Name: Devanshi Jain
Er-No:22162101006
Batch: 51[CBA]

Institute of Computer Technology
B. Tech Computer Science and Engineering

Sub: Algorithm Analysis and Design

Practical 6

Given a sequence of matrices, we want to find the most efficient way to multiply these matrices together to obtain the minimum number of multiplications. The problem is not actually to perform the multiplication of the matrices but to obtain the minimum number of multiplications.

We have many options because matrix multiplication is an associative operation, meaning that the order in which we multiply does not matter. The optimal order depends only on the dimensions of the matrices.

The brute-force algorithm is to consider all possible orders and take the minimum. This is a very inefficient method.

Implement the minimum multiplication algorithm using dynamic programming and determine where to place parentheses to minimize the number of multiplications.

Find an optimal parenthesization of a matrix chain product whose sequence of dimensions are (5, 10, 3, 12, 5, 50, 6).

CODE:

.py:

```
from flask import Flask, render_template, request

app = Flask(__name__)

def matrix_chain_order(dimensions):
    n = len(dimensions) - 1
    m = [[0] * n for _ in range(n)]
    s = [[0] * n for _ in range(n)]

    for length in range(2, n + 1):  # length of the chain
        for i in range(n - length + 1):
            j = i + length - 1
            m[i][j] = float('inf')
            for k in range(i, j):
                q = m[i][k] + m[k + 1][j] + dimensions[i] * dimensions[k + 1] *
dimensions[j + 1]
                if q < m[i][j]:
                    m[i][j] = q
                    s[i][j] = k

    return m, s

def optimal_parenthesization(s, i, j):
    if i == j:
        return f"A{i + 1}"
    else:
        k = s[i][j]
        left = optimal_parenthesization(s, i, k)
        right = optimal_parenthesization(s, k + 1, j)
        return f"({left} x {right})"

@app.route('/', methods=['GET', 'POST'])
def index():
    if request.method == 'POST':
        dimensions = request.form['dimensions']
        try:
            dimensions = list(map(int, dimensions.split(',')))
            m, s = matrix_chain_order(dimensions)
            min_mult = m[0][len(dimensions) - 2]
            parenthesization = optimal_parenthesization(s, 0, len(dimensions) -
2)
```

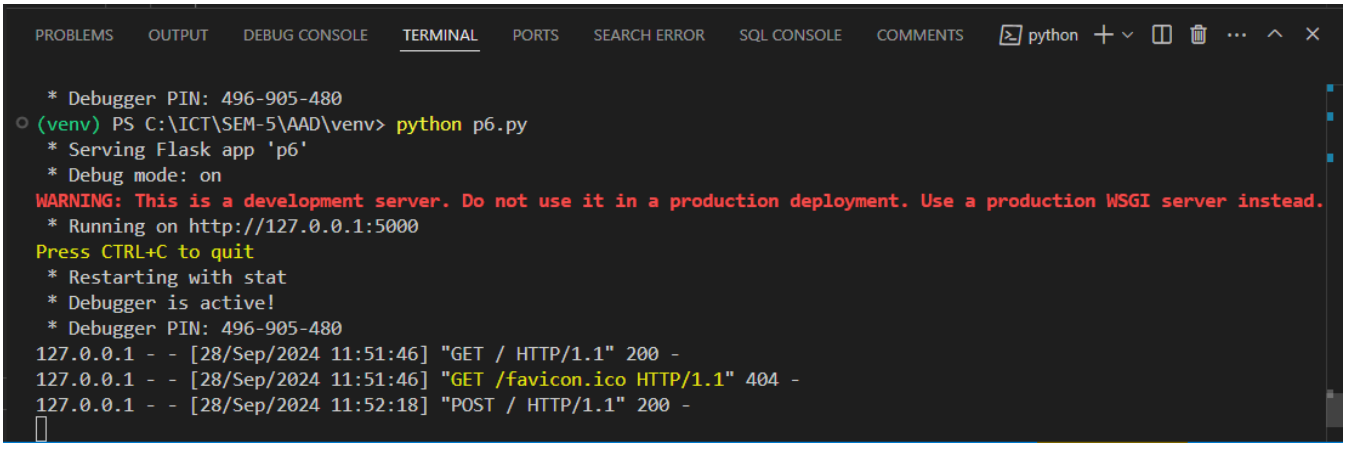
```
            return render_template('p6.html', min_mult=min_mult,
parenthesization=parenthesization, table=m, error=None)
        except ValueError:
            return render_template('p6.html', error="Invalid input. Please enter
integers separated by commas.", min_mult=None, parenthesization=None, table=None)

    return render_template('p6.html', min_mult=None, parenthesization=None,
table=None, error=None)

if __name__ == '__main__':
    app.run(debug=True)
```

