NAME :- MARVIN PATEL
ER NO :- 22162101013
SUB :- AAD

PRACTICAL-12

Institute of Computer Technology
B. Tech Computer Science and Engineering

Sub: Algorithm Analysis and Design

Practical 12

“Rocket Singh: Salesman of the Year” is a travelling salesman, who sales good in various cities. One day in the morning, he decided to visit all the cities to sales good and come back to the starting city (from where he has started). Travelling Salesman Problem (TSP) is a touring problem in which n cities and distance between each pair is given. We have to help him to find a shortest route to visit each city exactly once and come back to the starting point.

Sample Input:
[[∞, 20, 30, 10, 11],
[15, ∞, 16, 4, 2],
[3, 5, ∞, 2, 4],
[19, 6, 18, ∞, 3],
[16, 4, 7, 16, ∞]]

Sample Output:
Minimum Path
1 – 4 = 10
4 – 2 = 6
2 – 5 = 2
5 – 3 = 7
3 – 1 = 3

Minimum cost: 28
Path Taken: 1 - 4 - 2 - 5 - 3 - 1

Python code:-

```
from flask import Flask, render_template, request
import itertools


app = Flask(__name__)
```

NAME :- MARVIN PATEL
ER NO :- 22162101013
SUB :- AAD

PRACTICAL-12

```
@app.route("/", methods=["GET", "POST"])
def index():
    output = None
    if request.method == "POST":
        # Get the number of nodes (cities)
        nodes = int(request.form['nodes'])

        # Initialize the distance matrix (input data)
        matrix = []
        for i in range(nodes):
            row = []
            for j in range(nodes):
                key = f"weight_{i}_{j}"
                weight_value = request.form.get(key, "∞")   # Handle missing keys
                row.append(weight_value if weight_value != "∞" else float('inf'))   # Default to inf if missing
            matrix.append(row)

        # Solve the TSP (for now, we'll just simulate a simple path for demonstration)
        path, cost, segments = solve_tsp(matrix)

        output = {
            'input_matrix': matrix,
            'path': path,
            'cost': cost,
            'segments': segments
        }

    return render_template("index.html", output=output)
```

NAME :- MARVIN PATEL
ER NO :- 22162101013
SUB :- AAD

PRACTICAL-12

```python
def solve_tsp(matrix):
    # A simple TSP solver implementation (this is just a placeholder, you can implement the real TSP logic here)
    nodes = len(matrix)
    # Generate all possible permutations of cities
    all_permutations = itertools.permutations(range(nodes))
    min_cost = float('inf')
    best_path = None
    path_segments = []

    # Check each possible path and calculate the cost
    for perm in all_permutations:
        current_cost = 0
        segments = []
        for i in range(nodes - 1):
            current_cost += matrix[perm[i]][perm[i + 1]]
            segments.append(f"City {perm[i] + 1} -> City {perm[i + 1] + 1} = {matrix[perm[i]][perm[i + 1]]}")
        current_cost += matrix[perm[-1]][perm[0]]  # Add the cost to return to the starting city
        segments.append(f"City {perm[-1] + 1} -> City {perm[0] + 1} = {matrix[perm[-1]][perm[0]]}")

        if current_cost < min_cost:
            min_cost = current_cost
            best_path = perm
            path_segments = segments

    # Convert best path from indices to city numbers (1-indexed)
    best_path = " -> ".join([f"City {x + 1}" for x in best_path])
    return best_path, min_cost, ", ".join(path_segments)
```

NAME :- MARVIN PATEL
ER NO :- 22162101013
SUB :- AAD

PRACTICAL-12

```
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>TSP Solver</title>
    <link rel="stylesheet" href="{{ url_for('static',
filename='style.css') }}" />
</head>
<body>
    <h1>Travelling Salesman Problem Solver</h1>

    <div class="container">
        <div class="form-container">
            <form id="nodes-form" method="post">
                <div style="display: flex; align-items:
center;">
                    <label for="nodes" style="margin-right:
10px;">Enter the number of cities:</label>
                    <input type="number" id="nodes" name="nodes"
min="2" required>
                    <button type="button"
onclick="generateMatrixInputs()">Generate Distance
Matrix</button>
                </div>
            </form>
```

