8) Find Minimum Cost Spanning Tree of a given undirected graph using Kruskal's algorithm.

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

int i, j, k, a, b, u, v, n, ne = 1;
int min, mincost = 0, cost[9][9], parent[9];

int find(int);
int uni(int, int);

void main() {
   printf("\n\n\tImplementation of Kruskal's algorithm\n\n");
   printf("\nEnter the number of vertices\n");
   scanf("%d", &n);

   printf("\nEnter the cost adjacency matrix\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("\nThe edges of Minimum Cost Spanning Tree are\n\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);
      v = find(v);
      if (uni(u, v)) {
         printf("\n%d edge (%d, %d) = %d\n", ne++, a, b, min);
         mincost += min;
      }
      cost[a][b] = cost[b][a] = 999;
   }
   printf("\n\tMinimum cost = %d\n", mincost);
}

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

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

```
        Implementation of Kruskal's algorithm


Enter the no. of vertices
6

Enter the cost adjacency matrix
0 2 3 4 5 6
2 0 1 1 2 4
3 1 0 0 7 0
4 1 0 0 0 2
5 2 7 0 0 1
6 4 0 2 1 0

The edges of Minimum Cost Spanning Tree are


1 edge (2,3) =1

2 edge (2,4) =1

3 edge (5,6) =1

4 edge (1,2) =2

5 edge (2,5) =2

        Minimum cost = 7
```

9) Find Minimum Cost Spanning Tree of a given undirected graph using Prim's algorithm

```
#include <stdio.h>

int a, b, u, v, n, i, j, ne = 1;
int visited[10] = {0}, min, mincost = 0, cost[10][10];

void main() {
  printf("\n Enter the number of nodes:");
  scanf("%d", &n);
  printf("\n Enter the adjacency matrix:\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;
    }

  visited[1] = 1;
  printf("\n");

  while (ne < n) {
    for (i = 1, min = 999; i <= n; i++)
      for (j = 1; j <= n; j++)
        if (cost[i][j] < min)
          if (visited[i] != 0) {
            min = cost[i][j];
            a = u = i;
            b = v = j;
          }

    if (visited[u] == 0 || visited[v] == 0) {
      printf("\n Edge %d: (%d %d) cost: %d", ne++, a, b, min);
      mincost += min;
      visited[b] = 1;
    }
```

```
    cost[a][b] = cost[b][a] = 999;
  }

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

```
 Enter the number of nodes:6

 Enter the adjacency matrix:
0 2 3 4 5 6
2 0 1 1 2 4
3 1 0 0 7 0
4 1 0 0 0 2
5 2 7 0 0  1
6 4 0 2 1 0


 Edge 1:(1 2) cost:2
 Edge 2:(2 3) cost:1
 Edge 3:(2 4) cost:1
 Edge 4:(2 5) cost:2
 Edge 5:(5 6) cost:1
 Minimun cost=7
```