Assignment No. 1

Github Link : https://github.com/theakshaymore/IP-Assignments

1) Calculator :

- ***HTML CODE***

```
<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="UTF-8" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <meta http-equiv="X-UA-Compatible" content="ie=edge" />

    <link rel="stylesheet" href="style.css" />

    <title>Calculator</title>

  </head>


  <body>

    <div class="container">

      <div class="calculator">

        <input type="text" name="screen" id="screen" />

        <table>

          <tr>

            <td><button>(</button></td>

            <td><button>)</button></td>
```

```
 <td><button>C</button></td>
 <td><button>%</button></td>
</tr>
<tr>
 <td><button>7</button></td>
 <td><button>8</button></td>
 <td><button>9</button></td>
 <td><button>X</button></td>
</tr>
<tr>
 <td><button>4</button></td>
 <td><button>5</button></td>
 <td><button>6</button></td>
 <td><button>-</button></td>
</tr>
<tr>
 <td><button>1</button></td>
 <td><button>2</button></td>
 <td><button>3</button></td>
 <td><button>+</button></td>
</tr>
<tr>
 <td><button>0</button></td>
 <td><button>.</button></td>
 <td><button>/</button></td>
```

```
          <td><button>=</button></td>
         </tr>
        </table>
      </div>
    </div>
  </body>
  <script src="script.js"></script>
</html>
```

- ***CSS CODE***

```
.container {
  text-align: center;
  margin-top: 23px;
}

table {
  margin: auto;
}

input {
  border-radius: 5px;
  /* border: 5px solid #000000; */
  font-size: 34px;
  height: 65px;
  width: 456px;
}
```

```
button {

  border-radius: 10px;

  font-size: 40px;

  background: #e0610e;

  width: 102px;

  height: 90px;

  margin: 6px;

}


.calculator {

  border: 4px solid #000000;

  background-color: chocolate;

  padding: 23px;

  border-radius: 25px;

  display: inline-block;

}


h1 {

  font-size: 28px;

  font-family: "Courier New", Courier, monospace;

}
```

- ***<u>JAVASCRIPT CODE</u>***

```
let screen = document.getElementById("screen");

buttons = document.querySelectorAll("button");
```

```
let screenValue = "";

for (item of buttons) {

  item.addEventListener("click", (e) => {

    buttonText = e.target.innerText;

    console.log("Button text is ", buttonText);

    if (buttonText == "X") {

      buttonText = "*";

      screenValue += buttonText;

      screen.value = screenValue;

    } else if (buttonText == "C") {

      screenValue = "";

      screen.value = screenValue;

    } else if (buttonText == "=") {

      screen.value = eval(screenValue);

    } else {

      screenValue += buttonText;

      screen.value = screenValue;

    }

  });

}
```

2) TicTacToe :

- HTML CODE:

```
<!DOCTYPE html>

<html lang="en">
```

```html
<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <meta http-equiv="X-UA-Compatible" content="ie=edge" />
  <link rel="stylesheet" href="styles.css" />
  <script src="script.js" defer></script>
  <title>Tic Tac Toe</title>
</head>
<body>
  <div class="board" id="board">
    <div class="cell" data-cell></div>
    <div class="cell" data-cell></div>
    <div class="cell" data-cell></div>
    <div class="cell" data-cell></div>
    <div class="cell" data-cell></div>
    <div class="cell" data-cell></div>
    <div class="cell" data-cell></div>
    <div class="cell" data-cell></div>
    <div class="cell" data-cell></div>
  </div>
  <div class="winning-message" id="winningMessage">
    <div data-winning-message-text></div>
    <button id="restartButton">Restart</button>
  </div>
</body>
```

```
</html>
```

