Chapter 7 - Encryption\caesar_cipher.html

```html
<!DOCTYPE html>
<html>

<head>
    <title>Caesar Cipher Project</title>
    <style>
        body {
            padding: 15px 0;
            text-align: center;
            background: rgb(91, 222, 255);
            font-family: 'Trebuchet MS', 'Lucida Sans Unicode', 'Lucida Grande', 'Lucida Sans', Arial, sans-serif;
            color: rgb(25, 0, 37)
        }

        .div1 {
            text-align: center;
            font-family: Cambria, Cochin, Georgia, Times, 'Times New Roman', serif;
            font-size: 150%;
            background-color: #0077ff;
            width: 75%;
            border: 5px solid rgb(8, 0, 116);
            padding: 25px;
            margin: auto
        }

        .customEncrypt {
            /* background/text */
            background-color: rgb(196, 75, 75);
            font: Times;
            font-size: 60px;
            color: rgb(23, 239, 192);

            /* border/spacing */
            border-style: dashed;
            border-radius: 10px;
            border-width: 5px;
            border-color: rgb(65, 109, 253);
            padding: 15px;

            /* centering */
            margin: auto;
            width: fit-content;
        }

        .customDecrypt {
            /* background/text */
            background-color: rgb(0, 255, 0);
            font: helvetica;
            font-size: 30px;
            color: rgb(91, 92, 80);

            /* border/spacing */
            border-style: groove;
            border-radius: 20px;
            border-width: 5px;
            border-color: rgb(232, 239, 23);
            padding: 5px;

            /* centering */
            margin: auto;
            width: fit-content;
        }

        .customPinkRandomKey {
            /* background/text */
            background-color: rgb(148, 126, 0);
            font: helvetica;
            font-size: 30px;
            color: rgb(35, 0, 133);

            /* border/spacing */
            border-style: groove;
            border-radius: 20px;
            border-width: 5px;
            border-color: rgb(65, 0, 54);
            padding: 5px;

            /* centering */
            margin: auto;
            width: fit-content;
        }

        .customInstructions {
            /* background/text */
            background-color: royalblue;
            font: helvetica;
            font-size: 40px;
            color: rgb(255, 255, 255);

            /* border/spacing */
            border-style: groove;
            border-radius: 20px;
            border-width: 15px;
            border-color: rgb(239, 59, 23);
            padding: 15px;

            /* centering */
            margin: auto;
            width: fit-content;
        }

        .customForce {
            /* background/text */
            background-color: rgb(0, 85, 33);
            font: helvetica;
            font-size: 40px;
            color: rgb(165, 201, 4);

            /* border/spacing */
            border-style: outset;
            border-radius: 20px;
            border-width: 15px;
            border-color: rgb(196, 0, 121);
            padding: 15px;

            /* centering */
            margin: auto;
            width: fit-content;
        }
    </style>
    <script>
        // Description: Takes a sentence or word and encrypts it with a key
        // Citations: None
        // Input/Parameters: Text input
        // Output/Return: Encrypted message
        function encrypt() {
            let key = idSelectConversion.value;
            let text = textAreaId.value;
            key = Number(key);
            text = text.toUpperCase();
            let textArray = text.split("");
            let output = ""

            for (let i = 0; i < textArray.length; i++) {
                let character = textArray[i]
                character = character.charCodeAt(0)
                if (character < 65 || character > 90) {
                    character = character - key
                }
                else if (character + key > 90 || character + key < 65) {
                    character = character - 26
                }
                character = character + key
                character = String.fromCharCode(character)
                output += character
            }
            outputDiv.innerHTML = output
            outputDiv.style.background = "red";
        }
        // Description: Takes the encrypted message and key and decrypts the message
        // Citations: None
        // Input/Parameters: Encrypted message
        // Output/Return: Decrypted message
        function decrypt() {
            let key = idSelectConversion.value
            let text = textAreaId.value;
            key = Number(key);
            text = text.toLowerCase();
            let textArray = text.split("");
            let output = ""

            for (let i = 0; i < textArray.length; i++) {
                let character = textArray[i]
                character = character.charCodeAt(0)
                if (character < 97 || character > 122) {
