Credit Task 6.4 - Custom Program Design

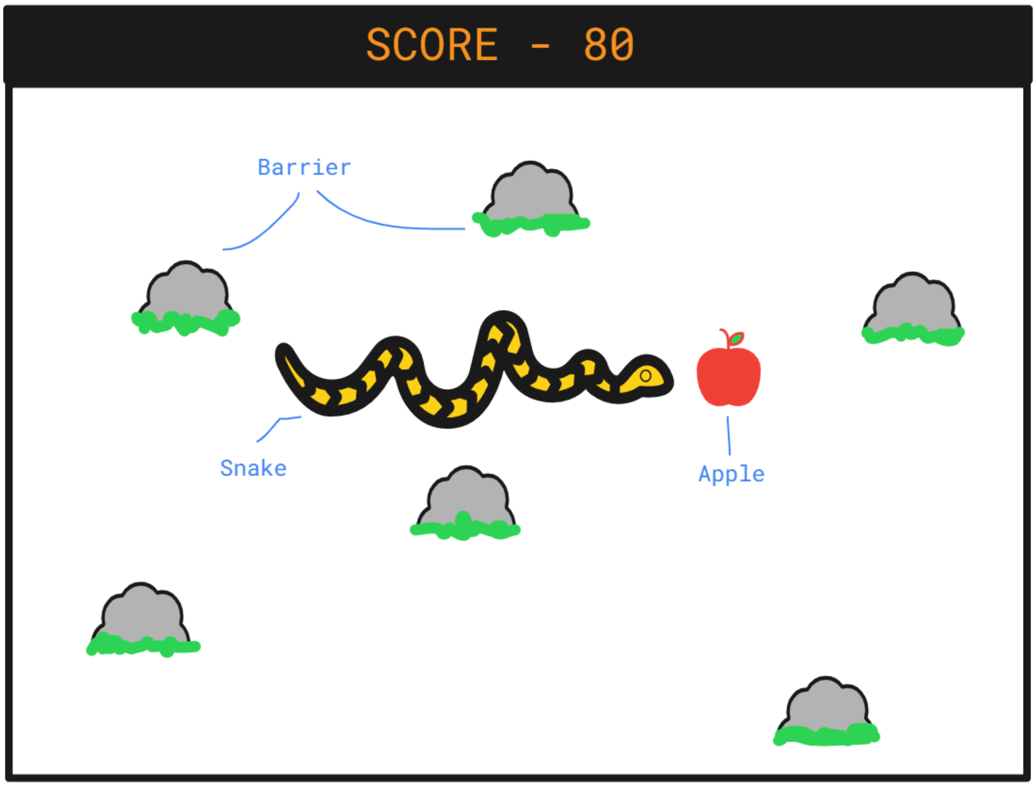
Object-Oriented Programming (Swinburne University of Technology)

Design Overview for << Snake game >>

Name: Senuri Tharushika Wickramarathne
Student ID: 102863056

Summary of Program

This program is similar to the snake game, where you get points when the snake eats the apples that appear on the screen. Each time the snake eats an apple the player gets 10 points. The snake will also get longer each time it eats an apple. After reaching a score of 100 the positions of the barriers will change to make the game a bit more complex. If the snake knocks on itself, on a barrier or the borders of the screen, then the player will lose the game. When the game ends the screen will show the points received by the player.



Required Data Types

Table 1: <<record name>> details

Field Name	Type	Notes
xpos	Integer	X position of the snake, apple and barrier
ypos	Integer	Y position of the snake, apple and barrier
speed	Integer	Movement speed of the snake
length	Integer	Length of the snake
ticker	Boolean	Determines if the snake should grow or not
score	Integer	Score of the player

Table 2: <<enumeration name>> details

Value	Notes
head_segment	The head segment of the snake
snake_face	Contains the image for the snake face
snake_back	Contains the image for the snake tail
direction	The direction in which the snake moves (right, left, up and down)
extend	The extending part of the snake each time it gets longer
block_move	Provides the direction to the body of the snake considering the X and Y values of the previous point.
apple	Apple eaten by the snake
barrier	The barriers the snake should avoid from hitting

…

Overview of Program Structure

First in the game window the snake, apple, barrier and score is initialized. Then you write a procedure to change the snake direction through the left, right, up and down keys and write another procedure to increase the score and the length of the snake once it eats an apple. Here you also write a procedure to make sure that the apple coordinates and the barrier coordinates differ, to make sure they do not overlap. Then four classes are created for the snake, segment, apple and the barrier. For the snake class first the direction of the snake, x position, y position, length, speed and the segments of the snake are initialized. Then the snake face and snake back is drawn and a procedure is written for the movement of them. Then you write the procedures to check if the snake hits itself, hits on a barrier or goes outside bounds. For the segment class you first initialize the x and y positions of the snake and then draw the extending part of the snake which needs to get longer when the snake eats an apple. For the apple class you first initialize the x and y positions of the apple and then draw the apple. For the barrier class you first initialize the x and y positions of the barrier and then draw the barrier.