```
#include <iostream>
#include <limits.h>
using namespace std;
#define SIZE 15
class OBST {
int prob[SIZE] = {};
int keys[SIZE] = {};
int weight[SIZE][SIZE] = {};
int cost[SIZE][SIZE] = {};
int root[SIZE][SIZE] = {};
int n;
public:
void get_data();
int Min_Value(int, int);
void build_OBST();
void build_tree();
void print(int [][SIZE], int);
};
void OBST::get_data() {
int i;
cout << "\nOptimal Binary Search Tree \n\nEnter the number of nodes: ";
cin >> n;
cout << "\nEnter " << n << " nodes: ";
for (i = 1; i <= n; i++)
cin >> keys[i];
cout << "\nEnter " << n << " probabilities: ";
for (i = 1; i <= n; i++)
cin >> prob[i];
```

```
}
int OBST::Min_Value(int i, int j) {
int l, k;
int minimum = INT_MAX;
for (l = root[i][j - 1]; l <= root[i + 1][j]; l++) {
if ((cost[i][l - 1] + cost[l][j]) < minimum) {
minimum = cost[i][l - 1] + cost[l][j];
k = l;
}
}

return k;
}
void OBST::build_OBST() {
int i, j, k, l;
for (i = 0; i < n; i++) {
weight[i][i] = root[i][i] = cost[i][i] = 0;
weight[i][i + 1] = cost[i][i + 1] = prob[i + 1];
root[i][i + 1] = i + 1;
}
weight[n][n] = root[n][n] = cost[n][n] = 0;
for (l = 2; l <= n; l++) {
for (i = 0; i <= n - l; i++) {
j = i + l;
weight[i][j] = weight[i][j - 1] + prob[j];
k = Min_Value(i, j);
cost[i][j] = weight[i][j] + cost[i][k - 1] + cost[k][j];
root[i][j] = k;
```

```
}

}

cout << "\nCost are: \n";

print(cost, n);

cout << "\nRoot are: \n";

print(root, n);

}

void OBST::build_tree() {

int i, j, k;

int queue[20], front = -1, rear = -1;

cout << "\nThe Optimal Binary Search Tree For the Given Nodes Is...\n";

cout << "\nThe Root of this OBST is:: " << keys[root[0][n]];

cout << "\nThe Cost of this OBST is:: " << cost[0][n];

cout << "\n\n\tNODE\tLEFT CHILD\tRIGHT CHILD\n";

queue[++rear] = 0;

queue[++rear] = n;

while (front != rear) {

i = queue[++front];

j = queue[++front];

k = root[i][j];

cout << "\n\t" << keys[k];

if (root[i][k - 1] != 0) {

cout << "\t\t" << keys[root[i][k - 1]];

queue[++rear] = i;


queue[++rear] = k - 1;

} else cout << "\t\t";

if (root[k][j] != 0) {
```

```
cout << "\t" << keys[root[k][j]];

queue[++rear] = k;

queue[++rear] = j;

} else cout << "\t";

}

cout << "\n";

}

void OBST::print(int arr[][SIZE], int n) {

int i, j;

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

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

cout << arr[i][j] << '\t';

cout << '\n';

}

}

int main() {

OBST obj;

obj.get_data();

obj.build_OBST();

obj.build_tree();

return 0;

}
```