

26.Implementation of Minimum Spanning Tree using Kruskal Algorithm

```
#include <stdio.h>

#define MAX 30

int find(int i, int parent[]) {
    while (parent[i])
        i = parent[i];
    return i;
}

int uni(int i, int j, int parent[]) {
    if (i != j) {
        parent[j] = i;
        return 1;
    }
    return 0;
}

int main() {
    int n, i, j, u, v, a, b, min, mincost = 0, ne = 1;
    int cost[MAX][MAX];
    int parent[MAX] = {0};
    printf("Enter number of vertices: ");
    scanf("%d", &n);
    printf("Enter the adjacency matrix (use 0 if no edge):\n");
    for (i = 1; i <= n; i++) {
        for (j = 1; j <= n; j++) {
            scanf("%d", &cost[i][j]);
            if (cost[i][j] == 0)
                cost[i][j] = 999;
        }
    }

    printf("\nEdges in the Minimum Spanning Tree:\n");
```

```

while (ne < n) {

    for (i = 1, min = 999; i <= n; i++) {

        for (j = 1; j <= n; j++) {

            if (cost[i][j] < min) {

                min = cost[i][j];

                a = u = i;

                b = v = j;

            }

        }

    }

    u = find(u, parent);

    v = find(v, parent);

    if (uni(u, v, parent)) {

        printf("Edge %d: (%d - %d) cost: %d\n", ne++, a, b, min);

        mincost += min;

    }

    cost[a][b] = cost[b][a] = 999; // Remove edge

}

printf("\nMinimum Cost = %d\n", mincost);

return 0;

}

```

main.c	Output
<pre> 1 #include <stdio.h> 2 3 #define MAX 30 4 5 int find(int i, int parent[]) { 6 while (parent[i]) 7 i = parent[i]; 8 return i; 9 } 10 11 int uni(int i, int j, int parent[]) { 12 if (i != j) { 13 parent[j] = i; 14 return 1; 15 } 16 return 0; 17 } 18 19 int main() { 20 int n, i, j, u, v, a, b, min, mincost = 0, ne = 1; 21 int cost[MAX][MAX]; 22 int parent[MAX] = {0}; 23 24 printf("Enter number of vertices: "); 25 scanf("%d", &n); 26 27 printf("Enter the adjacency matrix (use 0 if no edge):\n"); 28 for (i = 1; i <= n; i++) { 29 for (j = 1; j <= n; j++) { </pre>	<pre> Enter number of vertices: 5 Enter the adjacency matrix (use 0 if no edge): 1 2 3 4 5 5 6 7 8 9 9 8 7 4 5 6 1 2 3 5 4 5 6 1 2 Edges in the Minimum Spanning Tree: Edge 1: (4 - 2) cost: 1 Edge 2: (5 - 4) cost: 1 Edge 3: (1 - 2) cost: 2 Edge 4: (4 - 3) cost: 2 Minimum Cost = 6 === Code Execution Successful === </pre>