NAME :- MARVIN PATEL
ER NO :- 22162101013
SUB :- AAD

PRACTICAL-12

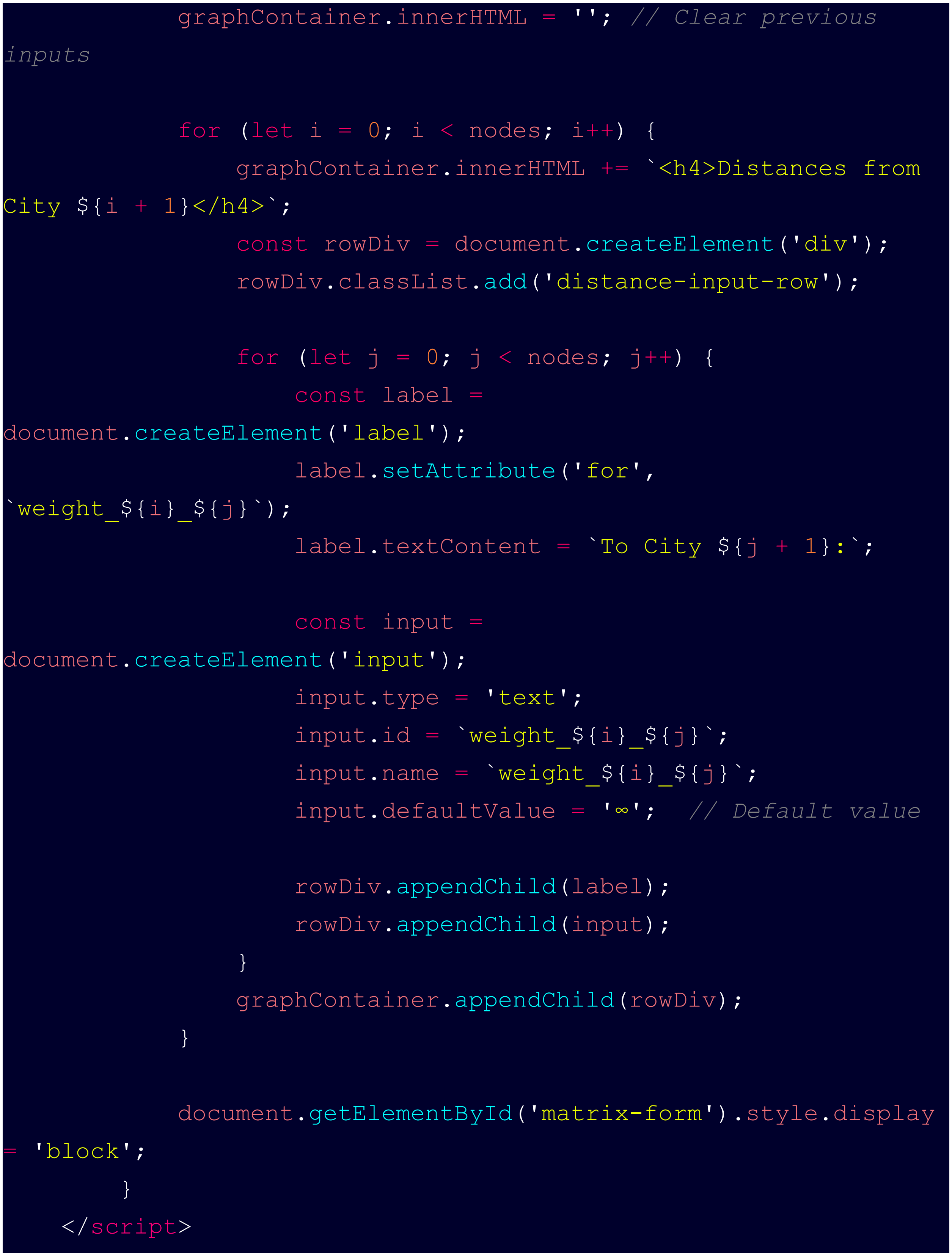
```
            <form id="matrix-form" method="post" style="display: none;">
                <div id="graphContainer"></div>
                <input type="hidden" id="nodes-hidden" name="nodes" value="">
                <input type="submit" value="Submit" class="submit-button">
            </form>
        </div>

        <div class="result-container">
            {% if output %}
                <h2>Input Distance Matrix</h2>
                <table>
                    <tr>
                        <th>From/To</th>
                        {% for j in range(output.input_matrix|length) %}
                            <th>City {{ j + 1 }}</th>
                        {% endfor %}
                    </tr>
                    {% for i in range(output.input_matrix|length) %}
                        <tr>
                            <th>City {{ i + 1 }}</th>
                            {% for j in range(output.input_matrix[i]|length) %}
                                <td>{{ output.input_matrix[i][j] }}</td>
                            {% endfor %}
                        </tr>
                    {% endfor %}
                </table>
```

