

Exp_14\Kruskal.c

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  #define MAX 30
5
6  typedef struct edge {
7      int u, v, w;
8  } edge;
9
10 typedef struct edge_list {
11     edge data[MAX];
12     int n;
13 } edge_list;
14
15 edge_list elist;
16 int Graph[MAX][MAX], n;
17 edge_list spanlist;
18
19 void kruskal();
20 int find(int belongs[], int vertexno);
21 void union1(int belongs[], int c1, int c2);
22 void sort();
23 void print();
24
25 void kruskal() {
26     int belongs[MAX], i, j, cno1, cno2;
27     elist.n = 0;
28
29     for (i = 1; i < n; i++)
30         for (j = 0; j < i; j++) {
31             if (Graph[i][j] != 0) {
32                 elist.data[elist.n].u = i;
33                 elist.data[elist.n].v = j;
34                 elist.data[elist.n].w = Graph[i][j];
35                 elist.n++;
36             }
37         }
38
39     sort();
40
41     for (i = 0; i < n; i++)
42         belongs[i] = i;
43
44     spanlist.n = 0;
45
46     for (i = 0; i < elist.n; i++) {
47         cno1 = find(belongs, elist.data[i].u);
48         cno2 = find(belongs, elist.data[i].v);
49
50         if (cno1 != cno2) {
51             spanlist.data[spanlist.n] = elist.data[i];
```

```

52         spanlist.n++;
53         union1(belongs, cno1, cno2);
54     }
55 }
56 }
57
58 int find(int belongs[], int vertexno) {
59     return belongs[vertexno];
60 }
61
62 void union1(int belongs[], int c1, int c2) {
63     int i;
64     for (i = 0; i < n; i++)
65         if (belongs[i] == c2)
66             belongs[i] = c1;
67 }
68
69 void sort() {
70     int i, j;
71     edge temp;
72
73     for (i = 1; i < elist.n; i++)
74         for (j = 0; j < elist.n - 1; j++)
75             if (elist.data[j].w > elist.data[j + 1].w) {
76                 temp = elist.data[j];
77                 elist.data[j] = elist.data[j + 1];
78                 elist.data[j + 1] = temp;
79             }
80 }
81
82 void print() {
83     int i, cost = 0;
84     printf("\nMinimum Spanning Tree Edges:\n");
85     for (i = 0; i < spanlist.n; i++) {
86         printf("\nEdge %d: (%d, %d) with weight %d",
87             i + 1, spanlist.data[i].u, spanlist.data[i].v, spanlist.data[i].w);
88         cost += spanlist.data[i].w;
89     }
90     printf("\n\nTotal Cost of Spanning Tree: %d\n", cost);
91 }
92
93 int main() {
94     int i, j, total_cost;
95
96     printf("Enter number of vertices: ");
97     scanf("%d", &n);
98
99     printf("\nEnter the adjacency matrix:\n");
100     for (i = 0; i < n; i++)
101         for (j = 0; j < n; j++)
102             scanf("%d", &Graph[i][j]);
103
104     kruskal();
105     print();

```

```
106 |  
107 |   return 0;  
108 | }
```