {
                        int x = rnd.nextInt(10);
                        int y = rnd.nextInt(10);
                        if (x < 5) {
                            e1.setCenterX(e1.getCenterX() - 10);
                        }
                        else {
                            e1.setCenterX(e1.getCenterX() + 10);
                        }

                        if (y < 5) {
                            e1.setCenterY(e1.getCenterY() - 10);
                        }
                        else {
                            e1.setCenterY(e1.getCenterY() + 10);
                        }
                    }

                    checkProximity();
                }
            } // end handle
        } // End Timer

} // End Main
```