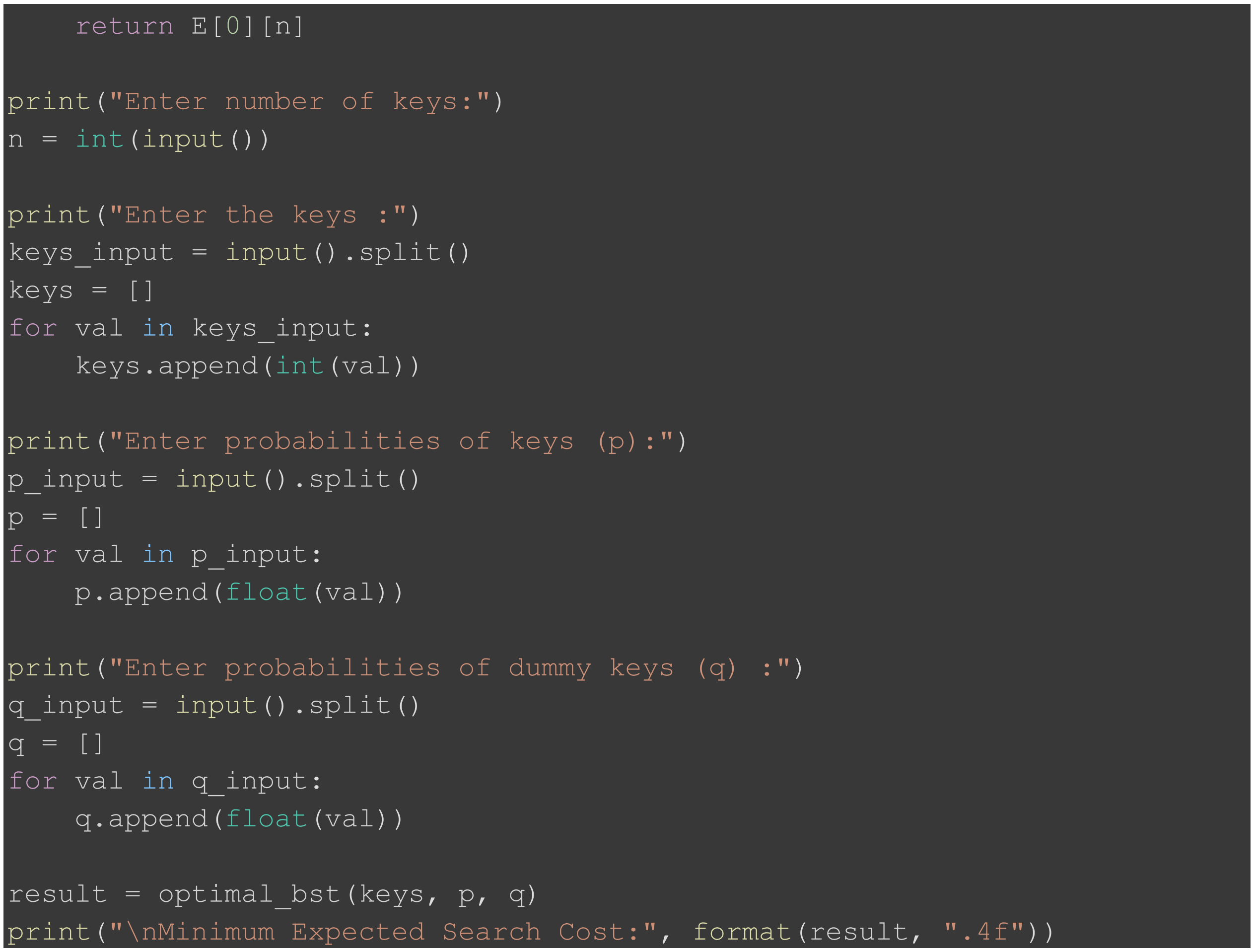
A5-B2-29

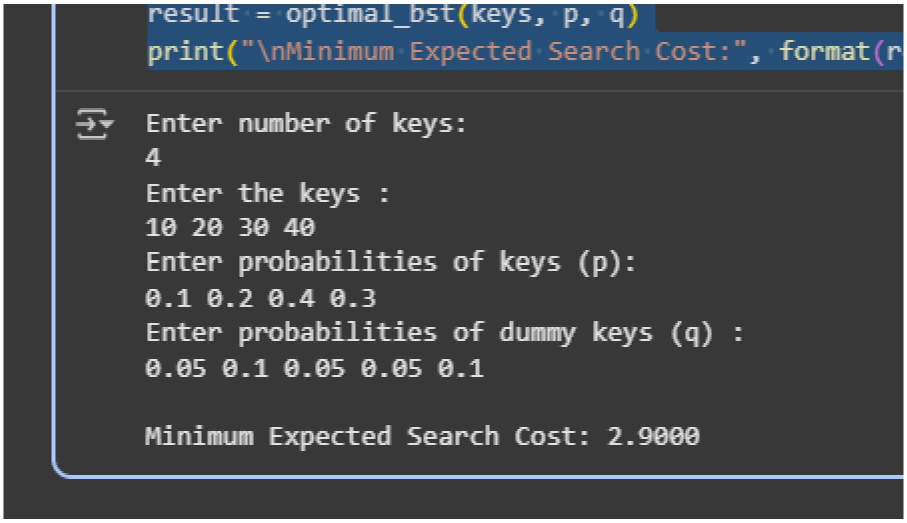
Rishit Saxena

TASK1

```
def optimal_bst(keys, p, q):
    n = len(keys)

    E = []
    for i in range(n + 1):
        row = []
        for j in range(n + 1):
            row.append(0.0)
        E.append(row)

    W = []
    for i in range(n + 1):
        row = []
        for j in range(n + 1):
            row.append(0.0)
        W.append(row)

    R = []
    for i in range(n + 1):
        row = []
        for j in range(n + 1):
            row.append(0)
        R.append(row)

    for i in range(n + 1):
        E[i][i] = q[i]
        W[i][i] = q[i]

    for d in range(1, n + 1):
        for i in range(n - d + 1):
            j = i + d
            E[i][j] = float('inf')
            W[i][j] = W[i][j - 1] + p[j - 1] + q[j]

            for k in range(i + 1, j + 1):
                cost = E[i][k - 1] + E[k][j] + W[i][j]
                if cost < E[i][j]:
                    E[i][j] = cost
                    R[i][j] = k
```

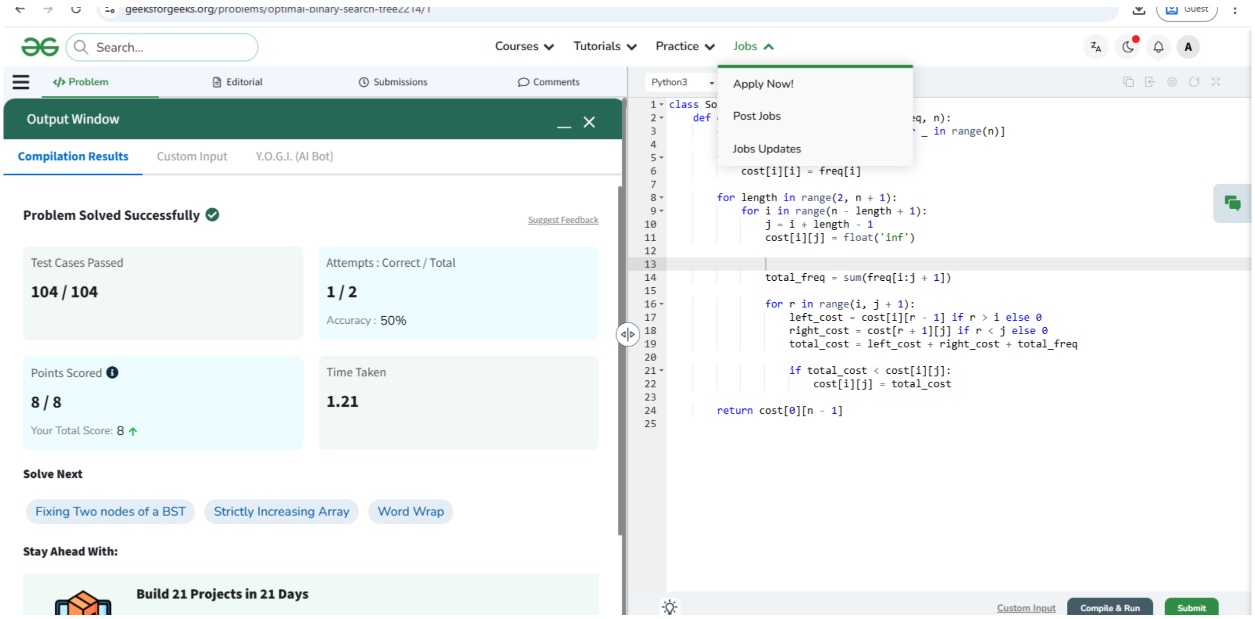


```
    return E[0][n]

print("Enter number of keys:")
n = int(input())

print("Enter the keys :")
keys_input = input().split()
keys = []
for val in keys_input:
    keys.append(int(val))

print("Enter probabilities of keys (p):")
p_input = input().split()
p = []
for val in p_input:
    p.append(float(val))

print("Enter probabilities of dummy keys (q) :")
q_input = input().split()
q = []
for val in q_input:
    q.append(float(val))

