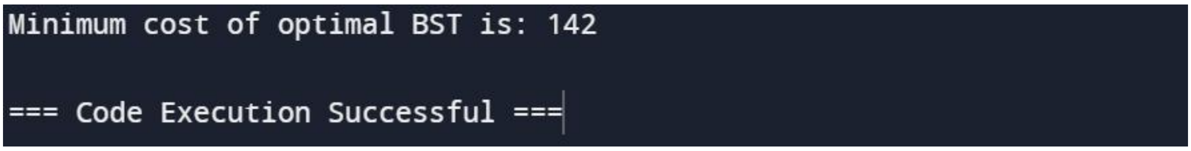
Exersice 109:- Optimal binary search tree

Program:-

```
def optimalBST(keys, freq):
    n = len(keys)
        cost = [[0] * n for _ in range(n)]
    freq_sum = [[0] * n for _ in range(n)]
        for i in range(n):
        cost[i][i] = freq[i]
        freq_sum[i][i] = freq[i]
        for length in range(2, n + 1):  # length of subtree
        for i in range(n - length + 1):
            j = i + length - 1
            cost[i][j] = float('inf')
            freq_sum[i][j] = freq_sum[i][j - 1] + freq[j]
                        for r in range(i, j + 1):
                c = (cost[i][r - 1] if r > i else 0) + (cost[r + 1][j] if r < j else 0) + freq_sum[i][j]
                if c < cost[i][j]:
                    cost[i][j] = c
    return cost[0][n - 1]
keys = [10, 12, 20]
freq = [34, 8, 50]
print("Minimum cost of optimal BST is:", optimalBST(keys, freq))
```

output:-



```
Minimum cost of optimal BST is: 142

=== Code Execution Successful ===
```

Time complexity:-O(n^2)