Name: Nachiket Jagdale

Class: A3-B1

Roll no.-02

PRACTICAL NO. 6

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).

Input Format

First line: integer n — number of book IDs.

Second line: n integers representing the sorted book IDs (keys).

Third line: n real numbers — probabilities of successful searches (p[i]).

Fourth line: n+1 real numbers — probabilities of unsuccessful searches (q[i]).

Keys: 10 20 30 40

P[i]: 0.1 0.2 0.4 0.3

Q[i]: 0.05 0.1 0.05 0.05 0.1

Output Format

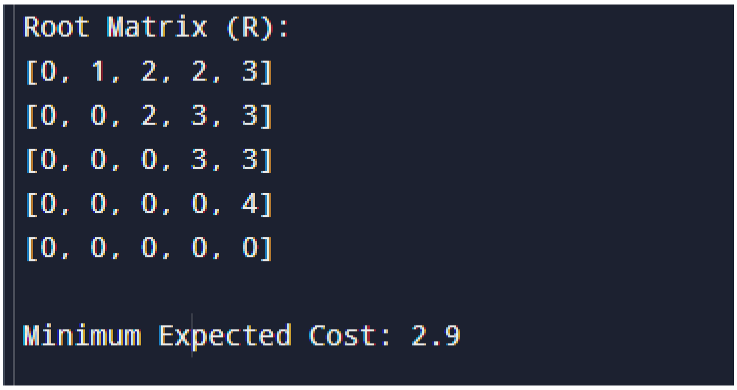
Print the minimum expected cost of the Optimal Binary Search Tree, rounded to 4 decimal places.

CODE:

```
#include <stdio.h>
#include <float.h>
#define MAX 100
void OptimalBST(float p[],float q[],int n,float E[MAX+1][MAX+1],float W[MAX+1][MAX+1],int R[MAX+1][MAX+1]){
 int i,j,k,d;float cost;
 for(i=0;i<=n;i++){E[i][i]=q[i];W[i][i]=q[i];R[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;R[i][j]=k;}
   }
  }
 }
}
```

```
int main(){
 int n,i;float p[MAX],q[MAX];float E[MAX+1][MAX+1],W[MAX+1][MAX+1];int R[MAX+1][MAX+1];int keys[MAX];
 printf("Enter number of book IDs: ");scanf("%d",&n);
 printf("Enter sorted book IDs:\n");for(i=1;i<=n;i++)scanf("%d",&keys[i]);
 printf("Enter probabilities of successful searches:\n");for(i=1;i<=n;i++)scanf("%f",&p[i]);
 printf("Enter probabilities of unsuccessful searches:\n");for(i=0;i<=n;i++)scanf("%f",&q[i]);
 OptimalBST(p,q,n,E,W,R);
 printf("\nMinimum expected cost of OBST: %.4f\n",E[0][n]);
 return 0;
}
```

OUTPUT:



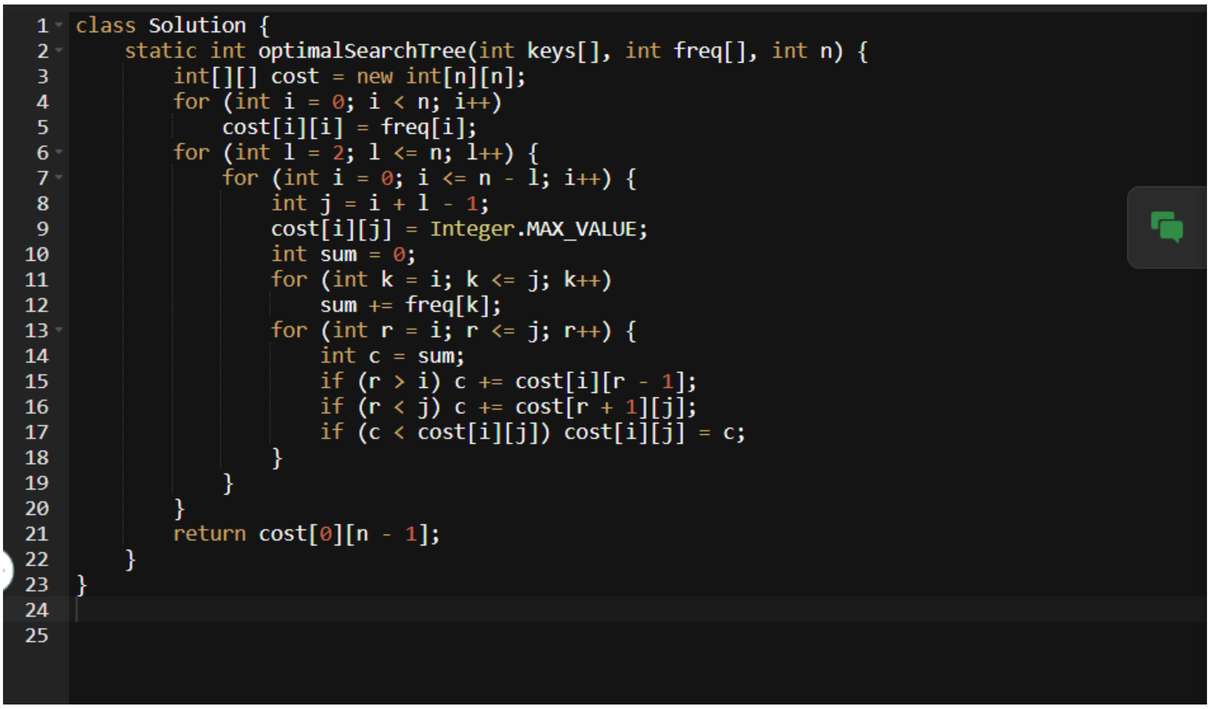
```
Root Matrix (R):
[0, 1, 2, 2, 3]
[0, 0, 2, 3, 3]
[0, 0, 0, 3, 3]
[0, 0, 0, 0, 4]
[0, 0, 0, 0, 0]

Minimum Expected Cost: 2.9
```

