# Assignment no 8

Problem statement: Given sequence k = k1 <k2 < … <kn of n sorted keys, with a search probability pi for each key ki . Build the Binary search tree that has the least search cost given the access probability for each key?

```
#include <iostream>
#include <vector>
#include <limits>
using namespace std;

double sum(const vector<double>& p, int i, int j) {
    double s = 0;
    for (int k = i; k <= j; k++) {
        s += p[k];
    }
    return s;
}

double optimalBST(const vector<int>& keys, const vector<double>& p, int n) {
    vector<vector<double>> cost(n, vector<double>(n, 0));


    for (int i = 0; i < n; i++) {
        cost[i][i] = p[i];
    }


    for (int L = 2; L <= n; L++) {
        for (int i = 0; i <= n - L; i++) {
            int j = i + L - 1;
            cost[i][j] = numeric_limits<double>::max();
```

```
            for (int r = i; r <= j; r++) {
                double leftCost = (r > i) ? cost[i][r - 1] : 0;
                double rightCost = (r < j) ? cost[r + 1][j] : 0;
                double totalCost = leftCost + rightCost + sum(p, i, j);

                if (totalCost < cost[i][j]) {
                    cost[i][j] = totalCost;
                }
            }
        }
    }
    return cost[0][n - 1];
}

int main() {
    vector<int> keys = {10, 20, 30, 40};
    vector<double> p = {0.1, 0.2, 0.4, 0.3};

    int n = keys.size();
    double minCost = optimalBST(keys, p, n);

    cout << "Minimum Cost of Optimal BST: " << minCost << endl;
    return 0;
}
```

Output:

Minimum Cost of Optimal BST: 1.7