result = optimal_bst(keys, p, q)
print("\nMinimum Expected Search Cost:", format(result, ".4f"))
```



```
result = optimal_bst(keys, p, q)
print("\nMinimum Expected Search Cost:", format(r
```

```
Enter number of keys:
4
Enter the keys :
10 20 30 40
Enter probabilities of keys (p):
0.1 0.2 0.4 0.3
Enter probabilities of dummy keys (q) :
0.05 0.1 0.05 0.05 0.1

Minimum Expected Search Cost: 2.9000
```



geeksforgeeks.org/problems/optimal-binary-search-tree2214/1

Output Window

Problem Solved Successfully ✓

Test Cases Passed 104 / 104	Attempts : Correct / Total 1 / 2 Accuracy : 50%
Points Scored 8 / 8 Your Total Score: 8 ↑	Time Taken 1.21

Solve Next

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

Stay Ahead With:

Build 21 Projects in 21 Days

```
class So
    def                                         eq, n):
                                              _ in range(n)]


            cost[i][i] = freq[i]

        for length in range(2, n + 1):
            for i in range(n - length + 1):
                j = i + length - 1
                cost[i][j] = float('inf')


                total_freq = sum(freq[i:j + 1])

                for r in range(i, j + 1):
                    left_cost = cost[i][r - 1] if r > i else 0
                    right_cost = cost[r + 1][j] if r < j else 0
                    total_cost = left_cost + right_cost + total_freq

                    if total_cost < cost[i][j]:
                        cost[i][j] = total_cost

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