Mandar Vivek Amte
A5-B2-27
DAA-Lab

Practical-6

TASK 1:

Code:

```
def OptimalBST(p, q, n):
    E = [[0] * (n + 2) for _ in range(n + 2)]
    W = [[0] * (n + 2) for _ in range(n + 2)]
    R = [[0] * (n + 1) for _ in range(n + 1)]

    for i in range(n + 1):
        E[i + 1][i] = q[i]
        W[i + 1][i] = q[i]

    for l in range(1, n + 1):
        for i in range(1, n - l + 2):
            j = i + l - 1
            W[i][j] = W[i][j - 1] + p[j - 1] + q[j]
            E[i][j] = float('inf')
            for k in range(i, j + 1):
                cost = E[i][k - 1] + E[k + 1][j] + W[i][j]
                if cost < E[i][j]:
                    E[i][j] = cost
                    R[i][j] = k

    return E, W, R

p_input = [0.15, 0.10, 0.05, 0.10, 0.20]
q_input = [0.10, 0.05, 0.05, 0.10, 0.15, 0.10]
n = len(p_input)

E, W, R = OptimalBST(p_input, q_input, n)
```

Mandar Vivek Amte
A5-B2-27
DAA-Lab

```
def print_matrix(matrix, n, decimals=2):
    for r in range(1, n + 2):
        formatted_row = [round(matrix[r][c], decimals) for c in range(1, n + 2)]
        print(formatted_row)


print("Expected Cost Matrix (E):")
print_matrix(E, n)


print("\nWeight Matrix (W):")
print_matrix(W, n)


print("\nRoot Matrix (R):")
for r in range(1, n + 1):
    print(R[r][1:n + 1])
```

OUTPUT:

```
Expected Cost Matrix (E):
[0.45, 0.85, 1.4, 2.25, 3.25, 0]
[0.05, 0.3, 0.75, 1.5, 2.35, 0]
[0, 0.05, 0.35, 0.95, 1.8, 0]
[0, 0, 0.1, 0.6, 1.35, 0]
[0, 0, 0, 0.15, 0.7, 0]
[0, 0, 0, 0, 0.1, 0]

Weight Matrix (W):
[0.3, 0.45, 0.6, 0.85, 1.15, 0]
[0.05, 0.2, 0.35, 0.6, 0.9, 0]
[0, 0.05, 0.2, 0.45, 0.75, 0]
[0, 0, 0.1, 0.35, 0.65, 0]
[0, 0, 0, 0.15, 0.45, 0]
[0, 0, 0, 0, 0.1, 0]

Root Matrix (R):
[1, 1, 2, 2, 4]
[0, 2, 2, 3, 4]
[0, 0, 3, 4, 4]
[0, 0, 0, 4, 5]
[0, 0, 0, 0, 5]
```

Mandar Vivek Amte
A5-B2-27
DAA-Lab

TASK 2: