#include<iostream>

using namespace std;

void construct\_obst();

void print(int, int);

float a[20], b[20], wt[20][20], cost[20][20];

int root[20][20], n;

int main() {

int i;

cout << "\n\*\*\*\*\*\* PROGRAM FOR OBST \*\*\*\*\*\*\n";

cout << "\nEnter the number of nodes : ";

cin >> n;

cout << "\nEnter the probability for successful search :\n";

for (i = 1; i <= n; i++)

cin >> a[i];

cout << "\nEnter the probability for unsuccessful search :\n";

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

cin >> b[i];

construct\_obst();

print(0, n);

cout << endl;

return 0;

}

void construct\_obst() {

int i, j, k, l;

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

cost[i][i] = 0.0;

root[i][i] = 0;

wt[i][i] = b[i];

}

cost[n][n] = 0.0;

root[n][n] = 0;

wt[n][n] = b[n];

for (i = 2; i <= n; i++) {

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

wt[j][j + i] = b[j + i] + a[j + i] + wt[j][j + i - 1];

cost[j][j + i] = 9999;

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

if (cost[j][j + i] > (cost[j][l - 1] + cost[l][j + i])) {

cost[j][j + i] = cost[j][l - 1] + cost[l][j + i];

root[j][j + i] = l;

}

}

cost[j][j + i] += wt[j][j + i];

}

}

cout << "\n\nOptimal BST is :: ";

cout << "\nTotal Weight: " << wt[0][n];

cout << "\nTotal Cost: " << cost[0][n];

cout << "\nRoot: " << root[0][n];

}

void print(int l1, int r1) {

if (l1 >= r1)

return;

if (root[l1][root[l1][r1] - 1] != 0)

cout << "\nLeft child of " << root[l1][r1] << " :: " << root[l1][root[l1][r1] - 1];

if (root[root[l1][r1]][r1] != 0)

cout << "\nRight child of " << root[l1][r1] << " :: " << root[root[l1][r1]][r1];

print(l1, root[l1][r1] - 1);

print(root[l1][r1], r1);

}