

Design and implement C/C++ Program to find Minimum Cost Spanning Tree of a given connected undirected graph using Kruskal's algorithm.

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
#define INF 999
```

```
int parent[100], cost[100][100];
```

```
int findParent(int i) {  
    while (parent[i] != 0) {  
        i = parent[i];  
    }  
    return i;  
}
```

```
int unionVertices(int i, int j) {  
    if (i != j) {  
        parent[j] = i;  
    }  
}
```

```

        return 1;
    }
    return 0;
}

int main() {
    int i, j, n, min, ne = 1;
    int u = 0, v = 0, a = 0, b = 0, mincost = 0;

    printf("Enter the number of vertices/nodes in the graph\n");
    scanf("%d", &n);

    printf("Enter the Cost/Weight matrix\n");
    for (i = 1; i <= n; i++) {
        parent[i] = 0;
        for (j = 1; j <= n; j++) {
            scanf("%d", &cost[i][j]);
            if (cost[i][j] == 0) {
                cost[i][j] = INF;
            }
        }
    }
}

```

```
}  
}
```

```
printf("The edges of Minimum spanning tree are:\n");
```

```
while (ne < n) {
```

```
    min = INF;
```

```
    for (i = 1; 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 = findParent(u);
```

```
    v = findParent(v);
```

```
    if (unionVertices(u, v)) {
```

```
        printf("%d Edge Selected (%d --- %d) Cost = %d\n",
ne++, a, b, min);

        mincost += min;

    }

    cost[a][b] = cost[b][a] = INF;

}

printf("Minimum cost = %d\n", mincost);

return 0;

}
```