Institute of Computer Technology

B. Tech Computer Science and Engineering

Sub: Algorithm Analysis and Design

Practical 10

Huffman coding assigns variable length code words to fixed length input characters based on their frequencies. More frequent characters are assigned shorter code words and less frequent characters are assigned longer code words. All edges along the path to a character contain a code digit. If they are on the left side of the tree, they will be a 0 (zero). If on the right, they'll be a 1 (one). Only the leaves will contain a letter and its frequency count. All other nodes will contain a null instead of a character, and the count of the frequency of all of it and its descendant characters.

Construct the Huffman tree for the following data and obtain its Huffman code.

Characters	A	B	C	D	E	-
Frequency/ Probability	0.5	0.35	0.5	0.1	0.4	0.2

(i) Encode text CAD-BE using the above code.

Input: CAD-BE

Output: 100111001101111100

(ii) Decode the text 1100110110 using the above information.

Input: 0011011100011100

Output: E-DAD

App.py

```
from flask import Flask, render_template, request

import heapq


app = Flask(__name__)


class Node:

    def __init__(self, freq, char=None, left=None, right=None):

        self.freq = freq

        self.char = char

        self.left = left

        self.right = right


    def __lt__(self, other):

        return self.freq < other.freq


# Build the Huffman tree

def build_huffman_tree(char_freq):

    heap = [Node(freq, char) for char, freq in char_freq.items()]

    heapq.heapify(heap)


    while len(heap) > 1:

        left = heapq.heappop(heap)

        right = heapq.heappop(heap)

        merged = Node(left.freq + right.freq, None, left, right)

        heapq.heappush(heap, merged)


    return heap[0]


# Generate Huffman codes

def generate_huffman_codes(node, prefix='', codebook={}):
    if node.char:
        codebook[node.char] = prefix
    else:
        generate_huffman_codes(node.left, prefix + '0', codebook)
        generate_huffman_codes(node.right, prefix + '1', codebook)
    return codebook

# Encode text
def encode(text, codebook):
    return ''.join([codebook[char] for char in text])

# Decode text
def decode(encoded_text, huffman_tree):
    decoded_text = []
    node = huffman_tree
    for bit in encoded_text:
        if bit == '0':
            node = node.left
        else:
            node = node.right
        if node.char:
            decoded_text.append(node.char)
            node = huffman_tree
    return ''.join(decoded_text)

@app.route('/', methods=['GET', 'POST'])
def huffman():
    if request.method == 'POST':
        # Get user input for characters and frequencies
        characters = request.form.getlist('characters')
        frequencies = list(map(float, request.form.getlist('frequencies')))
        char_freq = {characters[i]: frequencies[i] for i in range(len(characters))}


        # Build Huffman Tree and generate Huffman codes

        huffman_tree = build_huffman_tree(char_freq)

        codebook = generate_huffman_codes(huffman_tree)


        # Encode and decode user-specified text

        text_to_encode = request.form['text_to_encode']

        encoded_text = encode(text_to_encode, codebook)

        text_to_decode = request.form['text_to_decode']

        decoded_text = decode(text_to_decode, huffman_tree)


        return render_template('result.html', encoded_text=encoded_text,
decoded_text=decoded_text, codebook=codebook)


    return render_template('index.html')


if __name__ == '__main__':

    app.run(debug=True)
```

Index.html

```
<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>Huffman Encoding and Decoding</title>

    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0-
beta3/css/all.min.css">

    <style>

        body {
```

```
        font-family: Arial, sans-serif;
        background-color: #f4f4f4;
        margin: 0;
        padding: 20px;
    }
    h1, h3 {
        color: #333;
    }
    form {
        background: #fff;
        padding: 20px;
        border-radius: 5px;
        box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
    }
    label {
        display: block;
        margin: 10px 0 5px;
    }
    input[type="text"], input[type="number"] {
        width: calc(100%-20px);
        padding: 10px;
        border: 1px solid #ccc;
        border-radius: 5px;
        margin-bottom: 10px;
    }
    button {
        background: #28a745;
        color: #fff;
        border: none;
        padding: 10px 15px;
        border-radius: 5px;
```