- ***<u>CSS CODE</u>***

```
*,
*::after,
*::before {
  box-sizing: border-box;
}

:root {
  --cell-size: 100px;
  --mark-size: calc(var(--cell-size) * 0.9);
}

body {
  margin: 0;
  background-color: wheat;
}

.board {
  width: 100vw;
  height: 100vh;
  display: grid;
  justify-content: center;
  align-content: center;
}

  justify-items: center;

  align-items: center;

  grid-template-columns: repeat(3, auto);

}


.cell {

  width: var(--cell-size);

  height: var(--cell-size);

  border: 1px solid black;

  display: flex;

  justify-content: center;

  align-items: center;

  position: relative;

  cursor: pointer;

}


.cell:first-child,

.cell:nth-child(2),

.cell:nth-child(3) {

  border-top: none;

}


.cell:nth-child(3n + 1) {

  border-left: none;

}
```

```
.cell:nth-child(3n + 3) {
  border-right: none;
}

.cell:last-child,
.cell:nth-child(8),
.cell:nth-child(7) {
  border-bottom: none;
}

.cell.x,
.cell.circle {
  cursor: not-allowed;
}

.cell.x::before,
.cell.x::after,
.cell.circle::before {
  background-color: #e0610e;
}

.board.x .cell:not(.x):not(.circle):hover::before,
.board.x .cell:not(.x):not(.circle):hover::after,
.board.circle .cell:not(.x):not(.circle):hover::before {
  background-color: lightgrey;
}

.cell.x::before,
.cell.x::after,
.board.x .cell:not(.x):not(.circle):hover::before,
.board.x .cell:not(.x):not(.circle):hover::after {
  content: "";
  position: absolute;
  width: calc(var(--mark-size) * 0.15);
  height: var(--mark-size);
}

.cell.x::before,
.board.x .cell:not(.x):not(.circle):hover::before {
  transform: rotate(45deg);
}

.cell.x::after,
.board.x .cell:not(.x):not(.circle):hover::after {
  transform: rotate(-45deg);
}

.cell.circle::before,
.cell.circle::after,
.board.circle .cell:not(.x):not(.circle):hover::before,

.board.circle .cell:not(.x):not(.circle):hover::after {

  content: "";

  position: absolute;

  border-radius: 50%;

}


.cell.circle::before,

.board.circle .cell:not(.x):not(.circle):hover::before {

  width: var(--mark-size);

  height: var(--mark-size);

}


.cell.circle::after,

.board.circle .cell:not(.x):not(.circle):hover::after {

  width: calc(var(--mark-size) * 0.7);

  height: calc(var(--mark-size) * 0.7);

  background-color: white;

}


.winning-message {

  display: none;

  position: fixed;

  top: 0;

  left: 0;
  right: 0;

  bottom: 0;

  background-color: rgba(0, 0, 0, 0.9);

  justify-content: center;

  align-items: center;

  color: white;

  font-size: 5rem;

  flex-direction: column;

}


.winning-message button {

  font-size: 3rem;

  background-color: #e0610e;

  border: 1px solid black;

  padding: 0.25em 0.5em;

  cursor: pointer;

}


.winning-message button:hover {

  background-color: black;

  color: white;

  border-color: white;

}


.winning-message.show {
  display: flex;
}
```

- ***<u>JAVASCRIPT CODE :</u>***

```
const X_CLASS = 'x'
const CIRCLE_CLASS = 'circle'
const WINNING_COMBINATIONS = [
  [0, 1, 2],
  [3, 4, 5],
  [6, 7, 8],
  [0, 3, 6],
  [1, 4, 7],
  [2, 5, 8],
  [0, 4, 8],
  [2, 4, 6]
]
const cellElements = document.querySelectorAll('[data-cell]')
const board = document.getElementById('board')
const winningMessageElement = document.getElementById('winningMessage')
const restartButton = document.getElementById('restartButton')
const winningMessageTextElement = document.querySelector('[data-winning-message-text]')
let circleTurn

startGame()
```

```
restartButton.addEventListener('click', startGame)


function startGame() {
  circleTurn = false
  cellElements.forEach(cell => {
    cell.classList.remove(X_CLASS)
    cell.classList.remove(CIRCLE_CLASS)
    cell.removeEventListener('click', handleClick)
    cell.addEventListener('click', handleClick, { once: true })
  })
  setBoardHoverClass()
  winningMessageElement.classList.remove('show')
}


function handleClick(e) {
  const cell = e.target
  const currentClass = circleTurn ? CIRCLE_CLASS : X_CLASS
  placeMark(cell, currentClass)
  if (checkWin(currentClass)) {
    endGame(false)
  } else if (isDraw()) {
    endGame(true)
  } else {
    swapTurns()
    setBoardHoverClass()
  }
}

function endGame(draw) {
  if (draw) {
    winningMessageTextElement.innerText = 'Draw!'
  } else {
    winningMessageTextElement.innerText = `${circleTurn ? "O's" : "X's"} Wins!`
  }
  winningMessageElement.classList.add('show')
}

function isDraw() {
  return [...cellElements].every(cell => {
    return cell.classList.contains(X_CLASS) || cell.classList.contains(CIRCLE_CLASS)
  })
}

function placeMark(cell, currentClass) {
  cell.classList.add(currentClass)
}

function swapTurns() {
  circleTurn = !circleTurn
}
```

```
function setBoardHoverClass() {
  board.classList.remove(X_CLASS)
  board.classList.remove(CIRCLE_CLASS)
  if (circleTurn) {
    board.classList.add(CIRCLE_CLASS)
  } else {
    board.classList.add(X_CLASS)
  }
}

function checkWin(currentClass) {
  return WINNING_COMBINATIONS.some(combination => {
    return combination.every(index => {
      return cellElements[index].classList.contains(currentClass)
    })
  })
}
```