CODE-CODE GAME INDEX.HTML ,APP.JS, STYLE.CSS

1.GAME TIC-TAC-TOE

- **Index.html**

```
<!DOCTYPE html>
<html lang="en">

<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Tic-Tac-Toe</title>
   <link rel="stylesheet" href="style.css"/>
</head>

<body>
   <main>

      <div class="msg-container hide">
         <p id="msg">Winner</p>

         <button id="new-btn">New Game</button>
      </div>



      <div class="container">
         <h1 id="hd">Tic Tac Toe</h1>
         <div class="game">
            <button class="box"></button>
            <button class="box"></button>
            <button class="box"></button>
            <button class="box"></button>
            <button class="box"></button>
            <button class="box"></button>
            <button class="box"></button>
            <button class="box"></button>
            <button class="box"></button>
         </div>
         <button id="reset-btn">Reset Game</button>
      </div>
```

```
    </main>
    <script src="app.js"></script>
</body>

</html>
```

- **APP.JS**

```
let boxes = document.querySelectorAll(".box");
let resetBtn = document.querySelector("#reset-btn");
let newGameButton = document.querySelector("#new-btn");
let msgContainer = document.querySelector(".msg-container");
let msgContainer2 = document.querySelector(".msg-container2");
let msg = document.querySelector("#msg");
let msg2 = document.querySelector("#msg2");
let container = document.querySelector(".container");
let count = 0;
let turnO = true;
const winPatterns = [
    [0, 1, 2],
    [0, 3, 6],
    [0, 4, 8],
    [3, 4, 5],
    [6, 7, 8],
    [1, 4, 7],
    [2, 5, 8],
    [2, 4, 6]
];

const resetGame = () => {
    turnO = true;
    count=0;
    enableBoxes();
    msgContainer.classList.add("hide");
    container.classList.remove("hide");
}
boxes.forEach((box) => {
    box.addEventListener("click", () => {
        if (count === 9 || checkWinner()) {
```

```
            return;
        }
        if (turnO) {
          //playerO
          box.innerText = "O";
          box.style.color="white";
          turnO = false;
        } else {
          //playerX
          box.innerText = "X";
          box.style.color="cyan";
          turnO = true;
        }
        box.disabled = true;
        count++;

        let isWinner = checkWinner();

        if (count === 9) {
          drawGame();
        }
      });
    });

const disableBoxes = () => {
    for (let box of boxes) {
        box.disabled = true;
    }
}
const enableBoxes = () => {
    for (let box of boxes) {
        box.disabled = false;
        box.innerText = "";
    }
}
const showWinner = (winner) => {
    msg.innerText = Congratulations, Winner is ${winner};
    msgContainer.classList.remove("hide");
    container.classList.add("hide");
    if(hideDisplay=="inline"){
```

```
        hide2.style.display="none";
    }
    disableBoxes();
}

const drawGame = () => {
    if (!checkWinner()) {
        msg.innerText = "This Game is a Draw.";
        msgContainer.classList.remove("hide");
        container.classList.add("hide");
        disableBoxes();
    }
}

const checkWinner = () => {
    for (let pattern of winPatterns) {

        let pos1Val = boxes[pattern[0]].innerText;
        let pos2Val = boxes[pattern[1]].innerText;
        let pos3Val = boxes[pattern[2]].innerText;
        if (pos1Val != "" && pos2Val != "" && pos3Val != "") {
            if (pos1Val === pos2Val && pos2Val === pos3Val) {
                console.log("winner is ", pos1Val)
                showWinner(pos1Val);
                return true;
            }

        }
    }
    return false;
};

newGameButton.addEventListener("click", resetGame);
resetBtn.addEventListener("click", resetGame);
```