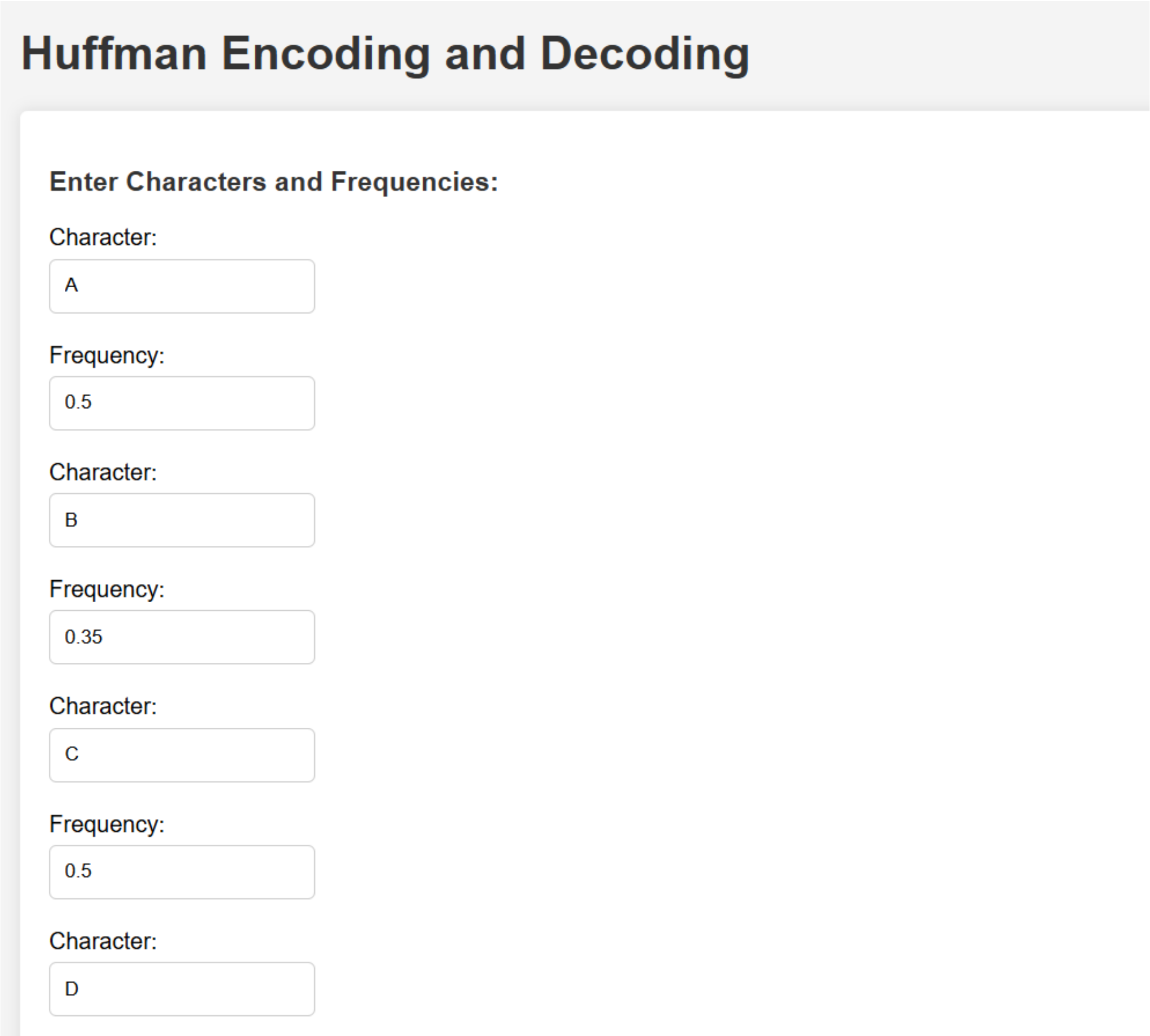
```
            cursor: pointer;

            font-size: 16px;

        }

        button:hover {

            background: #218838;

        }

        .codebook {

            margin-top: 20px;

            padding: 15px;

            border-radius: 5px;

            background: #e9ecef;

        }

        .codebook ul {

            list-style-type: none;

            padding: 0;

        }

        .codebook li {

            padding: 5px 0;

        }

    </style>

</head>

<body>

    <h1>Huffman Encoding and Decoding</h1>

    <form method="POST">

        <h3>Enter Characters and Frequencies:</h3>

        <div id="character-freq-inputs">

            <div>

                <label>Character:</label>

                <input type="text" name="characters" required>

                <label>Frequency:</label>

                <input type="number" step="0.01" name="frequencies" required>
            </div>
        </div>
        <button type="button" onclick="addCharacterInput()">Add More Characters</button>
        <br><br>
        <label for="text_to_encode">Text to Encode (e.g., CAD-BE):</label>
        <input type="text" id="text_to_encode" name="text_to_encode" required>
        <br><br>
        <label for="text_to_decode">Encoded Text to Decode (e.g., 10011100110111100):</label>
        <input type="text" id="text_to_decode" name="text_to_decode" required>
        <br><br>
        <input type="submit" value="Submit">
    </form>

    <script>
        function addCharacterInput() {
            const div = document.createElement('div');
            div.innerHTML = `
                <label>Character:</label>
                <input type="text" name="characters" required>
                <label>Frequency:</label>
                <input type="number" step="0.01" name="frequencies" required>
            `;
            document.getElementById('character-freq-inputs').appendChild(div);
        }
    </script>
</body>
</html>
```

