Practical 6 (DAA)

Name : Mohammad Ejaz Mehendi

Section : A3-B3

Roll no : 48

Aim: Construction of OBST

Problem Statement: Smart Library Search Optimization

Task 1:

Scenario:

A university digital library system stores frequently accessed books using a binary search

mechanism. The library admin wants to minimize the average search time for book lookups by

arranging the book IDs optimally in a binary search tree.

Each book ID has a probability of being searched successfully and an associated probability for

unsuccessful searches (when a book ID does not exist between two keys).

Your task is to determine the minimum expected cost of searching using an Optimal Binary

Search Tree (OBST).

Code :

#include <stdio.h>

#include <float.h>

#define MAX 20

float OptimalBST(float p[], float q[], int n) {

float e[MAX][MAX], w[MAX][MAX];

int root[MAX][MAX];

int i, j, k, d;

float cost;

for (i = 0; i <= n; i++) {

e[i][i] = q[i];

w[i][i] = q[i];

root[i][i] = 0;

}

for (d = 1; d <= n; d++) {

for (i = 0; i <= n - d; i++) {

j = i + d;

e[i][j] = FLT\_MAX;

w[i][j] = w[i][j - 1] + p[j] + q[j];

for (k = i + 1; k <= j; k++) {

cost = e[i][k - 1] + e[k][j] + w[i][j];

if (cost < e[i][j]) {

e[i][j] = cost;

root[i][j] = k;

}

}

}

}

return e[0][n];

}

int main() {

int n = 4;

int keys[] = {0, 10, 20, 30, 40};

float p[] = {0, 0.1, 0.2, 0.4, 0.3};

float q[] = {0.05, 0.1, 0.05, 0.05, 0.1};

float minCost = OptimalBST(p, q, n);

printf("Keys: 10 20 30 40\n");

printf("P[i]: 0.1 0.2 0.4 0.3\n");

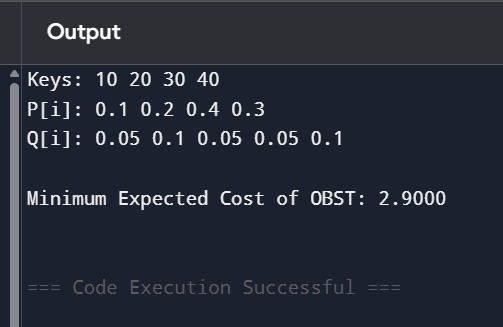
printf("Q[i]: 0.05 0.1 0.05 0.05 0.1\n\n");

printf("Minimum Expected Cost of OBST: %.4f\n", minCost);

return 0;

}

Output :



Task 2:

<https://www.geeksforgeeks.org/problems/optimal-binary-search-tree2214/1>

Output :