NAME :- MARVIN PATEL
ER NO :- 22162101013
SUB :- AAD

PRACTICAL-12

```
                <h2>Minimum Path Results</h2>
                <p>Path Taken: {{ output.path }}</p>
                <p>Minimum cost: {{ output.cost }}</p>

                <h3>Path Details:</h3>
                <table style="width: 50%; margin: auto;">
                    <tr>
                        <th>Segment</th>
                        <th>Cost</th>
                    </tr>
                    {% for segment in output.segments.split(',
') %}
                        <tr>
                            <td>{{ segment.split(' = ')[0]
}}</td>
                            <td>{{ segment.split(' = ')[1]
}}</td>
                        </tr>
                    {% endfor %}
                </table>
            {% endif %}
        </div>
    </div>

    <script>
        function generateMatrixInputs() {
            const nodes =
document.getElementById('nodes').value;
            document.getElementById('nodes-hidden').value =
nodes;
            const graphContainer =
document.getElementById('graphContainer');
```

PRACTICAL-12

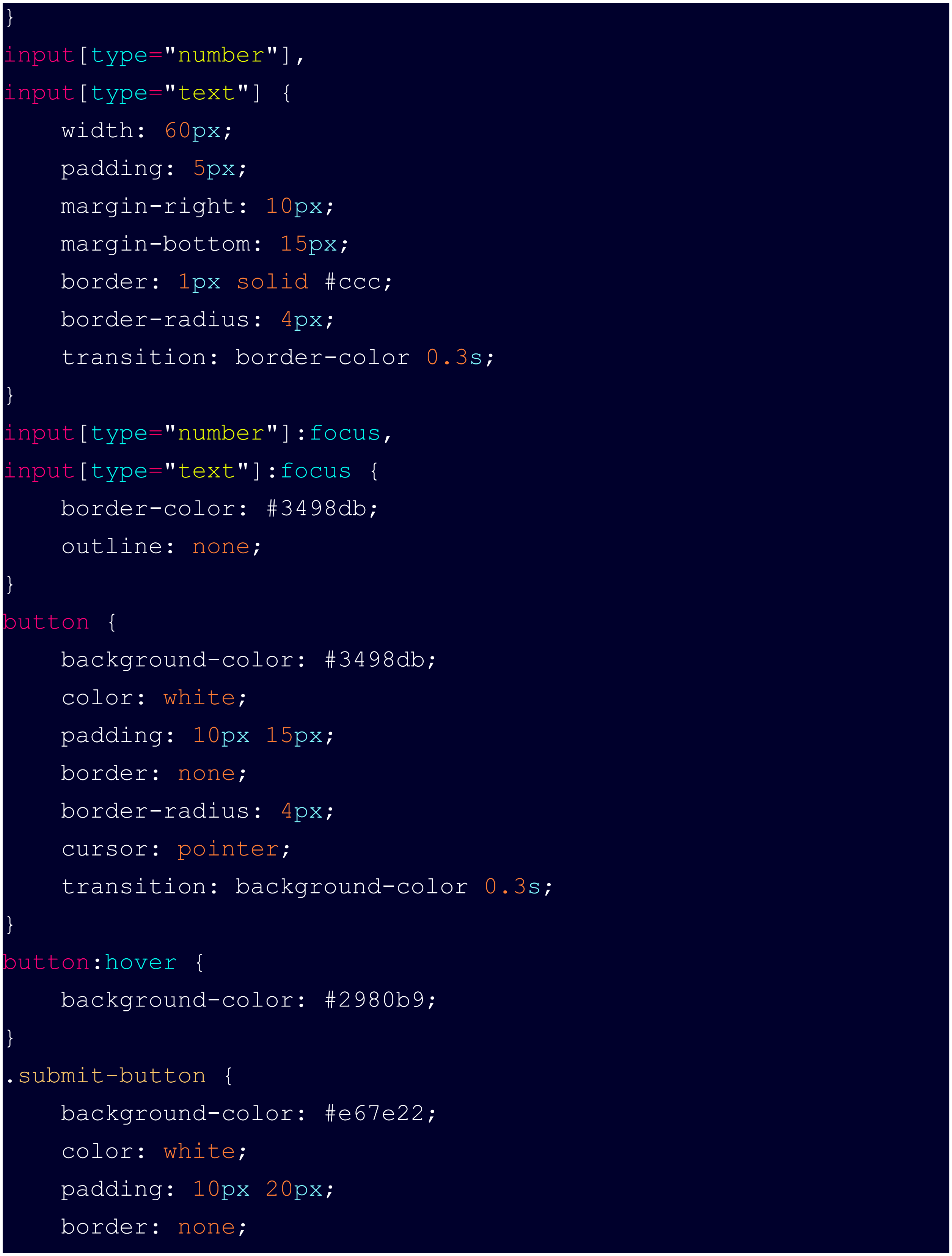


```
            graphContainer.innerHTML = ''; // Clear previous inputs

            for (let i = 0; i < nodes; i++) {
                graphContainer.innerHTML += `<h4>Distances from City ${i + 1}</h4>`;
                const rowDiv = document.createElement('div');
                rowDiv.classList.add('distance-input-row');

                for (let j = 0; j < nodes; j++) {
                    const label = document.createElement('label');
                    label.setAttribute('for', `weight_${i}_${j}`);
                    label.textContent = `To City ${j + 1}:`;

                    const input = document.createElement('input');
                    input.type = 'text';
                    input.id = `weight_${i}_${j}`;
                    input.name = `weight_${i}_${j}`;
                    input.defaultValue = '∞';  // Default value

                    rowDiv.appendChild(label);
                    rowDiv.appendChild(input);
                }
                graphContainer.appendChild(rowDiv);
            }

            document.getElementById('matrix-form').style.display = 'block';
        }
    </script>
```

```
</body>
</html>
```