- **STYLE.CSS**

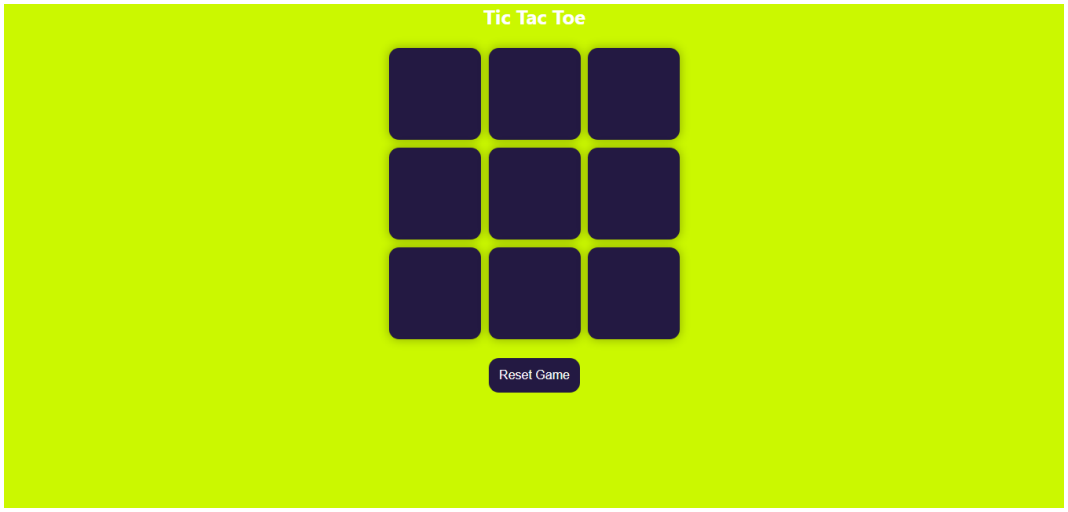
```
*{
    margin: 0;
    padding: 0;
```

```
}

body{
    background-color: #cbf800;
    text-align: center;
}

#hd{
    margin:30px;
    margin-top:30px;
    color: #fff;
        font-family: system-ui, -apple-system, BlinkMacSystemFont, 'Segoe UI', Roboto, Oxygen, Ubuntu, Cantarell, 'Open Sans', 'Helvetica Neue', sans-serif;
}
.container{
    margin-top:30px;
    height: 70vh;
    display: flex;
    flex-direction: column;
    justify-content: center;
    align-items: center;

}

.game{
    height: 60vmin;
    width: 60vmin;
    display: flex;
    flex-wrap: wrap;
    justify-content: center;
    align-items: center;
    gap:1.5vh;
}

.box{
    height: 18vmin;
    width: 18vmin;
    border-radius: 1rem;
    border: none;
    box-shadow: 0 0 1rem rgba(0, 0, 0, 0.3);
```

```
    font-size: 8vmin;
    color: #e0b1cb;
    background-color: #231942;
}

#reset-btn{
    margin: 30px;
    padding: 1rem;
    font-size: 1.25rem;
    background-color: #231942;
    border-radius: 1rem;
    border:none;
    color:#fff;
}

#new-btn{
    padding: 1rem;
    font-size: 1.25rem;
    background-color: #231942;
    border-radius: 1rem;
    border:none;
    color:#fff;
}

#msg{
    color: #fff;
    font-size: 8vh;
       font-family: system-ui, -apple-system, BlinkMacSystemFont, 'Segoe UI', Roboto, Oxygen, Ubuntu, Cantarell, 'Open Sans', 'Helvetica Neue', sans-serif;
}

.msg-container{
    height: 100vmin;
    display: flex;
    justify-content: center;
    align-items: center;
    flex-direction: column;
    gap: 2rem;
}
```

```
.hide{
    display: none;
}

.hide2{
    display:inline;
}
```

GAMBAR :



2. GAME ULAR

- **INDEX.HTML**

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Snake Game</title>
  <link rel="stylesheet" href="style.css">
</head>
<body>
  <h1>Snake Game</h1>
  <div id="game-board"></div>
  <p>Score: <span id="score">0</span></p>
```

```
  <script src="app.js"></script>
</body>
</html>
```

- **APP.JS**

```
const board = document.getElementById("game-board");
const scoreDisplay = document.getElementById("score");
const boardSize = 20;
let snake = [{ x: 10, y: 10 }];
let food = { x: 5, y: 5 };
let dx = 0, dy = 0;
let score = 0;

function draw() {
  board.innerHTML = "";

  // Draw food
  const foodEl = document.createElement("div");
  foodEl.style.gridColumnStart = food.x;
  foodEl.style.gridRowStart = food.y;
  foodEl.classList.add("food");
  board.appendChild(foodEl);

  // Draw snake
  snake.forEach(segment => {
    const el = document.createElement("div");
    el.style.gridColumnStart = segment.x;
    el.style.gridRowStart = segment.y;
    el.classList.add("snake");
    board.appendChild(el);
  });
}

function moveSnake() {
  const head = { x: snake[0].x + dx, y: snake[0].y + dy };

  // Game over if out of bounds or hitting itself
  if (
    head.x < 1 || head.y < 1 ||
```

```
    head.x > boardSize || head.y > boardSize ||
    snake.some(seg => seg.x === head.x && seg.y === head.y)
  ) {
    alert("Game Over! Score: " + score);
    snake = [{ x: 10, y: 10 }];
    dx = dy = 0;
    score = 0;
    scoreDisplay.textContent = score;
    return;
  }

  snake.unshift(head);

