# INDEX:

* **CLASS : BCA – AI & DS**
* **SEC TION: “D”**
* **YEAR & SEM: I & I**
* **PROJECT:1**
* **SUBJECT CODE: HBCA21A02**

## TOPIC – PYTHON CODE FOR CREATING A

**SNAKE GAME**

ABSTRACT :

This Python code presents a basic implementation of the classic Snake Game using the Pygame library. The game features a snake that moves within a window, consuming red food items to grow longer. The user controls the snake's direction using arrow keys, and the game ends if the snake collides with the window boundaries or itself. The code employs Pygame for graphical rendering, creating a visually interactive gaming experience. The user has the option to quit the game or restart it upon game over. This project serves as an accessible introduction to game development in Python, providing hands-on experience with user input, collision detection, and basic game mechanics. The simplicity of the code encourages beginners to explore and expand upon the fundamentals of Python programming and game development.

**Aim of the Snake Game Project:**

The primary aim of this Snake Game project is to introduce beginners to the fundamentals of game development using Python and the Pygame library. The project is designed to achieve the following objectives:

OBJECTIVE :

1. **Introduction to Pygame:** Provide a practical introduction to Pygame, a popular library for game development in Python.
2. **User Input Handling:** Familiarize users with capturing and responding to user input, enabling them to control the snake's movement within the game.
3. **Collision Detection:** Introduce the concept of collision detection, a crucial aspect of game development, by implementing checks for collisions with boundaries and the snake's own body.
4. **Graphics and Visualization:** Explore basic graphics rendering using Pygame, showcasing how to display elements such as the snake, food, and game window.
5. **Game Loop:** Demonstrate the implementation of a game loop, a fundamental structure in game development, to continuously update the game state and handle user input.
6. **Scorekeeping:** Implement a simple scoring system to track and display the player's score based on the length of the snake.
7. **Game Over Handling:** Illustrate how to manage game over scenarios, allowing users to either quit the game or restart it for further gameplay.
8. **Encouraging Exploration:** Encourage users to explore and modify the code, fostering a sense of creativity and curiosity to enhance the game or experiment with new features.

