Course: 113-1 Embedded System Answer Sheet

Assignment 8: SenseHat Game

|  |
| --- |
| **Practice 1: Source code (Please DO NOT provide a screenshot)** |
| from sense\_hat import SenseHat  import time  import random  sense = SenseHat()  # 顏色定義  GREEN = (0, 255, 0) # 蛇  RED = (255, 0, 0) # 食物  BLACK = (0, 0, 0) # 背景  # 初始化  snake = [(3, 3)] # 蛇的起始位置  food = (random.randint(0, 7), random.randint(0, 7)) # 隨機生成食物  direction = "UP" # 初始方向  start\_time = time.time() # 開始時間  # 確保食物不生成在蛇的身體上  def generate\_food():  while True:  new\_food = (random.randint(0, 7), random.randint(0, 7))  if new\_food not in snake:  return new\_food  # 顯示遊戲畫面  def draw():  sense.clear()  for segment in snake:  sense.set\_pixel(segment[0], segment[1], GREEN) # 畫蛇  sense.set\_pixel(food[0], food[1], RED) # 畫食物  # 更新蛇的位置  def move():  global snake, food, direction  head\_x, head\_y = snake[0]  # 根據方向移動蛇頭  if direction == "UP":  head\_y -= 1  elif direction == "DOWN":  head\_y += 1  elif direction == "LEFT":  head\_x -= 1  elif direction == "RIGHT":  head\_x += 1  # 撞牆或自我碰撞時遊戲結束  if head\_x < 0 or head\_x > 7 or head\_y < 0 or head\_y > 7 or (head\_x, head\_y) in snake:  end\_time = time.time()  survival\_time = int(end\_time - start\_time)  final\_score = len(snake) \* survival\_time  sense.show\_message("Game Over! Score: {}".format(final\_score), text\_colour=RED)  quit()  # 更新蛇頭  new\_head = (head\_x, head\_y)  # 吃到食物  if new\_head == food:  snake.insert(0, new\_head) # 增加蛇長度  food = generate\_food() # 生成新食物  else:  snake.insert(0, new\_head)  snake.pop() # 移除尾部  # 改變方向  def change\_direction(event):  global direction  if event.action == "pressed":  if event.direction == "up" and direction != "DOWN":  direction = "UP"  elif event.direction == "down" and direction != "UP":  direction = "DOWN"  elif event.direction == "left" and direction != "RIGHT":  direction = "LEFT"  elif event.direction == "right" and direction != "LEFT":  direction = "RIGHT"  # 綁定搖桿事件  sense.stick.direction\_up = change\_direction  sense.stick.direction\_down = change\_direction  sense.stick.direction\_left = change\_direction  sense.stick.direction\_right = change\_direction  # 主遊戲循環  while True:  draw()  move()  time.sleep(0.5) |

**Student No: 113598043**

**Student Name: 張育丞**

|  |
| --- |
| **Practice 1: Embed the video link (YouTube) please refer to the assignment document.** |
| [HW08-Practice 1](https://www.youtube.com/embed/zMUKKbU-KRk?feature=oembed) |

|  |
| --- |
| **Practice 2: Source code (Please DO NOT provide a screenshot)** |
| from sense\_hat import SenseHat  import time  sense = SenseHat()  sense.clear()  # Define colors  WHITE = (255, 255, 255)  GREEN = (0, 255, 0)  YELLOW = (255, 255, 0)  PURPLE = (128, 0, 128)  BLACK = (0, 0, 0)  # Ball starting position and velocity  ball\_pos = [4, 5]  ball\_vel = [1, -1]  # Paddle position  paddle\_x = 3  # Brick layout  bricks = {}  for x in range(8):  bricks[(x, 0)] = 'yellow' # Initial brick state  def draw\_bricks():  for position, state in bricks.items():  color = YELLOW if state == 'yellow' else PURPLE  sense.set\_pixel(position[0], position[1], color)  def draw\_paddle():  for x in range(paddle\_x, paddle\_x + 3):  sense.set\_pixel(x % 8, 7, GREEN)  def draw\_ball():  sense.set\_pixel(ball\_pos[0], ball\_pos[1], WHITE)  def handle\_events():  global paddle\_x  for event in sense.stick.get\_events():  if event.action == 'pressed':  if event.direction == 'left':  paddle\_x = (paddle\_x - 1) % 8  elif event.direction == 'right':  paddle\_x = (paddle\_x + 1) % 8  game\_over = False  while not game\_over:  handle\_events()  # Clear previous positions  sense.clear()  # Update paddle position  draw\_paddle()  # Update ball position  ball\_pos[0] += ball\_vel[0]  ball\_pos[1] += ball\_vel[1]  # Check for wall collisions (X-axis)  if ball\_pos[0] < 0:  ball\_pos[0] = 0  ball\_vel[0] \*= -1  elif ball\_pos[0] > 7:  ball\_pos[0] = 7  ball\_vel[0] \*= -1  # Check for wall collisions (Y-axis)  if ball\_pos[1] < 0:  ball\_pos[1] = 0  ball\_vel[1] \*= -1  # Check for paddle collision  if ball\_pos[1] == 7 and ((ball\_pos[0] - paddle\_x) % 8) in [0, 1, 2]:  ball\_vel[1] \*= -1  # Check for ground collision  if ball\_pos[1] > 7:  game\_over = True  break  # Check for brick collisions  hit\_brick = (ball\_pos[0], ball\_pos[1])  if hit\_brick in bricks:  ball\_vel[1] \*= -1  # Update brick state  if bricks[hit\_brick] == 'yellow':  bricks[hit\_brick] = 'purple'  elif bricks[hit\_brick] == 'purple':  del bricks[hit\_brick]  # Draw updated positions  draw\_bricks()  draw\_paddle()  draw\_ball()  # Check for win condition  if not bricks:  sense.show\_message("You Win!")  break  time.sleep(0.3)  if game\_over:  sense.show\_message("Game Over!") |

|  |
| --- |
| **Practice 2: Embed the video link (YouTube) please refer to the assignment document.** |
| [HW08-Practice 2](https://www.youtube.com/embed/Omc5VPLHMUw?feature=oembed)  [Practice 2 lose](https://www.youtube.com/embed/AMDyFSvZb9g?feature=oembed) |