Style.css

```
body {
    font-family: Arial, sans-serif;
    background-color: #f4f4f4;
    color: #333;
    margin: 0;
    padding: 20px;
}
h1, h2, h3 {
    color: #2c3e50;
}
.container {
    display: flex;
    flex-wrap: wrap;
    justify-content: space-between;
}
.form-container, .result-container {
    background: #ffffff;
    padding: 20px;
    border-radius: 8px;
    box-shadow: 0 2px 5px rgba(0, 0, 0, 0.1);
    margin: 0 10px;
    flex: 1;
    min-width: 300px;
    max-width: 48%;
}
label {
    display: block;
    margin-bottom: 8px;
    font-weight: bold;
```



```
}
input[type="number"],
input[type="text"] {
    width: 60px;
    padding: 5px;
    margin-right: 10px;
    margin-bottom: 15px;
    border: 1px solid #ccc;
    border-radius: 4px;
    transition: border-color 0.3s;
}
input[type="number"]:focus,
input[type="text"]:focus {
    border-color: #3498db;
    outline: none;
}
button {
    background-color: #3498db;
    color: white;
    padding: 10px 15px;
    border: none;
    border-radius: 4px;
    cursor: pointer;
    transition: background-color 0.3s;
}
button:hover {
    background-color: #2980b9;
}
.submit-button {
    background-color: #e67e22;
    color: white;
    padding: 10px 20px;
    border: none;
```

