

SNAKE GAME REPORT

Ali Benazzouz

1/ Introduction :

The Snake Game is a classic arcade game where the player controls a snake that grows in length by eating food while avoiding collisions with the walls of the game window and its own body.

The concept occurred to me while contemplating the possibility of undertaking a project that would be minimalist in nature but not overly simple to design.

As an avid enthusiast of retro games like Tetris and Snake, with Tetris already having been created, Snake was the only option remaining game that held a special place in my childhood memories.

2/ Game setup and initialization :

I first decided to divide my project into small tasks, each task would lead to the beginning of the following one.

In the one hand, I focused on creating the actual “map” with the snake at it’s first form with random food that pops anywhere in the map.

2.1/ Used libraries :

The Snake Game utilizes the Pygame library, which is a popular choice for developing 2D games in Python.

The game window is initialized with a default size of 640x480 pixels and a title of "Snake Game." Additionally, essential colors for the game, such as black, green, red, and white, are defined to be used in various elements of the game's visuals.

2.2/ Snake and game elements :

The snake is represented as a collection of connected squares or segments. The size of each segment is defined by the variable “snake_size”.

The initial position of the snake is set at the center of the game window (window_width // 2, window_height // 2).

The snake's movement direction is controlled by “snake_dx” and “snake_dy” variables, which determine the number of pixels the snake moves horizontally and vertically in each frame.

The game keeps track of the player's score, which initially starts at zero. The score is displayed on the game window using a font obtained from the “pygame . font” module. Additionally, the game maintains the position of the food, which the snake must consume to increase its length and score. The food is randomly positioned within the game window boundaries using the “random “ module.

2.3/ Game Loop and Event Handling :

The game logic is contained within a `game_loop()` function, which serves as the main game loop. The loop runs until the `game_over` variable is set to True, indicating that the game has ended. Within the loop, the function processes various events, including user input and game-related events such as quitting the game.

The player controls the snake's movement using the arrow keys. Each key press event is captured and triggers a change in the snake's direction, ensuring that the snake cannot reverse its course into its own body. Additionally, pressing the spacebar pauses the game, displaying a "PAUSED" message on the screen. The player can resume the game by pressing space again or quit by pressing Q.

Snake Movement and Collision Detection: The snake's position is updated in each frame based on its current direction. The `snake_x` and `snake_y` variables are incremented or decremented by the values of `snake_dx` and `snake_dy`, respectively, moving the snake across the game window. Collision detection occurs in multiple scenarios.

Firstly, if the snake's position matches the position of the food, it means the snake has "eaten" the food. In such cases, the score is incremented, and a new food position is randomly generated within the game window.

Secondly, collision detection is performed between the snake's head and its body segments. If the snake's head collides with any segment of its body, it indicates that the snake has collided with itself, resulting in a game over condition.

Thirdly, the snake's position is checked against the boundaries of the game window. If the snake goes beyond these boundaries, it collides with the walls, leading to the end of the game.

2.4/ Rendering and Visuals :

The game window is filled with a white color (`WHITE`) in each frame to provide a clean canvas for rendering the game elements. The food is represented by a red square (`RED`) with the size of `snake_size`. The snake is drawn by iterating through its list of segments (`snake_list`) and rendering each segment as a green square (`GREEN`) with the size of `snake_size`.

The player's score is continuously displayed on the screen using a font obtained from the `pygame . font` module. The score is rendered in black color (`BLACK`) at the top left corner of the game window.

2.5/ Game Over and Restart :

When the game ends, a red screen is displayed, and a "GAME OVER" message is shown at the center of the screen. The player is provided with options to either restart the game or quit.

If the player chooses to restart, the game variables, such as snake position, direction, score, and length, are reset to their initial values. A new game loop is initiated, allowing the player to start playing again.

If the player chooses to quit, the game window is closed, and the program terminates.