  // If eating food
  if (head.x === food.x && head.y === food.y) {
    score++;
    scoreDisplay.textContent = score;
    placeFood();
  } else {
    snake.pop();
  }
}

function placeFood() {
  food = {
    x: Math.floor(Math.random() * boardSize) + 1,
    y: Math.floor(Math.random() * boardSize) + 1,
  };
}

function update() {
  moveSnake();
  draw();
}

setInterval(update, 150);

document.addEventListener("keydown", e => {
  switch (e.key) {
    case "ArrowUp": if (dy === 0) [dx, dy] = [0, -1]; break;
```

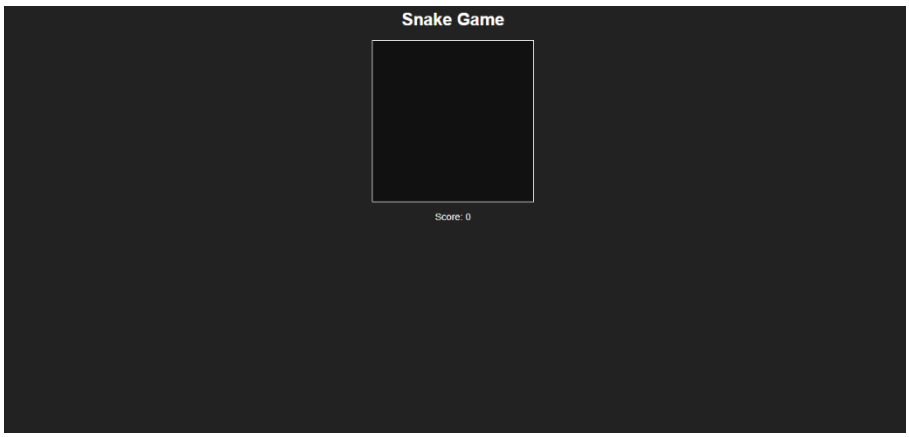
```
    case "ArrowDown": if (dy === 0) [dx, dy] = [0, 1]; break;
    case "ArrowLeft": if (dx === 0) [dx, dy
```

- STYLE.CSS

```
body {
  font-family: Arial, sans-serif;
  background: #222;
  color: #fff;
  text-align: center;
  margin: 0;
  padding: 0;
}

h1 {
  margin: 20px 0;
}

#game-board {
  width: 300px;
  height: 300px;
  background-color: #111;
  margin: 0 auto;
  display: grid;
  grid-template-columns: repeat(20, 15px);
  grid-template-rows: repeat(20, 15px);
  border: 2px solid #fff;
}

.snake {
  background-color: limegreen;
}

.food {
  background-color: red;
}

p {
  font-size: 18px;
```

```
}
```

GAMBAR :



3. GAME CLIKE THE CLIKE

- INDEX.HTML

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0"/>
  <title>Click the Circle</title>
  <link rel="stylesheet" href="style.css"/>
</head>
<body>
  <h1>Click the Circle!</h1>
  <div id="circle-game">
    <div id="circle"></div>
  </div>
      <p>Time Left: <span id="time">30</span>s | Score: <span id="score">0</span></p>
  <script src="app.js"></script>
</body>
</html>
```

- APP.JS

```
const circle = document.getElementById('circle');
const scoreDisplay = document.getElementById('score');
const timeDisplay = document.getElementById('time');

let score = 0;
let timeLeft = 30;

function moveCircle() {
  const x = Math.random() * 260;
  const y = Math.random() * 260;
  circle.style.left = ${x}px;
  circle.style.top = ${y}px;
}

circle.addEventListener('click', () => {
  if (timeLeft > 0) {
    score++;
    scoreDisplay.textContent = score;
    moveCircle();
  }
});

const timer = setInterval(() => {
  timeLeft--;
  timeDisplay.textContent = timeLeft;

  if (timeLeft <= 0) {
    clearInterval(timer);
    circle.style.display = 'none';
    alert(Game over! Your score is ${score});
  }
}, 1000);

moveCircle();
```

- **STYLE.CSS**

```
body {
```

```
  font-family: Arial, sans-serif;
  text-align: center;
  background: radial-gradient(circle, #fbc2eb 0%, #a6c1ee 100%);
  margin: 0;
  padding: 0;
}

h1 {
  margin-top: 30px;
  color: #333;
}

#circle-game {
  position: relative;
  width: 300px;
  height: 300px;
  margin: 30px auto;
  border: 2px solid #fff;
  background-color: #fff;
  overflow: hidden;
  border-radius: 10px;
}

#circle {
  width: 40px;
  height: 40px;
  background-color: purple;
  border-radius: 50%;
  position: absolute;
  top: 50px;
  left: 50px;
  cursor: pointer;
  transition: all 0.2s ease;
}

p {
  font-size: 18px;
  color: #fff;
}
```

GAMBAR :