                    character = character + key
                }
                else if (character - key > 122 || character - key < 97) {
                    character = character + 26
                }
                character = character - key
                character = String.fromCharCode(character)
                output += character
            }

            outputDiv.innerHTML = output
            outputDiv.style.background = "green";

        }
        // Description: Return a random integer, n, such that min <= n <= max
        // Citation: Mozilla Foundation
        // https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Math/random
        // Input/Parameters: Two floating point numbers, min and max
        // Output/Return: Returns a random integer between min and max
        function randomInt(min, max) {
            min = Math.ceil(min);
            max = Math.floor(max);

            return Math.floor(Math.random() * (max - min + 1)) + min;
        }
        function pickRanKey() {
            ranKey = randomInt(1, 25)
            idSelectConversion.value = ranKey
        }
        // Description: Shows instructions when mouse hovers over them
        // Citations: None
        // Input/Parameters: None
        // Output/Return: Instructions
        function instructions() {
            divInstruct.innerHTML = "Enter a plaintext, select an encryption key, and then press 'Encrypt' to view the ciphertext." + "<br>" +
                "Or you may enter a ciphertext, select a decryption key, and then press 'Decrypt' to view the plaintext"
        }
        // Description: No instructions display when mouse isn't hovering over the button
        // Citations: None
        // Input/Parameters: None
        // Output/Return: Nothing
        function noInstructions() {
            divInstruct.innerHTML = ""
        }
        // Description: Decrypts the message using every single key and displays
        // Citations: None
        // Input/Parameters: Encrypted message
        // Output/Return: 25 different messages using 25 different keys but only one is correct
        function bruteForce() {
            let text = textAreaId.value;
            text = text.toLowerCase();
            let textArray = text.split("")
            let output = ""
            let number = 1
            for (let i = 0; i < 25; i++) {
                for (let i = 0; i < textArray.length; i++) {
                    let character = textArray[i];
                    parseInt(character);
                    character = character.charCodeAt(0)
                    if (character >= 65 && character <= 90) {
                        character = character - number;

                        if (character < 97) {
                            character = character + 26
                        }
                    }
                    else if (character >= 97 && character <= 122) {
                        character = character - number;

                        if (character < 97) {
                            character = character + 26
                        }
                    }
                    else{
                        character = character
                    }
                    character = String.fromCharCode(character)
                    output += character;
                }
                output += "<br>"
                number = number + 1
            }
            outputDiv.innerHTML = output
            outputDiv.style.background = "royalblue";

        }

    </script>
</head>

<body>
    <h1>Caesar Cipher Project</h1>
    <textarea id="textAreaId" rows="10" cols="50"></textarea>
    <br>
    <br>
    <br>
    Key: <select id="idSelectConversion">
        <option>1</option>
        <option>2</option>
        <option>3</option>
        <option>4</option>
        <option>5</option>
        <option>6</option>
        <option>7</option>
        <option>8</option>
        <option>9</option>
        <option>10</option>
        <option>11</option>
```

```html
        <option>12</option>
        <option>13</option>
        <option>14</option>
        <option>15</option>
        <option>16</option>
        <option>17</option>
        <option>18</option>
        <option>19</option>
        <option>20</option>
        <option>21</option>
        <option>22</option>
        <option>23</option>
        <option>24</option>
        <option>25</option>
    </select>
    <br>
    <br>
    <br>
    <span class="customEncrypt" onclick="encrypt()">Encrypt</span>
    <span class="customDecrypt" onclick="decrypt()">Decrypt</span>
    <br><br><br><br><br>
    <span class="customPinkRandomKey" onclick="pickRanKey()">Pick Random Key</span>
    <span class="customInstructions" onmouseover="instructions()" onmouseout="noInstructions()">Instructions</span>
    <br><br><br><br><br>
    <span class="customForce" onclick="bruteForce()">Brute Force Attack</span>
    <br><br><br><br><br>
    <div class="div1" id="outputDiv"></div>
    <div id="divInstruct"></div>
</body>

</html>
```