Table of Contents

Introduction/Concept**2**

Modules3

Code Explanation5

**Snake Game Concept**

**Objective:**  
The objective of this project is to develop a classic Snake game using the Pygame library in Python using Object Oriented Programming (OOP), Inheritance, and Modules. OOP is a programming paradigm that uses objects and classes to structure software. This approach enhances code reusability, readability, and ease of maintenance. Inheritance allows us to define a child class that inherits all the methods and properties from a parent class. Modules are simply files with the “. py” extension containing Python code that can be imported inside another Python Modules Operations Program. I take advantage of modules to split up my code into sections making it easier to debug and locate something. The game will involve controlling a snakes’ movement with the aim to consume the apple without colliding with the borders causing the snake to get longer while simultaneously increasing your score and allowing you to play longer.

**How to play:**

Run the main.py code. The pygame window will open, press on it and using the arrows on your keyboard or the WASD keys, select a direction to go. Get to the apple and once the snake head reaches the apple your score will increase by 1 and another apple will be placed. Avoid the borders of the grid as well as colliding the snake head with it’s body. Because if you collide with them the game will end. Please press 1 key at once, as pressing 2 keys at once could result in the game ending depending on the circumstances. Last but not least, aim to eat as many apples as possible and have fun!

Please read the DOC strings in the code for better visualisation and understanding.

**Note:**

If you run the game and it prints out “ Final score 0 “ without you playing, just run it again and it will work.

I have uploaded my project on GitHub, here is the link:

https://github.com/adamaccs/Adams-Snake-Game

**Game Elements:**

1. Snake:

* The player controls the movement of the snake which is represented by a snake head and a snake body.
* The snake moves block by block.
* The length of the snake body increases by one block each time it consumes an apple.

2. Apple:

* An apple randomly appears on the game board at different locations.
* When the apple is consumed by the snake, it respawns elsewhere on the game board.
* Every time an apple is consumed, the score increases by 1.

3. Game Board:

* The game board is a rectangle with a grid on it consisting of blocks or cells.
* The snake and apple are within the boundaries of the game board.
* (DINT TOUCH BOUNDARRIES GAME OVER)

4. Score:

* The score increases by 1 each time an apple is consumed by the snake.
* When the game is over, your final score is given.

**Modules Used:**

1. Pygame: Pygame is a popular open-source library for making video games in Python. It provides modules designed to handle graphics, sound, and other multimedia elements necessary for game development. Pygame is built on top of the Simple DirectMedia Layer (SDL) library, which is widely used in the industry for game development. /Pygame is used in the Snake Game to handle graphics, user input, and game timing, making it easier to create and manage the game's visual and interactive elements.
2. Random: Random is a well-known Python module used to generate random numbers and perform random operations. It provides functions for generating random integers, floating-point numbers, selecting random elements from a sequence, and more. These are pseudo-random numbers means they are not truly random. / Random is used to place the apple at random positions on the gameboard. It is specifically used to generate random coordinates for the apple, ensuring it appears at different locations each time it is eaten. This adds an element of unpredictability and challenge to the game.
3. Enum: An Enum, short for enumeration, is a data type in Python used to define a set of named constants. Enums make code more readable and maintainable by providing descriptive names for values. /The Direction Enum is used to represent the possible movement directions of the snake: RIGHT, LEFT, UP, and DOWN. It also assigns a number to them. This Enum helps in improving code readability and clarity by providing meaningful names for the directions, making it easier to understand, use, and maintain the code.
4. Named Tuple: A named tuple is a data structure in Python that behaves like a regular tuple but allows accessing elements by name as well as by index. It combines the benefits of tuples (immutable, lightweight) with the ability to access elements by meaningful names. Named tuple is used to represent coordinates on the game grid. It consists of two fields: x and y, representing the horizontal and vertical positions, respectively. By using a named tuple, you can access these coordinates using descriptive names (point.x and point.y) instead of relying solely on indices (point[0] and point[1]). This improves code readability and makes the code more self-explanatory. Additionally, named tuples are immutable, ensuring that the coordinates remain consistent throughout the game.