Result.html

```
<!DOCTYPE html>
```

```
<html lang="en">

<head>

   <meta charset="UTF-8">

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

   <title>Huffman Result</title>

   <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0-
beta3/css/all.min.css">

   <style>

      body {

         font-family: Arial, sans-serif;

         background-color: #f4f4f4;

         margin: 0;

         padding: 20px;

      }

      h1, h3 {

         color: #333;

      }

      .result-container {

         background: #fff;

         padding: 20px;

         border-radius: 5px;

         box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);

      }

      .codebook {

         margin-top: 20px;

         padding: 15px;

         border-radius: 5px;

         background: #e9ecef;

      }

      .codebook ul {

         list-style-type: none;
```

```
            padding: 0;
        }
        .codebook li {
            padding: 5px 0;
        }
        .btn-back {
            margin-top: 20px;
            padding: 10px 15px;
            background: #007bff;
            color: #fff;
            border: none;
            border-radius: 5px;
            cursor: pointer;
            font-size: 16px;
        }
        .btn-back:hover {
            background: #0056b3;
        }
    </style>
</head>
<body>
    <h1>Huffman Encoding and Decoding Result</h1>
    <div class="result-container">
        <h3>Huffman Codebook:</h3>
        <div class="codebook">
            <ul>
                {% for char, code in codebook.items() %}
                    <li><strong>{{ char }}:</strong> {{ code }}</li>
                {% endfor %}
            </ul>
        </div>
        <h3>Encoded Text:</h3>

        <p>{{ encoded_text }}</p>

        <h3>Decoded Text:</h3>

        <p>{{ decoded_text }}</p>

        <button class="btn-back" onclick="window.location.href='/'">Go back</button>

    </div>

</body>

</html>
```



Huffman Encoding and Decoding

Enter Characters and Frequencies:

