**Description of Code**

**Student name:** Yifan Feng

**Student QMUL email:** [y.feng@se20.qmul.ac.uk](mailto:y.feng@se20.qmul.ac.uk)

**Date:** 4 Oct 2020

**General Purpose**

The four sketches together aim to reproduce a simplified video game: “Snake Game”. The main game concept is that players steer a three-unit long line which grows in length after gaining a bonus point (resembling an apple). The player will lose when the line (representing a snake) runs into the screen borderlines or itself. BestScore will be calculated and shown on the game board after the game goes to an end.

**How the Sketch Works**

The reproducing work is inspired by several external tutorial videos based on Java and JavaScript language. The main resources are:

* <https://thecodingtrain.com/CodingChallenges/003-snake-game-p5.html>
* <https://www.bilibili.com/video/BV1YE411T7dG>
* <https://www.learnjavacoding.com/definitions/snake/>

*Sketch\_step1:*

Sketch one aims to set up basic information regarding background color, snake color and (maximum) length, moving speed and original direction and build up the snake itself (body, head and size). I found that each movement has x and y axis change. Therefore, I created variables *snake\_head\_x* and *snake\_body\_y* to store each grid (a body part) coordinates and every positional update. The variables here are designed for *for loop* to iterate three section of snake (head, body1 and body2). As for body length variations, I created two array list named x and y to record snake length regarding x and y axis change and these two array lists help to locate each iteration’s new position in the canvas. Moreover, basic settings like color palate and canvas/screen size were referred to the given snake game website.

*Sketch\_step2:*

Sketch two built upon the first one that I added a direction control. By default, snake runs from left to right. Here I used *keypressed()* and system-defined keycode for switching four directions. I used if / else if statements to discuss four possible running routes. In this sketch, the movement is controlled by head coordinates change and default running speed. For example, if the snake goes up, the x position will remain the same, but y position needs to be updated by detracting moving speed.

*Sketch\_step3:*

Step three discusses collision and game end and I thereby created a Boolean value *game\_over*. The function Boolean *game\_over* aims to show the final score after Boolean *snake\_die* returns true. *Snake\_die* verifies four conditions that can lead to the snake’s death and basic concept is that the game ends when snake runs out of four screen borderlines by comparing snake’s head coordinates and boundary values.

*Sketch\_step4:*

Sketch four includes the point calculation and making random food(apple). The main variables are first the *apple’s x and y coordinates* due to its random occurrence feature; and second the Boolean values of apple occurrence location (*apple\_given and apple\_outbody*). The Boolean values help to rule out the problem that apple can be drawn on snake’s head and body (clashed coordinates/positions). Basic logics are by not giving apple, apple-not-on-body is true, creating new apple should be functional. By giving apple, Boolean will verify on / off body statements and snake gets one unit longer if snake head overlaps with apple’s position.

**Reflections and Improvements**

This is my first Java programming attempt. Generally, I spent more than 20 hours in coding and debugging (not even including code organizing and report writing) and found it stressful to draw my thoughts on Java. To program this game, I realized how important to construct self-defined functions like apple\_appearence and snake\_body\_change. Most importantly, I noticed variables such as head\_x and head\_y and array list to save dynamic change. I also found that a good schedule is needed for organizing codes and understanding for example how to coordinate snake’s head and body movement in x and y axis.

As for improvements, I should discuss a condition where snake can run into itself. Also, I failed to make three modes of speed (slow, moderate and fast). I found my logic not clear when discussing snake and apple’s clash (*apple\_appearenc*e) and snake body movement (*snake\_body\_change*) and resulting in unorganized code snippets. I should write separate coordinate change rather than for loop for better understanding. Additionally, I think it would be nicer if the play function is more complete by adding pause, restart and more direction control keys. Some music elements also should be applied to make the game more interactive and appealing.