

Experiment N0. 9

Aim : Given sequence $k = k_1 < k_2 < \dots < k_n$ of n sorted keys, with a search probability p_i for each key k_i . Build the Binary search tree that has the least search cost given the access probability for each key?

Code :

```
#include <bits/stdc++.h>
```

```
using namespace std;
```

```
int sum(int frequency[], int i, int j)
```

```
{  
    int sum = 0;  
    for (int x = i; x <= j; x++)  
        sum += frequency[x];  
    return sum;  
}
```

```
int optimalCost(int frequency[], int i, int j)
```

```
{  
    if (j < i)  
        return 0;  
    if (j == i)  
        return frequency[i];  
  
    int frequencySum = sum(frequency, i, j);  
  
    int min = INT_MAX;  
  
    for (int r = i; r <= j; ++r)  
    {  
        int cost = optimalCost(frequency, i, r - 1) + optimalCost(frequency, r + 1, j);  
        if (cost < min)  
            min = cost;  
    }  
  
    return min + frequencySum;  
}
```

```

int optimalSearchTree(int keys[], int frequency[], int n)
{
    return optimalCost(frequency, 0, n - 1);
}

int main()
{
    int keys[] = {10, 12, 20};
    int frequency[] = {34, 8, 50};

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

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

    return 0;
}

```

Output :

The screenshot displays a C++ IDE with the source code for an optimal BST algorithm. The code defines a function `optimalCost` that uses dynamic programming to find the minimum cost of a BST, and a function `optimalSearchTree` that calls it. The `main` function initializes the keys and frequency arrays and prints the result.

The output window shows the execution results:

```

C:\Users\System21\Documents\prac9thdsal.exe
Cost of Optimal BST is 142
-----
Process exited after 0.05017 seconds with return value 0
Press any key to continue . . .

```