**SOURCE CODE:**

**import pygame**

**import time**

**import random**

**pygame.init()**

**# Define colors**

**white = (255, 255, 255)**

**yellow = (255, 255, 102)**

**black = (0, 0, 0)**

**red = (213, 50, 80)**

**green = (0, 255, 0)**

**blue = (50, 153, 213)**

**# Set up the game window**

**dis\_width = 800**

**dis\_height = 600**

**dis = pygame.display.set\_mode((dis\_width, dis\_height))**

**pygame.display.set\_caption('Snake Game by OpenAI')**

**# Set up the game clock**

**clock = pygame.time.Clock()**

**snake\_block = 10**

**snake\_speed = 15**

**font\_style = pygame.font.SysFont(None, 50)**

**# Function to draw the snake**

**def our\_snake(snake\_block, snake\_list):**

**for x in snake\_list:**

**pygame.draw.rect(dis, green, [x[0], x[1], snake\_block, snake\_block])**

**# Function to display the score**

**def Your\_score(score):**

**value = font\_style.render("Your Score: " + str(score), True, white)**

**dis.blit(value, [0, 0])**

**# Function to run the game**

**def gameLoop():**

**game\_over = False**

**game\_close = False**

**# Initialize the snake**

**x1 = dis\_width / 2**

**y1 = dis\_height / 2**

**x1\_change = 0**

**y1\_change = 0**

**snake\_list = []**

**length\_of\_snake = 1**

**# Initialize the food position**

**foodx = round(random.randrange(0, dis\_width - snake\_block) / 10.0) \* 10.0**

**foody = round(random.randrange(0, dis\_height - snake\_block) / 10.0) \* 10.0**

**while not game\_over:**

**while game\_close == True:**

**dis.fill(blue)**

**Your\_score(length\_of\_snake - 1)**

**pygame.display.update()**

**for event in pygame.event.get():**

**if event.type == pygame.KEYDOWN:**

**if event.key == pygame.K\_q:**

**game\_over = True**

**game\_close = False**

**if event.key == pygame.K\_c:**

**gameLoop()**

**for event in pygame.event.get():**

**if event.type == pygame.QUIT:**

**game\_over = True**

**if event.type == pygame.KEYDOWN:**

**if event.key == pygame.K\_LEFT:**

**x1\_change = -snake\_block**

**y1\_change = 0**

**elif event.key == pygame.K\_RIGHT:**

**x1\_change = snake\_block**

**y1\_change = 0**

**elif event.key == pygame.K\_UP:**

**y1\_change = -snake\_block**

**x1\_change = 0**

**elif event.key == pygame.K\_DOWN:**

**y1\_change = snake\_block**

**x1\_change = 0**

**if x1 >= dis\_width or x1 < 0 or y1 >= dis\_height or y1 < 0:**

**game\_close = True**

**x1 += x1\_change**

**y1 += y1\_change**

**dis.fill(blue)**

**pygame.draw.rect(dis, red, [foodx, foody, snake\_block, snake\_block])**

**snake\_head = []**

**snake\_head.append(x1)**

**snake\_head.append(y1)**

**snake\_list.append(snake\_head)**

**if len(snake\_list) > length\_of\_snake:**

**del snake\_list[0]**

**for x in snake\_list[:-1]:**

**if x == snake\_head:**

**game\_close = True**

**our\_snake(snake\_block, snake\_list)**

**Your\_score(length\_of\_snake - 1)**

**pygame.display.update()**

**if x1 == foodx and y1 == foody:**

**foodx = round(random.randrange(0, dis\_width - snake\_block) / 10.0) \* 10.0**

**foody = round(random.randrange(0, dis\_height - snake\_block) / 10.0) \* 10.0**

**length\_of\_snake += 1**

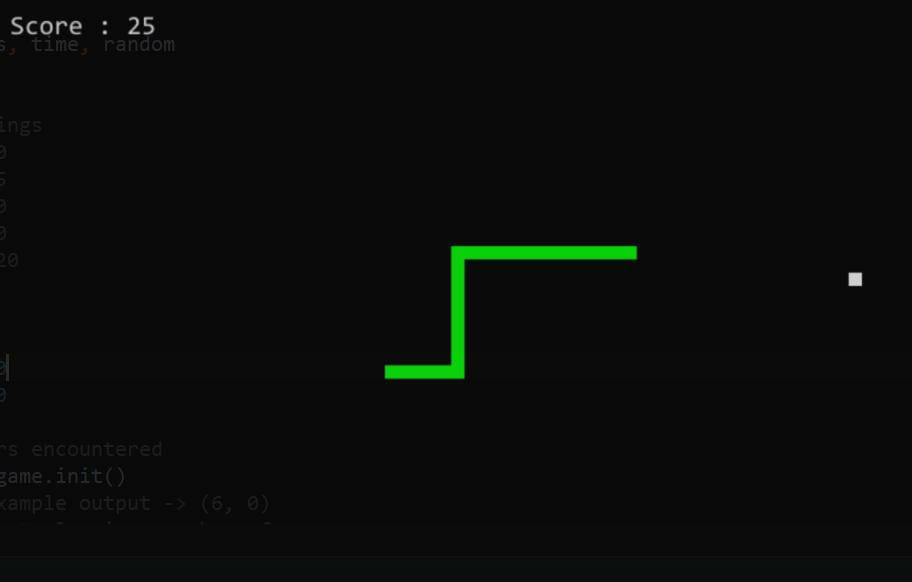
**clock.tick(snake\_speed)**

**pygame.quit()**

**quit()**

**gameLoop()**

**OUTPUT :**



**CONCLUSION :**

**In conclusion, the Snake Game project using Python and Pygame provides a practical entry point into game development, offering hands-on experience with fundamental concepts. By navigating through user input, collision detection, and game loop implementation, this project not only enhances Python programming skills but also lays the groundwork for future exploration in game development. The simplicity of the code encourages beginners to experiment and serves as a stepping stone for those venturing into the dynamic world of interactive application development.**

Top of Form

Top of Form

Top of Form