**Python Game Project**

**Muhammad Sikandar Hussain**

**CMS ID: 502808**

**Summary of Gameplay:**

* The player controls a character that runs and jumps over obstacles.
* The objective is to avoid obstacles by jumping at the right time.
* A score is maintained based on how many obstacles the player successfully jumps over.
* The game ends when the player collides with an obstacle, and the player can restart by pressing the spacebar.

**How I made this game?**

I took a lecture on Youtube to know about pygame before making this game I tried many other also but I found this good. I have watched a lecture on youtube for 4 hours and made this game. I have made changes in this however sprite is same because spirites are not available free on google. I have made a lot of changes in this game like made my own variables and keywords.

**Description of Game:**

* **Initializing Pygame and Game Components:**
* pygame.init() initializes the Pygame library.
* A game window of size 800x400 is created using pygame.display.set\_mode().
* The game’s title is set to "Runner Game" using pygame.display.set\_caption().
* The game uses pygame.time.Clock() to control the frame rate.
* **Font and Game States:**
* A font (test\_font) is loaded with size 50 to render text in the game.
* The variable game\_active controls whether the game is running (True) or over (False).
* **Loading Assets:**
* background\_img, obstacles\_img, and player\_img load the background, obstacle (a snail in this game), and player images respectively.
* **Game Over screen:**
* The message "Game Over Press space to Restart" is rendered when the game ends. This is displayed on the screen when the player collides with an obstacle.
* **Score Tracking:**
* The score variable tracks how many obstacles the player has successfully passed. The score is incremented every time the player jumps over an obstacle.
* The score is displayed at the top-left corner of the screen during gameplay.
* **Obstacle Movement and Collision:**
* The obstacle moves from right to left at a constant speed of 5 units per frame.
* When the obstacle moves off-screen, it reappears on the right side of the screen to simulate continuous gameplay.
* If the player’s character collides with the obstacle (obstacle\_rect.colliderect(player\_rect)), the game stops, and the game-over screen is displayed.
* **Game Restart Mechanism:**
* When the game is over and the player presses the spacebar again, the game resets by setting game\_active = True, repositioning the obstacle, and resetting the score.
* **Game Loop:**
* The core game logic runs inside a while True loop.
* The loop handles player inputs, updates game components (like player movement, obstacle position, score), checks for collisions, and renders the screen.
* The screen updates at 60 frames per second using clock.tick(60).