Task 2:

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

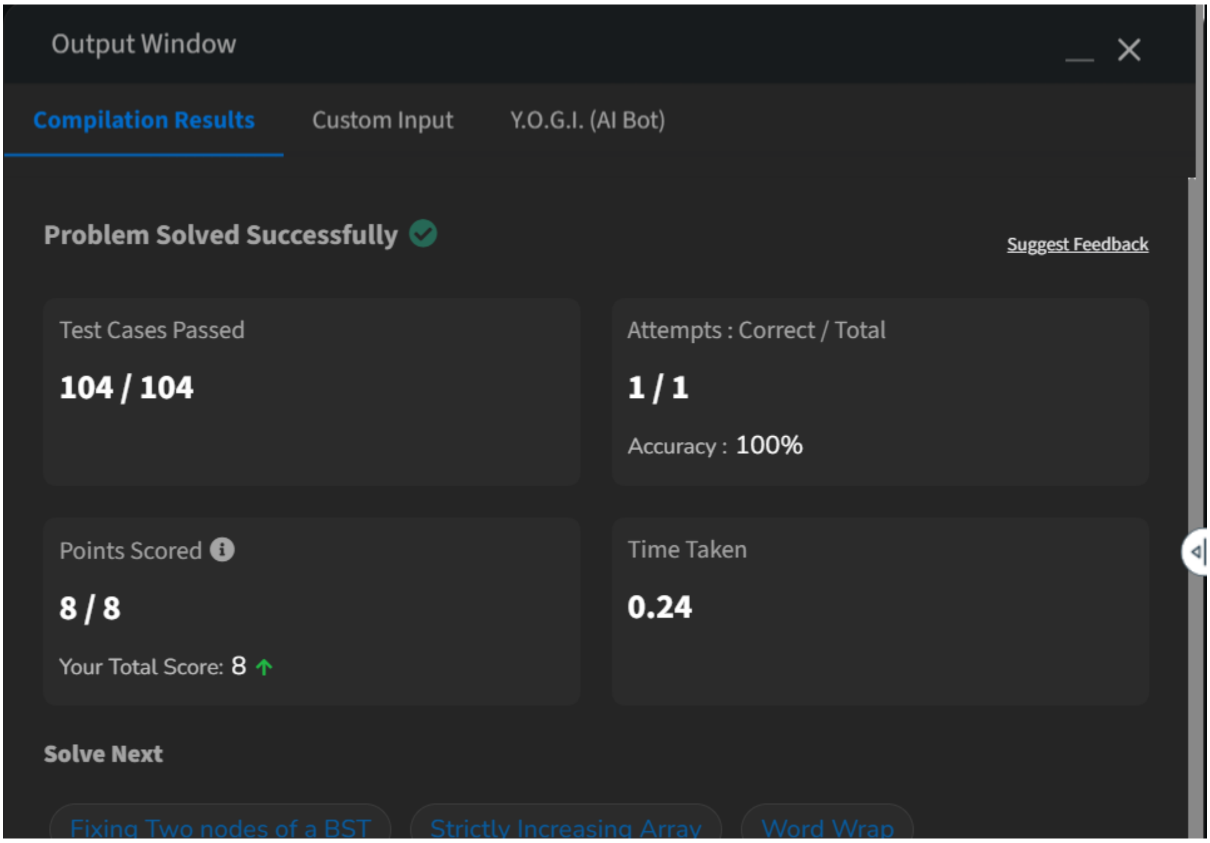
CODE:

```
class Solution {
    static int optimalSearchTree(int keys[], int freq[], int n) {
        int[][] cost = new int[n][n];
        for (int i = 0; i < n; i++)
            cost[i][i] = freq[i];
        for (int l = 2; l <= n; l++) {
            for (int i = 0; i <= n - l; i++) {
                int j = i + l - 1;
                cost[i][j] = Integer.MAX_VALUE;
                int sum = 0;
                for (int k = i; k <= j; k++)
                    sum += freq[k];
                for (int r = i; r <= j; r++) {
                    int c = sum;
                    if (r > i) c += cost[i][r - 1];
                    if (r < j) c += cost[r + 1][j];
                    if (c < cost[i][j]) cost[i][j] = c;
                }
            }
        }
        return cost[0][n - 1];
    }
}
```



```java
class Solution {
    static int optimalSearchTree(int keys[], int freq[], int n) {
        int[][] cost = new int[n][n];
        for (int i = 0; i < n; i++)
            cost[i][i] = freq[i];
        for (int l = 2; l <= n; l++) {
            for (int i = 0; i <= n - l; i++) {
                int j = i + l - 1;
                cost[i][j] = Integer.MAX_VALUE;
                int sum = 0;
                for (int k = i; k <= j; k++)
                    sum += freq[k];
                for (int r = i; r <= j; r++) {
                    int c = sum;
                    if (r > i) c += cost[i][r - 1];
                    if (r < j) c += cost[r + 1][j];
                    if (c < cost[i][j]) cost[i][j] = c;
                }
            }
        }
        return cost[0][n - 1];
    }
}
```

OUTPUT:



Problem Solved Successfully ✓

Test Cases Passed

104 / 104

Attempts : Correct / Total

1 / 1

Accuracy : 100%

Points Scored

8 / 8

Your Total Score: 8 ↑

Time Taken

0.24

Solve Next