Character: A

Frequency: 0.5

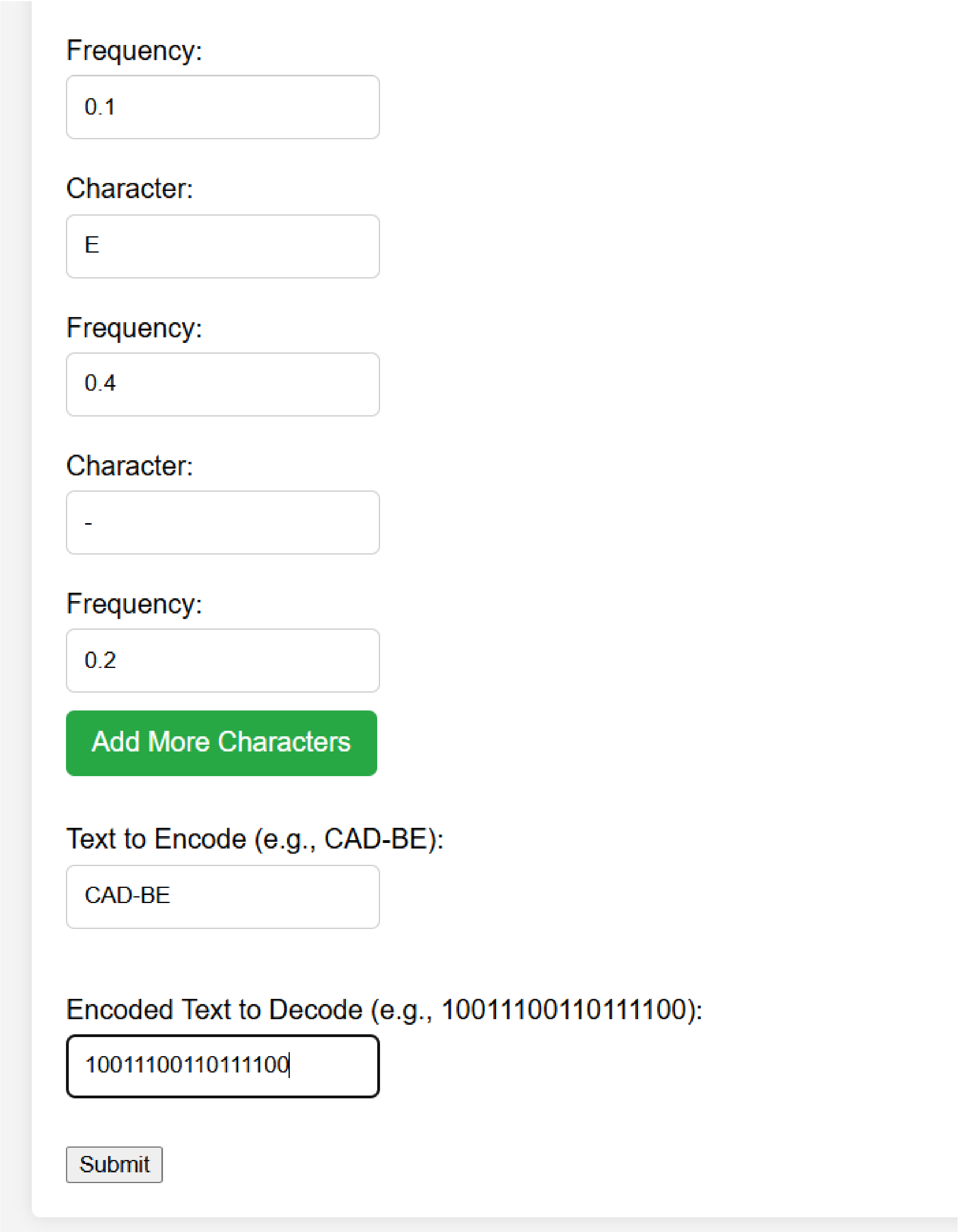
Character: B

Frequency: 0.35

Character: C

Frequency: 0.5

Character: D



Frequency:

0.1

Character:

E

Frequency:

0.4

Character:

-

Frequency:

