4. Kruskal algorithm

```
#include <stdio.h>

int matrix[10][10], vertices, edges[10][2], totalWeight;

void computeKruskal(int matrix[10][10], int vertices);
int findRoot(int sets[10], int node);

int main() {
    int i, j;

    printf("Enter the number of vertices: ");
    scanf("%d", &vertices);

    printf("Enter the cost adjacency matrix:\n");
    for (i = 0; i < vertices; i++) {
        for (j = 0; j < vertices; j++) {
            scanf("%d", &matrix[i][j]);
        }
    }

    computeKruskal(matrix, vertices);

    printf("Edges of the minimal spanning tree:\n");
    for (i = 0; i < vertices - 1; i++) {
        printf("(%d, %d) ", edges[i][0], edges[i][1]);
    }

    printf("\nSum of minimal spanning tree: %d\n", totalWeight);

    return 0;
}

void computeKruskal(int matrix[10][10], int vertices) {
    int set[10], included = 0, i, j;
    int u, v, small, edgeIdx = 0;

    for (i = 0; i < vertices; i++) {
        set[i] = i;
    }

    totalWeight = 0;
    while (included < vertices - 1) {
        small = 999;
        u = -1;
        v = -1;

        for (i = 0; i < vertices; i++) {
            for (j = 0; j < vertices; j++) {
                if (findRoot(set, i) != findRoot(set, j) && matrix[i][j] < small) {
                    small = matrix[i][j];
                    u = i;
                    v = j;
                }
            }
        }

        if (u != -1 && v != -1) {
            int rootU = findRoot(set, u);
            int rootV = findRoot(set, v);

            if (rootU != rootV) {
                set[rootU] = rootV;
                edges[edgeIdx][0] = u;
                edges[edgeIdx][1] = v;
                totalWeight += small;
                edgeIdx++;
                included++;
            }
        }
    }
}

int findRoot(int sets[10], int node) {
    while (sets[node] != node) {
        node = sets[node];
    }
    return node;
}
```

```
Output
Enter the number of vertices: 4
Enter the cost adjacency matrix:
0 5 8 0
5 0 10 15
```

```
8 10 0 20
0 15 20 0
Edges of the minimal spanning tree:
(0, 1) (0, 2) (1, 3)
Sum of minimal spanning tree: 30
```