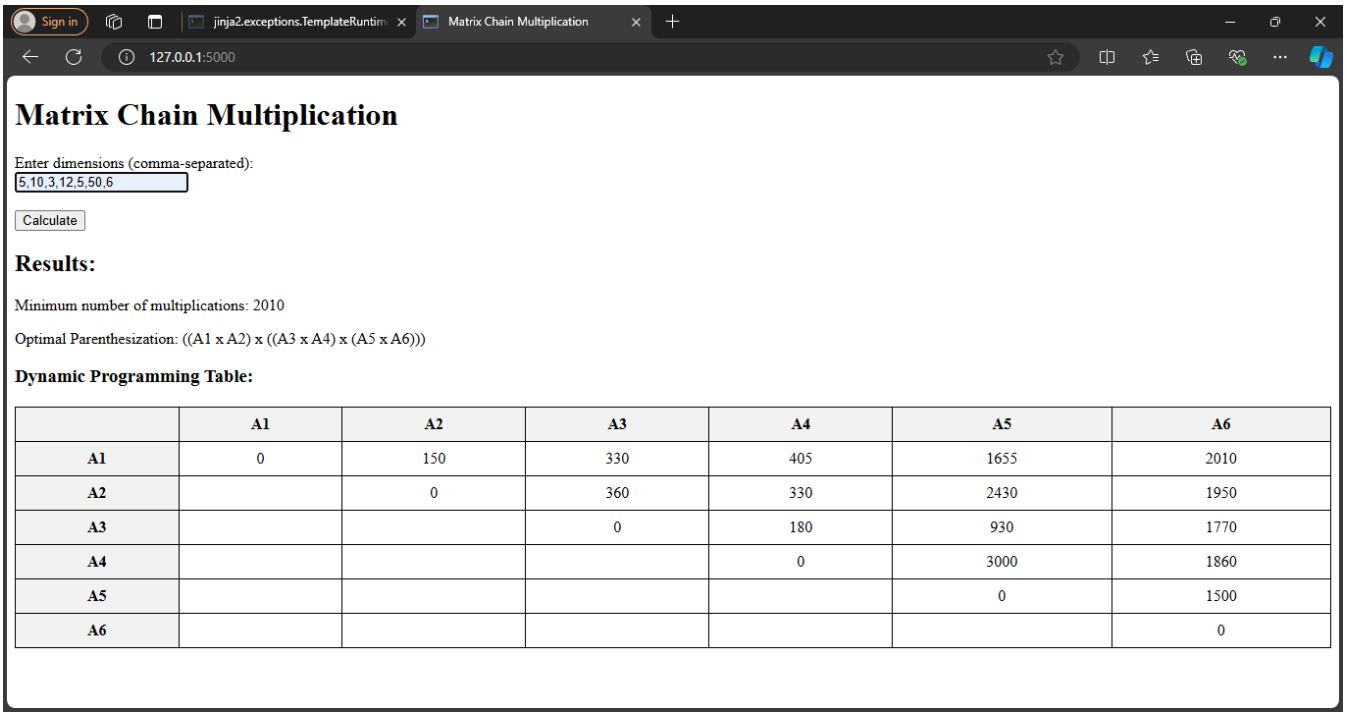
.html:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Matrix Chain Multiplication</title>
    <style>
        table {
            border-collapse: collapse;
            width: 100%;
            margin-top: 20px;
        }
        table, th, td {
            border: 1px solid black;
            padding: 8px;
            text-align: center;
        }
        th {
            background-color: #f2f2f2;
        }
    </style>
</head>
<body>
    <h1>Matrix Chain Multiplication</h1>
    <form method="post">
        <label for="dimensions">Enter dimensions (comma-separated):</label><br>
        <input type="text" id="dimensions" name="dimensions" required><br><br>
        <input type="submit" value="Calculate">
    </form>

    {% if error %}
        <p style="color: red;">{{ error }}</p>
    {% endif %}
```

```
    {% if min_mult is not none %}
        <h2>Results:</h2>
        <p>Minimum number of multiplications: {{ min_mult }}</p>
        <p>Optimal Parenthesization: {{ parenthesization }}</p>

        <h3>Dynamic Programming Table:</h3>
        <table>
            <tr>
                <th></th>
                {% for j in range(table|length) %}
                    <th>A{{ j + 1 }}</th>
                {% endfor %}
            </tr>
            {% for i in range(table|length) %}
                <tr>
                    <th>A{{ i + 1 }}</th>
                    {% for j in range(table|length) %}
                        <td>{{ table[i][j] if j >= i else '' }}</td>
                    {% endfor %}
                </tr>
            {% endfor %}
        </table>
    {% endif %}
</body>
</html>
```

OUTPUT:





	A1	A2	A3	A4	A5	A6
A1	0	150	330	405	1655	2010
A2		0	360	330	2430	1950
A3			0	180	930	1770
A4				0	3000	1860
A5					0	1500
A6						0