3/ Player Feedback and User Experience

3.1/ Player Feedback :

During the development process of the Snake Game, I sought feedback from some friends to gather their thoughts and impressions. The feedback I received was valuable in understanding the game's strengths and areas for improvement.

1/ Mathis Goudjil commented on the addictive nature of the game, mentioning that he found themselves constantly trying to beat their high score. He appreciated the simplicity of the gameplay and the nostalgic feel it evoked.

2/ Big Mike mentioned that they enjoyed the smooth controls and responsiveness of the snake's movement. They found the difficulty progression to be well-balanced, offering a challenge without becoming frustrating.

3/ Albion Thomas provided some constructive criticism, suggesting the inclusion of different levels with varying obstacles or patterns to add more variety and depth to the gameplay.

3.2/ User Experience :

The user experience of the Snake Game is designed to be intuitive, engaging, and visually appealing. Here are some key aspects that contribute to the overall user experience:

1. **Intuitive Controls:** The game's controls, using arrow keys for snake movement, are simple and familiar to most players. They allow for precise maneuvering, which is essential for avoiding obstacles and collecting food.
2. **Visual Clarity:** The game utilizes a clean and minimalist design, with contrasting colors for the snake, food, and background. This enhances visibility and makes it easy for players to identify the elements on the screen.
3. **Fluid Gameplay:** The smooth movement of the snake and the responsive controls create a seamless and enjoyable gameplay experience. The collision detection is precise, ensuring that the game accurately registers when the snake interacts with food or itself.
4. **Immersive Sound Effects:** The inclusion of sound effects, such as the satisfying "chomp" sound when the snake consumes food, adds to the immersion and enhances the overall experience of playing the game.
5. **Addictive Gameplay:** The combination of simple mechanics, increasing difficulty, and the drive to achieve a high score creates an addictive gameplay loop. Players often find themselves immersed in the game, constantly striving to improve their performance.
6. **Visual Enhancements:** To enhance the visual experience, I added subtle animations, such as a smooth transition when the snake changes direction or a blinking effect when the snake collides with an obstacle. These small details contribute to the overall polish and appeal of the game.

In conclusion, the feedback from players and the user experience of the Snake Game showcase its appeal and engagement. The intuitive controls, clear visuals, and addictive gameplay make it an enjoyable experience for players. The feedback received has provided valuable insights for potential improvements, such as adding different levels. By incorporating player feedback and continually refining the user experience, my Snake Game can be further enhanced to provide an even more captivating gaming experience for players of all ages.

4/ Conclusion :

In conclusion, the Snake Game that I created holds a special place in my heart. It was inspired by my love for retro games like Tetris and Snake, which brought back nostalgic memories from my childhood. I wanted to undertake a minimalist project that would challenge me in terms of design and coding skills, and the Snake Game fit perfectly.

Developing the game using the Pygame library allowed me to bring the classic Snake gameplay to life. I enjoyed implementing the movement mechanics, collision detection, and scoring system, as well as adding some extra features like pausing the game and providing options to restart or quit. It was a fulfilling experience to see the snake grow longer as it consumed the food and to witness the game over screen when the snake collided with itself or the walls.

Throughout the development process, I encountered challenges and learned valuable lessons in game programming. I gained a deeper understanding of event handling, rendering graphics, and managing game states. Debugging and testing the game helped me refine my problem-solving skills and improve the overall quality of the gameplay.

The Snake Game project allowed me to combine my passion for gaming with my coding abilities. It not only provided me with a creative outlet but also served as a reminder of the simple joys and entertainment that retro

games can bring. I hope that the others students can experience the same sense of nostalgia and enjoyment while playing my Snake Game.

In the future, I plan to continue exploring game development and further enhance the Snake Game by adding new features, levels, and perhaps even multiplayer functionality. I believe that this project is just the beginning of my journey into the exciting world of game development, and i look forward to creating more engaging and memorable gaming experiences.