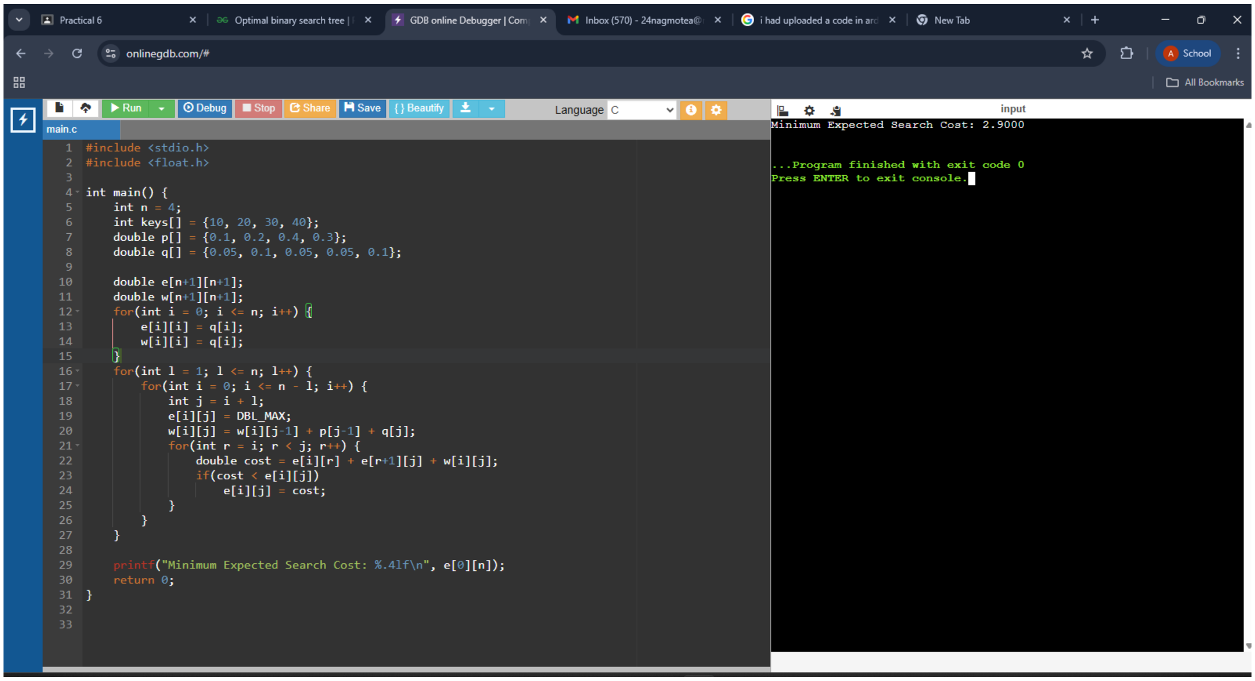
DAA Lab Pract6

Name: Anshika Tiwari

Roll No : A7_B1_14

Aim: Construction of OBST

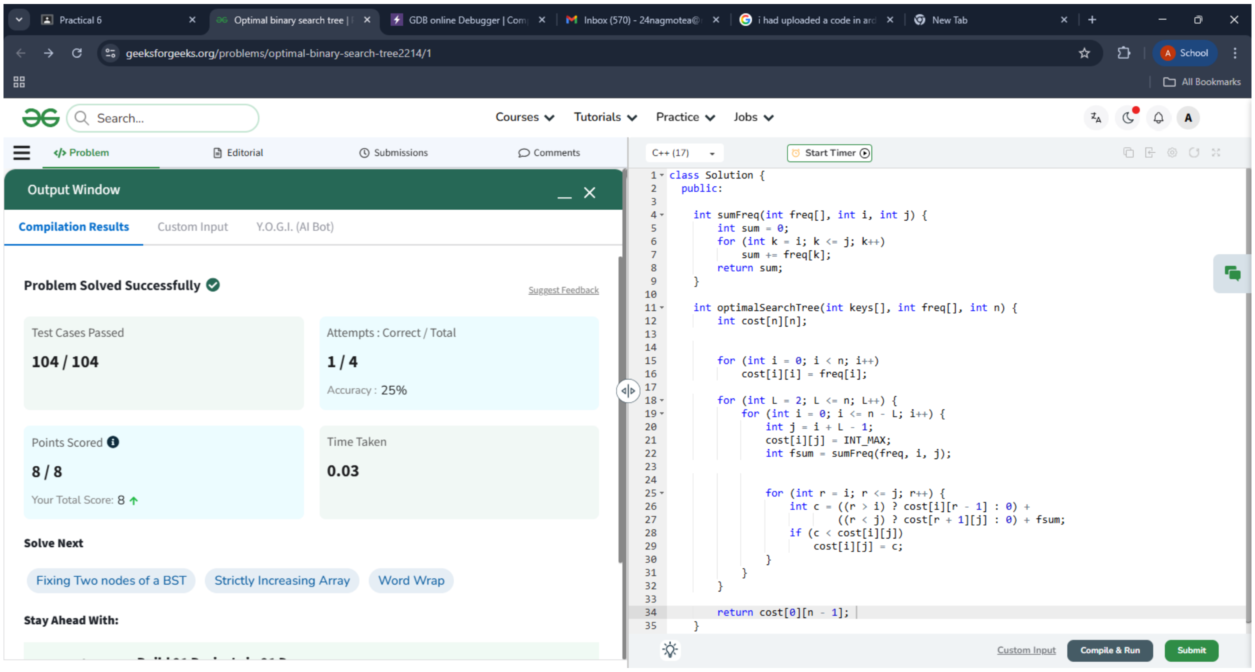
Code & Output:


Practical 6
Optimal binary search tree |
GDB online Debugger | Com
Inbox (570) - 24nagmotea@
i had uploaded a code in arc
New Tab
onlinegdb.com/#
School
All Bookmarks
Run
Debug
Stop
Share
Save
{ } Beautify
Language C
main.c
input

```c
#include <stdio.h>
#include <float.h>

int main() {
    int n = 4;
    int keys[] = {10, 20, 30, 40};
    double p[] = {0.1, 0.2, 0.4, 0.3};
    double q[] = {0.05, 0.1, 0.05, 0.05, 0.1};

    double e[n+1][n+1];
    double w[n+1][n+1];
    for(int i = 0; i <= n; i++) {
        e[i][i] = q[i];
        w[i][i] = q[i];
    }
    for(int l = 1; l <= n; l++) {
        for(int i = 0; i <= n - l; i++) {
            int j = i + l;
            e[i][j] = DBL_MAX;
            w[i][j] = w[i][j-1] + p[j-1] + q[j];
            for(int r = i; r < j; r++) {
                double cost = e[i][r] + e[r+1][j] + w[i][j];
                if(cost < e[i][j])
                    e[i][j] = cost;
            }
        }
    }

    printf("Minimum Expected Search Cost: %.4lf\n", e[0][n]);
    return 0;
}
```

```
Minimum Expected Search Cost: 2.9000


...Program finished with exit code 0
Press ENTER to exit console.
```

geeks of geeks submission:


Practical 6
Optimal binary search tree |
GDB online Debugger | Com
Inbox (570) - 24nagmotea@
i had uploaded a code in ar
New Tab
geeksforgeeks.org/problems/optimal-binary-search-tree2214/1
School
All Bookmarks
Search...
Courses
Tutorials
Practice
Jobs
Problem
Editorial
Submissions
Comments
C++ (17)
Start Timer
Custom Input
Compile & Run
Submit

Output Window

Compilation Results | Custom Input | Y.O.G.I. (AI Bot)

Problem Solved Successfully

Suggest Feedback

Test Cases Passed
104 / 104

Attempts : Correct / Total
1 / 4
Accuracy : 25%

Points Scored
8 / 8
Your Total Score: 8

Time Taken
0.03

Solve Next

Fixing Two nodes of a BST | Strictly Increasing Array | Word Wrap

Stay Ahead With:

```cpp
class Solution {
  public:

    int sumFreq(int freq[], int i, int j) {
        int sum = 0;
        for (int k = i; k <= j; k++)
            sum += freq[k];
        return sum;
    }

    int optimalSearchTree(int keys[], int freq[], int n) {
        int cost[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] = INT_MAX;
                int fsum = sumFreq(freq, i, j);


                for (int r = i; r <= j; r++) {
                    int c = ((r > i) ? cost[i][r - 1] : 0) +
                            ((r < j) ? cost[r + 1][j] : 0) + fsum;
                    if (c < cost[i][j])
                        cost[i][j] = c;
                }
            }
        }

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

Link for my git hub repo: https://github.com/24tiwaria2-code/DAA-