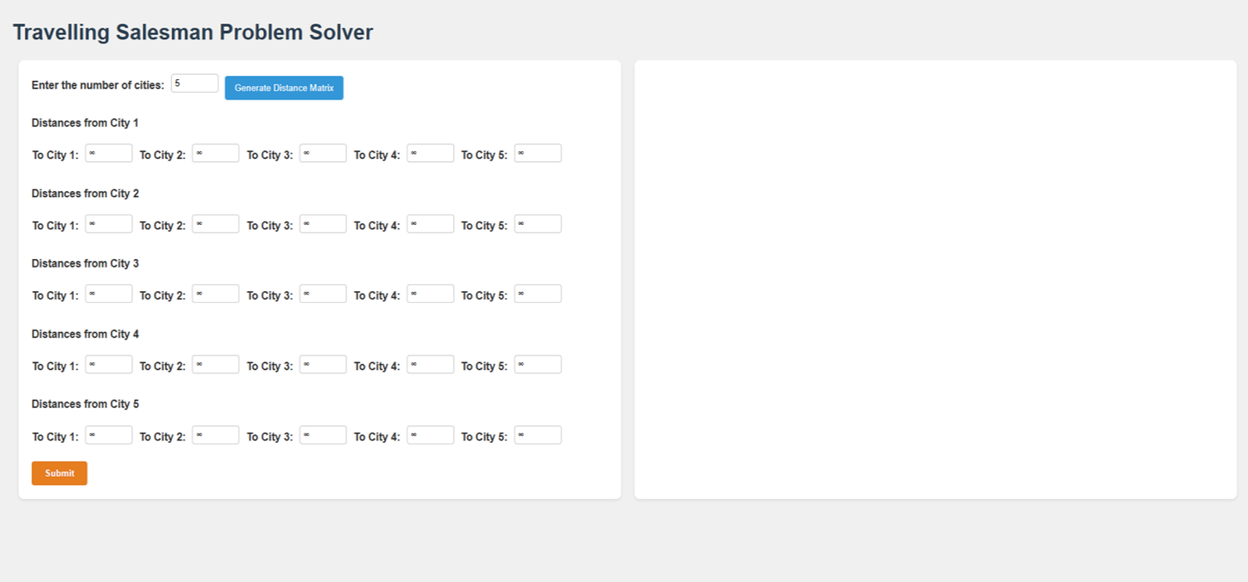
PRACTICAL-12

```
    border-radius: 4px;
    cursor: pointer;
    transition: background-color 0.3s;
    font-weight: bold;
}
.submit-button:hover {
    background-color: #d35400;
}
table {
    border-collapse: collapse;
    width: 100%;
    margin: 20px 0;
    background-color: #ffffff;
}
table, th, td {
    border: 1px solid #ddd;
}
th, td {
    padding: 12px;
    text-align: center;
}
th {
    background-color: #3498db;
    color: white;
}
tr:nth-child(even) {
    background-color: #f2f2f2;
}
tr:hover {
    background-color: #d1e7fd;
}
.input-row {
    display: flex;
    align-items: center;
    margin-bottom: 15px;
}
.city-names, .distance-inputs {
    display: flex;
    justify-content: flex-start;
    flex-wrap: wrap;
    margin-bottom: 15px;
}
.distance-input-row {
    display: flex;
    align-items: center;
    margin-bottom: 10px;
}
.distance-input-row label {
    margin-right: 10px;
}
```

Output:-

NAME :- MARVIN PATEL
ER NO :- 22162101013
SUB :- AAD

PRACTICAL-12



Travelling Salesman Problem Solver

Enter the number of cities: 5

Distances from City 1

To City 1: ∞ To City 2: ∞ To City 3: ∞ To City 4: ∞ To City 5: ∞

Distances from City 2

To City 1: ∞ To City 2: ∞ To City 3: ∞ To City 4: ∞ To City 5: ∞

Distances from City 3

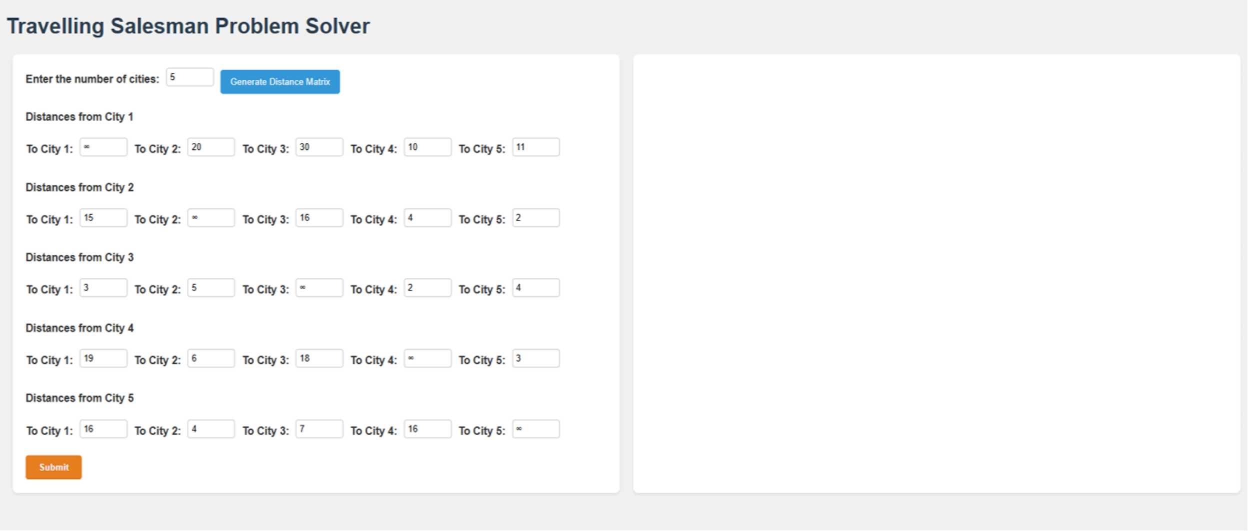
To City 1: ∞ To City 2: ∞ To City 3: ∞ To City 4: ∞ To City 5: ∞

