

```

1: #include<iostream>
2: #define MAXN 20
3:
4: using namespace std;
5:
6: void OptimalBST(int Keys[],int p[],int q[],int
  c[][MAXN],int w[][MAXN],int r[][MAXN],int n)
7: {
8:     int l,i,j,k,min,minK,sum;
9:
10:    for(l=1;l<=n+1;l++)
11:    {
12:        for(i=0;i<=(n+1)-l;i++)
13:        {
14:            j = i + l - 1;
15:            if(i==j)
16:            {
17:                w[i][j] = q[i];
18:            }
19:            else
20:            {
21:                w[i][j] = w[i][j-1] + q[j] +
p[j];
22:            }
23:        }
24:    }
25:
26:    for(l=1;l<=n+1;l++)
27:    {
28:        for(i=0;i<=(n+1)-l;i++)

```

```

29:         {
30:             j = i + 1 - 1;
31:             if(i==j)
32:             {
33:                 c[i][j] = r[i][j] = 0;
34:             }
35:             else
36:             {
37:                 min = 99999;
38:                 minK = -1;
39:                 for(k=i+1;k<=j;k++)
40:                 {
41:                     sum = c[i][k-1] + c[k][j] +
w[i][j];
42:                     if(sum<min)
43:                     {
44:                         min = sum;
45:                         minK = k;
46:                     }
47:                 }
48:                 c[i][j] = min;
49:                 r[i][j] = minK;
50:             }
51:         }
52:     }
53: }
54:
55: int main()
56: {
57:     int i,j;

```

```

58:
59:  /*  int Keys[MAXN] = {0, 15, 32, 39, 41, 59};
60:      int p[MAXN] = {0, 5, 2, 6, 3, 5};
61:      int q[MAXN] = {4, 8, 11, 2, 9, 3};
62:      int n = 5;
63:  */
64:  /*  int Keys[MAXN] = {0, 10, 20, 30, 40};
65:      int p[MAXN] = {0, 3, 3, 1, 1};
66:      int q[MAXN] = {2, 3, 1, 1, 1};
67:      int n = 4;
68:  */
69:
70:      int Keys[MAXN] = {0, 10, 20, 30, 40};
71:      int p[MAXN] = {0, 1, 4, 2, 1};
72:      int q[MAXN] = {4, 2, 4, 1, 1};
73:      int n = 4;
74:
75:      int c[MAXN][MAXN];
76:      int w[MAXN][MAXN];
77:      int r[MAXN][MAXN];
78:
79:      float totalProb=0;
80:      for(i=0;i<=n;i++)
81:      {
82:          totalProb += p[i];
83:          totalProb += q[i];
84:      }
85:      OptimalBST(Keys,p,q,c,w,r,n);
86:
87:      //Printing c,w and r table

```

```
88:     cout<<"\n\nMatrix - w\n";
89:     for(i=0;i<=n;i++)
90:     {
91:         for(j=i;j<=n;j++)
92:         {
93:             cout<<"\t"<<w[i][j];
94:         }
95:         cout<<"\n";
96:     }
97:
98:
99:     cout<<"\n\nMatrix - c\n";
100:    for(i=0;i<=n;i++)
101:    {
102:        for(j=i;j<=n;j++)
103:        {
104:            cout<<"\t"<<c[i][j];
105:        }
106:        cout<<"\n";
107:    }
108:
109:    cout<<"\n\nMatrix - r\n";
110:    for(i=0;i<=n;i++)
111:    {
112:        for(j=i;j<=n;j++)
113:        {
114:            cout<<"\t"<<r[i][j];
115:        }
116:        cout<<"\n";
117:    }
```

```
118:
119:     cout<<"\n\nOptimal Cost:
    "<<(c[0][n]/totalProb);
120:
121: }
```