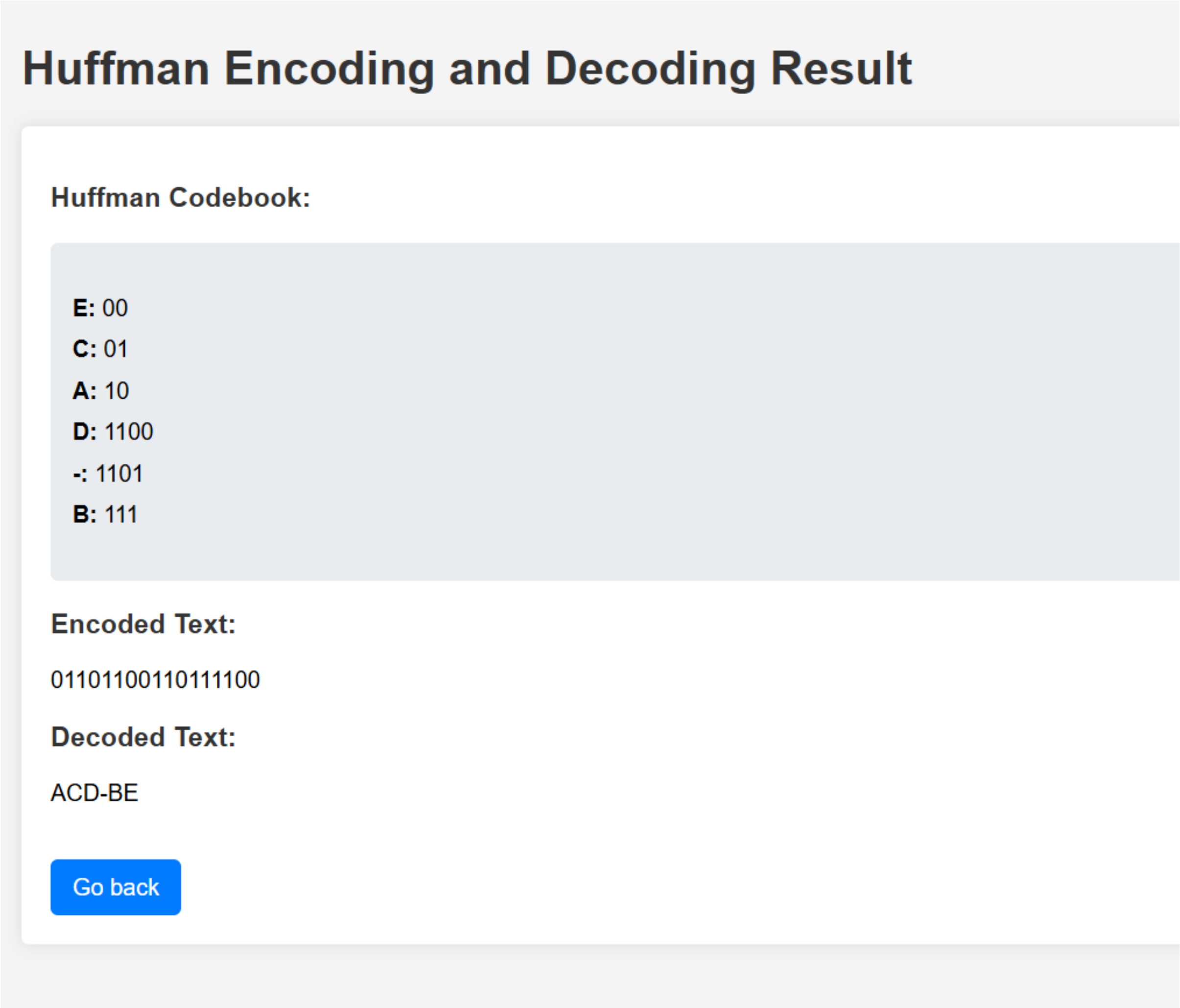
0.2

Text to Encode (e.g., CAD-BE):

CAD-BE

Encoded Text to Decode (e.g., 10011100110111100):

10011100110111100



Huffman Encoding and Decoding Result

Huffman Codebook:

E: 00
C: 01
A: 10
D: 1100
-: 1101
B: 111

Encoded Text:

01101100110111100

Decoded Text:

ACD-BE