25. Kruskal’s Algorithm (Minimum Spanning Tree)

Aim:

To find MST using Kruskal’s algorithm.

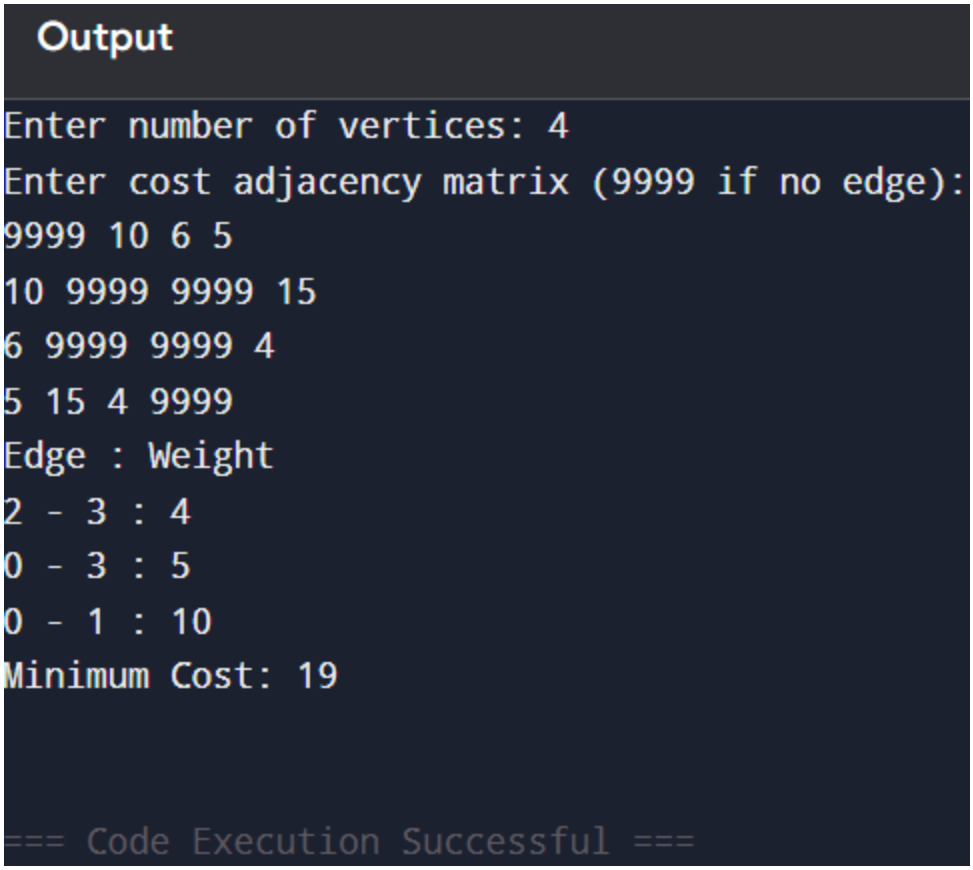
Algorithm:

1. Sort all edges by weight.
2. Pick the smallest edge that does not form a cycle (using Union-Find).
3. Repeat until MST has (V-1) edges.

CODE:

```
#include <stdio.h>
#define MAX 20

int parent[MAX];

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

void unionSet(int i, int j) {
    int a = find(i);
    int b = find(j);
    parent[a] = b;
}

void kruskal(int n, int cost[MAX][MAX]) {
    int mincost = 0, edge_count = 0;
    for (int i = 0; i < n; i++) parent[i] = i;

    printf("Edge : Weight\n");
```

```
    while (edge_count < n - 1) {
        int min = 9999, a = -1, b = -1;
        for (int i = 0; i < n; i++) {
            for (int j = 0; j < n; j++) {
                if (find(i) != find(j) && cost[i][j] < min) {
                    min = cost[i][j];
                    a = i;
                    b = j;
                }
            }
        }
        unionSet(a, b);
        printf("%d - %d : %d\n", a, b, min);
        mincost += min;
        edge_count++;
    }
    printf("Minimum Cost: %d\n", mincost);
}

int main() {
    int n, cost[MAX][MAX];
    printf("Enter number of vertices: ");
    scanf("%d", &n);
    printf("Enter cost adjacency matrix (9999 if no edge):\n");
    for (int i = 0; i < n; i++)
        for (int j = 0; j < n; j++)
            scanf("%d", &cost[i][j]);

    kruskal(n, cost);
    return 0;
}
```



RESULT:

The program successfully executed and displayed the Kruskal's Algorithm.