SPC2207 – Program Documentation

Project: Automated Number System Converter

Date: September 2025

1. Introduction

This program automates conversions between number systems (Decimal, Binary, Octal, Hexadecimal). The purpose is to simplify manual conversions by providing a fast, accurate, and user-friendly tool. JavaScript was chosen as the core programming language because it runs directly in web browsers, requires no additional setup for end users, and integrates seamlessly with HTML and CSS for building interactive interfaces.

2. System Requirements

- Visual Studio Code (VS Code) editor
- Modern web browser (e.g., Chrome, Firefox, Edge)
- Basic knowledge of HTML, CSS, and JavaScript

3. Program Design

The program consists of a simple user interface where the user enters a number, selects the type of conversion from a dropdown menu, and presses a button to execute the conversion. The UI is built with HTML and styled using CSS for clarity. The conversion logic is handled by JavaScript functions that transform input values from one number system to another. The convert() function acts as the main controller, reading user input, determining the conversion type, and invoking the appropriate conversion function.

4. Code Implementation

4.1 HTML Structure

```
<!DOCTYPE html>
<html>
<head>
  <title>Number System Converter</title>
  <link rel="stylesheet" href="style.css">
</head>
<body>
  <h1>Number System Converter</h1>
  <input type="text" id="inputValue" placeholder="Enter number">
  <select id="conversionType">
    <option value="decimalToBinary">Decimal to Binary</option>
    <option value="decimalToOctal">Decimal to Octal</option>
    <option value="decimalToHex">Decimal to Hexadecimal</option>
    <option value="binaryToDecimal">Binary to Decimal</option>
    <option value="octalToDecimal">Octal to Decimal</option>
    <option value="hexToDecimal">Hexadecimal to Decimal</option>
  </select>
  <button onclick="convert()">Convert</button>
  <p id="result"></p>
  <script src="script.js"></script>
</body>
</html>
```

4.2 CSS Styling

```
body {
  font-family: Arial, sans-serif;
  margin: 40px;
  text-align: center;
}
input, select, button {
  margin: 10px;
  padding: 10px;
  font-size: 16px;
}
button {
  background-color: #4CAF50;
  color: white;
  border: none;
  cursor: pointer;
}
button:hover {
  background-color: #45a049;
}
```

4.3 JavaScript Functions

```
function decimalToBinary(num) {
  return (num >>> 0).toString(2);
}

function decimalToOctal(num) {
  return num.toString(8);
}

function decimalToHex(num) {
  return num.toString(16).toUpperCase();
}

function binaryToDecimal(num) {
  return parseInt(num, 2);
}

function octalToDecimal(num) {
  return parseInt(num, 8);
}

function hexToDecimal(num) {
  return parseInt(num, 16);
}

function convert() {
  const value = document.getElementById("inputValue").value;
  const type = document.getElementById("conversionType").value;
  let result = "";

  switch(type) {
    case "decimalToBinary":
      result = decimalToBinary(parseInt(value));
      break;
    case "decimalToOctal":
      result = decimalToOctal(parseInt(value));
      break;
    case "decimalToHex":
      result = decimalToHex(parseInt(value));
      break;
    case "binaryToDecimal":
      result = binaryToDecimal(value);
      break;
    case "octalToDecimal":
      result = octalToDecimal(value);
      break;
    case "hexToDecimal":
      result = hexToDecimal(value);
      break;
    default:
      result = "Invalid conversion";
  }

  document.getElementById("result").innerText = "Result: " + result;
}
```

5. Sample Runs

Example 1: Input = 25, Conversion = Decimal to Binary → Output = 11001 Example 2: Input = 1011, Conversion = Binary to Decimal → Output = 11 Example 3: Input = 64, Conversion = Decimal to Hexadecimal → Output = 40

6. Conclusion

The program successfully performs conversions across decimal, binary, octal, and hexadecimal number systems. It demonstrates the integration of HTML, CSS, and JavaScript in a simple yet effective web-based application. Future improvements include deploying the converter as a mobile application using React Native, enabling Android and iOS support.