Distances from City 4

To City 1: ∞ To City 2: ∞ To City 3: ∞ To City 4: ∞ To City 5: ∞

Distances from City 5

To City 1: ∞ To City 2: ∞ To City 3: ∞ To City 4: ∞ To City 5: ∞



Travelling Salesman Problem Solver

Enter the number of cities: 5

Distances from City 1

To City 1: ∞ To City 2: 20 To City 3: 30 To City 4: 10 To City 5: 11

Distances from City 2

To City 1: 15 To City 2: ∞ To City 3: 16 To City 4: 4 To City 5: 2

Distances from City 3

To City 1: 3 To City 2: 5 To City 3: ∞ To City 4: 2 To City 5: 4

Distances from City 4

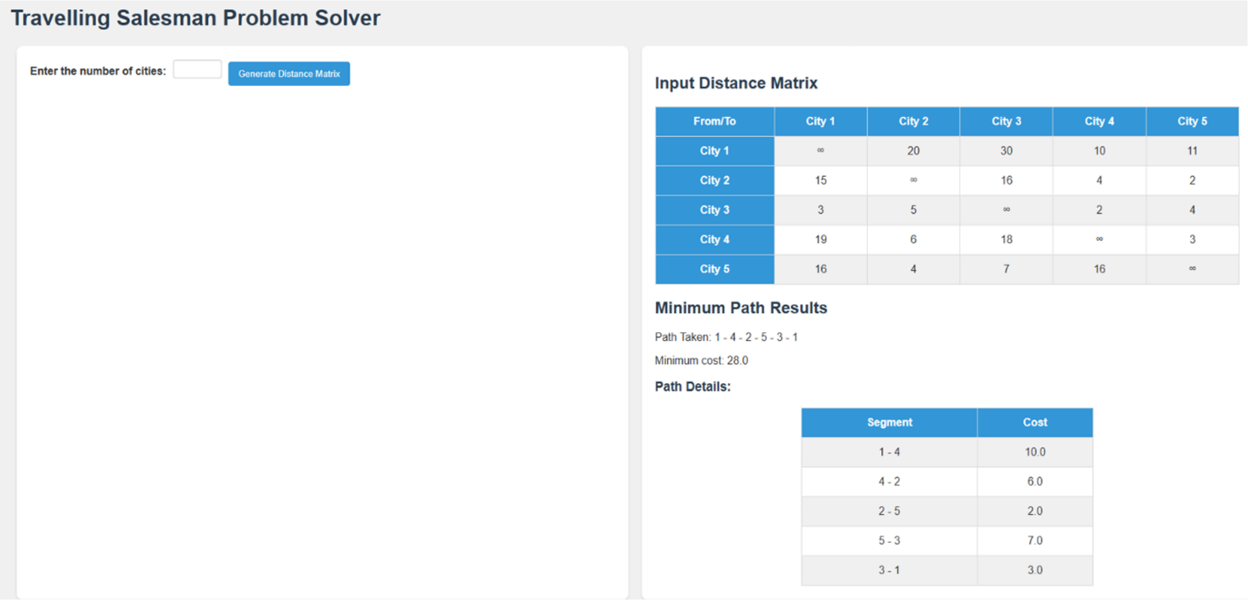
To City 1: 19 To City 2: 6 To City 3: 18 To City 4: ∞ To City 5: 3

Distances from City 5

To City 1: 16 To City 2: 4 To City 3: 7 To City 4: 16 To City 5: ∞

NAME :- MARVIN PATEL
ER NO :- 22162101013
SUB :- AAD

PRACTICAL-12



Travelling Salesman Problem Solver

Input Distance Matrix

From/To	City 1	City 2	City 3	City 4	City 5
City 1	∞	20	30	10	11
City 2	15	∞	16	4	2
City 3	3	5	∞	2	4
City 4	19	6	18	∞	3
City 5	16	4	7	16	∞

Minimum Path Results

Path Taken: 1 - 4 - 2 - 5 - 3 - 1

Minimum cost: 28.0

Path Details:

Segment	Cost
1 - 4	10.0
4 - 2	6.0
2 - 5	2.0
5 - 3	7.0
3 - 1	3.0