**Experiment No: 7**

**Name:Shubham Kaspate**

**Roll No:20141241**

**Batch:I3**

**optimal binary search tree using dynamic programming**

#include <bits/stdc++.h>

using namespace std;

int sum(int freq[], int i, int j);

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+1; i++)

{

int j = i+L-1;

cost[i][j] = INT\_MAX;

int off\_set\_sum = sum(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) +

off\_set\_sum;

if (c < cost[i][j])

cost[i][j] = c;

}

}

}

return cost[0][n-1];

}

int sum(int freq[], int i, int j)

{

int s = 0;

for (int k = i; k <= j; k++)

s += freq[k];

return s;

}

int main()

{

int keys[] = {10, 18, 33};

int freq[] = {40, 54, 50};

int n = sizeof(keys)/sizeof(keys[0]);

cout << "Cost of Optimal BST is " << optimalSearchTree(keys, freq, n);

return 0;

}

Output:-