**Directory Structure:**

AdvCodProject/

│

├── main.py

├── snakegame.py

├── gamebase.py

├── utilities.py

├── apple.png

└── snakehead.png

└── setup.py

**File Modules:**

1. main.py: Entry point for the SnakeGame, it initializes the game and starts the main loop. Contents:

* Imports the SnakeGame class from file snakegame.py.
* Initializes a SnakeGame instance.
* Runs the main game loop which calls the play\_step() method until the game is over.
* Prints the final score when the game is over.

1. utilities.py: Contains the utility functions, constants, and enumeration. It is used across different modules of the project. Contents:

* Constants: Define RGB colors, BLOCK\_SIZE, SPEED, SCORE\_AREA, and SPEED\_INC.
* Enums: Direction enum to represent the movement directions.
* Namedtuples: Point namedtuple to represent the coordinates (x, y).

1. gamebase.py: Defines the GameBase class, which is the parent class.

Contents/Methods:

* Initializes the game base with a specific width and height.
* Draws the grid on the game window.

1. snakegame.py: Contains the SnakeGame class which contains many methods and logics.

Contents/Methods:

* Inherits height and width from parent class GameBase and initializes them.
* Places food randomly using the random module and ensures that the food/apple isn’t placed on top of the snake.
* Excecutes a single step of the game, handles user input, updates game, chechs for collisions.
* Checks if the snake has collided with the boundaries or itself.
* Updates the game’s user interface by redrawing the snake, food, and score.
* Moves the snake in the specified direction.

**Images:**

This project utilizes 2 images. The 1st image is the apple. It is used as food and is displayed in the game for a more realistic and fun experience. The 2nd image is the snakes’ head. It is used to give a more realistic and fun experience. It also reacts to the direction change making it more reactive. They are both available in the snakegame.py file.

**Code Explanation by Module:**

main.py Module:

This module contains the entry point of the game.

* The game is initialized and started.
* It contains the main game loop that keeps running until the game is over.
* The loop repeatedly calls the play\_step() method to progress the game by one step.
* It checks if the game is over and prints the final score when the game ends.

utilities.py Module:

This module handles utility functions and data structures used throughout the game.

* Imports essential libraries such as random, Enum from enum, and namedtuple from collections.
* Defines the Direction enum to represent the possible movement directions of the snake.
* Uses namedtuple to define a Point object representing coordinates on the game grid.
* Sets up RGB color constants and game constants like block size, game speed, score area height, and speed increment.

game\_base.py Module:

This module sets up the basics of the game using the Pygame library.

* Imports the Pygame library and initializes it.
* Sets the font for displaying the score.

snake\_game.py Module:

This module contains the main logic for the Snake game, encapsulated within the SnakeGame class.

* Initializes the game window, sets the width and height of the game board, and reserves space for the score display.
* Initializes the display with the name "Snake" and sets the default game direction to none.
* Places the snake head at the starting location (middle) and initializes the snake body.
* Sets the initial score to 0 and places an apple on the game board.
* Loads the apple and snake head images and resizes them appropriately.
* Contains a method drawGrid to draw the game grid below the score area.
* Contains a method \_place\_food to place an apple at a random spot on the game board that does not overlap with the snake and is below the score area.
* Contains a method play\_step that:
* Handles the event loop to process user inputs and quits the game if the quit event is detected.
* Allows the user to control the snake using arrow keys or WASD keys.
* Prevents the snake from moving into itself.
* Keeps the snake in a stable position if no direction is set; otherwise, moves the snake in the assigned direction.
* Checks if the game is over by detecting collisions with the game board boundaries or the snake itself.
* Increases the score when the snake eats an apple, places a new apple, and increases the game speed after a certain score.
* Updates the UI and sets the game frame rate.
* Contains a method \_is\_collision to check if the snake has collided with the boundaries or itself, ending the game.
* Contains a method \_update\_ui to:
* Draw the grid and set the background color to white.
* Update the UI by determining the orientation of the snake head based on its direction.
* Draw the snake body with a colorful design.
* Draw the apple and snake head at their respective locations.
* Display the current score.
* Contains a method \_move to move the snake in the specified direction.