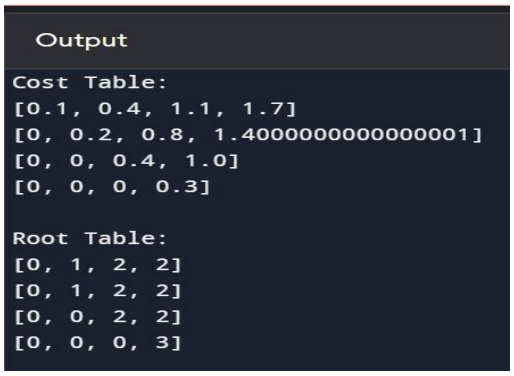
186. Implement the Optimal Binary Search Tree algorithm for the keys A,B,C,D with frequencies 0.1,0.2,0.4,0.3 Write the code using any programming language to construct the OBST for the given keys and frequencies. Execute your code and display the resulting OBST and its cost. Print the cost and root matrix.

Program:

```
def optimal_bst(keys, freq):
    n = len(keys)
    cost = [[0 for _ in range(n)] for _ in range(n)]
    root = [[0 for _ in range(n)] for _ in range(n)]

    for i in range(n):
        cost[i][i] = freq[i]
        root[i][i] = i

    for L in range(2, n + 1):
        for i in range(n - L + 1):
            j = i + L - 1
            cost[i][j] = float('inf')
            for r in range(i, j + 1):
                c = cost[i][r - 1] if r > i else 0
                c += cost[r + 1][j] if r < j else 0
                c += sum(freq[i:j + 1])
                if c < cost[i][j]:
                    cost[i][j] = c
                    root[i][j] = r

    return cost, root

keys = ['A', 'B', 'C', 'D']
freq = [0.1, 0.2, 0.4, 0.3]
cost_table, root_table = optimal_bst(keys, freq)

print("Cost Table:")
```

```
for row in cost_table:
    print(row)


print("\nRoot Table:")
for row in root_table:
    print(row)
```

Output:



Timecomplexity:O(n^3)