// Sakshi Yadav (79)
// PR_8

//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>
using namespace std;

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

float optimalBST(const vector<float>& p, int n) {
   vector<vector<float>> cost(n, vector<float>(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] = 1e9;

         for (int r = i; r <= j; r++) {
            float left = (r > i) ? cost[i][r - 1] : 0;
            float right = (r < j) ? cost[r + 1][j] : 0;
            float total = left + right + sum(p, i, j);

            if (total < cost[i][j])
               cost[i][j] = total;
         }
      }
   }

   return cost[0][n - 1];
}
```

```
int main() {
    int n;
    cout << "Enter number of keys: ";
    cin >> n;

    vector<int> keys(n);
    vector<float> probs(n);

    cout << "Enter sorted keys:\n";
    for (int i = 0; i < n; i++)
        cin >> keys[i];

    cout << "Enter search probabilities:\n";
    for (int i = 0; i < n; i++)
        cin >> probs[i];

    float minCost = optimalBST(probs, n);
    cout << "\nMinimum cost of Optimal BST: " << minCost << endl;

    return 0;
}
```