DAY 3

Note: The names for my HTML, CSS, and JavaScript files (like DAYTHREE.html, DAYTHREE.css, DAYTHREE.js) are **not recommended** as they are considered **bad practice** in professional projects. However, since this is a **mini project**, I prefer using these names to **distinguish them from my other files**. For **larger or professional projects**, it's better to use standard and descriptive file names like index.html, style.css, and script.js.

DAYTHREE.HTML

<link rel="shortcut icon" href="URL">

This line adds a **favicon** (the small icon shown in the browser tab) to your webpage:

<h1 style="...">DAY-3</h1>

- Shows a **heading** titled **DAY-3**.
- Underlined and styled in **white** color.

** Score: 0 **

- Displays the **current score** during the game.
- Score updates live using JavaScript.

<canvas id="gameCanvas"></canvas>

- The **main game area** where the snake moves and eats food.
- JavaScript draws the game on this canvas.

** **

- Contains **control buttons**:

NEW GAME

- Calls the restartGame() function to start a new game.

PAUSE

- Calls the togglePause() function to pause or resume.

ABOUT

- Opens a **PDF file** named about.pdf in a new browser tab.

** **

- Appears when the game ends (Game Over).
- Shows the **final score**.
- Has a **Restart** button to replay the game.

DAYTHREE.CSS

body

- **Sets the entire page's background and text:**
  - background-color: #111; → dark background (almost black)
  - color: #0f0; → neon green text
  - font-family: monospace; → computer-style font
  - text-align: center; → center-aligns content
  - margin: 0; padding: 20px; → removes default space and adds padding around content

#scoreBoard

- Styles the **score display** above the game:
  - font-size: 20px; → makes the score text a bit larger
  - margin-bottom: 10px; → gives space between score and canvas

#gameCanvas

- Styles the **game area** (canvas where the snake moves):
  - border: 2px solid #0f0; → green border around the game area
  - display: block; margin: 0 auto; → centers the canvas on the page
  - background-color: #000; → sets the game background to black

#controls

- Container that holds the control buttons (NEW GAME, PAUSE, ABOUT):

- margin-top: 15px; → space above the buttons

#controls button

- Styles each **button**:
  - background-color: darkorange; → orange button background
  - color: white; → white text on buttons
  - border: none; border-radius: 6px; → no border and rounded corners
  - padding: 12px 24px; → adds space inside the button
  - margin: 5px; → spacing between buttons
  - font-size: 16px; font-weight: bold; → bigger and bolder text
  - cursor: pointer; → cursor turns to hand on hover
  - transition: background-color 0.3s; → smooth color change when hovered

#controls button:hover

- When you **hover** the mouse over a button:
  - background-color: orangered; → turns a deeper red-orange

#gameOverScreen

- Hidden by default (display: none;)
- Styles the **Game Over popup**:
  - display: flex; flex-direction: column; → shows elements vertically
  - align-items: center; justify-content: center; → centers content
  - position: absolute; top: 30%; → places popup around the middle of the page
  - text-align: center; color: white; → white text, centered

#gameOverScreen button

- Styles the **Restart button** shown after Game Over:
  - Same orange color scheme as other buttons
  - margin-top: 10px; → adds space above it
  - font-weight: bold; → bold text
  - cursor: pointer; → hand cursor on hover

DAYTHREE.JS

1. Get the canvas and set its size

```
const canvas = document.getElementById("gameCanvas");
const ctx = canvas.getContext("2d");
canvas.width = 700;
canvas.height = 300;
```

- Finds the game screen (<canvas>)
- Makes it **wider** (700px) and **not too tall** (300px)
- Gets the 2D drawing area to draw snake, food, etc.

2. Game setup / variables

```
const box = 20;
let snake = [{ x: 200, y: 100 }];
let direction = "RIGHT";
let food = spawnFood();
let score = 0;
let gameInterval;
let isPaused = false;
let gameOver = false;
```

- Each **box** is a 20x20 square on the grid
- Snake starts at {200, 100}
- direction tells where the snake moves
- food spawns in a random place
- score keeps count of how much food eaten
- isPaused & gameOver are switches to control the game

3. Load apple image

```
const appleImg = new Image();
appleImg.src = "https://cdn-icons-png.flaticon.com/512/415/415733.png";
```

- Loads an **apple image** from the internet
- Will be used as the **food**

4. Keyboard controls

```
document.addEventListener("keydown", changeDirection);
```

- Listens when a key is pressed (arrows or P)

- Calls changeDirection() to update snake's direction or pause the game

5. changeDirection(event)

```
function changeDirection(event) {
  if (Arrow key pressed and not opposite) {
    update direction
  } else if (key is P or p) {
    togglePause();
  }
}
```

- Makes sure the snake doesn't go backwards
- If P is pressed → pause or resume

6. spawnFood()

```
function spawnFood() {
  return {
    x: random box x,
    y: random box y
  };
}
```

- Creates food in a **random spot** on the canvas grid

7. drawGame() → Main game loop

```
function drawGame() {
  if (gameOver) return;

  clear canvas;

  draw food (apple image or red circle);

  move snake in current direction;

  check if snake hit wall or itself → game over;

  if snake eats food:
    increase score;
    spawn new food;
```

```
  else:
    remove tail;

  update score display;

  draw snake segments;
  if paused → make snake blink;
}
```

More details:

- Snake's head moves 1 box forward each time
- If head hits food → score goes up
- If not → tail is removed so snake moves
- If snake hits wall or itself → game over
- Snake is drawn in **green**, head in **white**
- When paused → snake **blinks**

8. togglePause()

```
function togglePause() {
  if (gameOver) return;

  if paused → resume game;
  if running → pause game;
}
```

- Stops or resumes the game based on current state

9. showScore()

```
function showScore() {
  alert("Your Score: " + score);
}
```

- Pops up your current score

10. restartGame()

```
function restartGame() {
  reset all variables;
  hide game over screen;
```

```
  start game again;
}
```

- Resets everything
- Starts a **new game** from the beginning

11. Start the game

```
gameInterval = setInterval(drawGame, 150);
```

- Runs drawGame() every **150 milliseconds**
- Makes the game run in a loop

Section	Purpose
Canvas setup	Create and size the game screen
Game variables	Store snake, food, score, etc.
Apple image	Load picture to use as food
Controls	Let player use arrow keys and pause
spawnFood()	Puts food in a new random spot
drawGame()	Main game logic and drawing
togglePause()	Pause or resume the game
showScore()	Show score in a pop-up
restartGame()	Start a new game
Game start	Starts the game loop