

Practical-6

Name : Shambhavi Pagay

Roll No. : 31

Class : A4/B2

Subject : DAA Lab

Aim: Construction of OBST

Problem Statement: Smart Library Search Optimization

Task 1:

Code:-

```
#include
#define MAX 100
int main() {
    int n;
    printf("Enter number of book IDs (n): ");
    scanf("%d", &n);
    int keys[MAX];
    double p[MAX], q[MAX];
    printf("Enter %d sorted book IDs:\n", n);
    for (int i = 1; i <= n; i++) {
        scanf("%d", &keys[i]);
```

```
}
```

```
printf("Enter %d probabilities of successful  
searches(p[i]):\n",n);
```

```
for (int i = 1; i <= n; i++) {
```

```
scanf("%lf", &p[i]);
```

```
}
```

```
printf("Enter %d probabilities of unsuccessful searches  
(q[i]):\n", n + 1);
```

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

```
scanf("%lf", &q[i]);
```

```
}
```

```
double e[MAX][MAX], w[MAX][MAX];
```

```
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] = 1e9;
```

```
w[i][j] = w[i][j - 1] + p[j] + q[j];
```

```
for (int r = i + 1; r <= j; r++) {
```

```
double t = e[i][r - 1] + e[r][j] + w[i][j];
if (t < e[i][j]) { e[i][j] = t;
}
}
}
}

printf("Minimum expected search cost of OBST: %.4lf\n",
e[0][n]);
return 0;
}
```

Output:-

Output

Clear

```
Enter number of book IDs (n): 4
Enter 4 sorted book IDs:
10
20
30
40
Enter 4 probabilities of successful searches (p[i]):
0.1
0.2
0.4
0.3
Enter 5 probabilities of unsuccessful searches (q[i]):
0.05
0.1
0.05
0.05
0.1
```

Minimum expected search cost of OBST: 2.9000

=== Code Execution Successful ===

TASK-2

Python3 Start Timer

```
1 #User function Template for python3
2
3 class Solution:
4     def optimalSearchTree(self, keys, freq, n):
5         dp = [[0 for _ in range(n)] for _ in range(n)]
6         sum_freq = [[0 for _ in range(n)] for _ in range(n)]
7
8         for i in range(n):
9             sum_freq[i][i] = freq[i]
10            for j in range(i + 1, n):
11                sum_freq[i][j] = sum_freq[i][j - 1] + freq[j]
12
13            for length in range(1, n + 1):
14                for i in range(n - length + 1):
15                    j = i + length - 1
16                    dp[i][j] = float('inf')
17
18                    for r in range(i, j + 1):
19                        cost_left = dp[i][r - 1] if r > i else 0
20                        cost_right = dp[r + 1][j] if r < j else 0
21                        total_cost = cost_left + cost_right + sum_freq[i][j]
22                        dp[i][j] = min(dp[i][j], total_cost)
23
24            return dp[0][n - 1]
25
```

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 / 1

Accuracy : 100%

Points Scored ⓘ

8 / 8

Your Total Score: 8 ↑

Time Taken

2.15