13. Minimum Spanning Tree using Prim's Algorithm

Program :

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

#define MAX_VERTICES 100
int minKey(int key[], int mstSet[], int vertices) {
    int min = INT_MAX, minIndex;

    for (int v = 0; v < vertices; v++) {
        if (!mstSet[v] && key[v] < min) {
            min = key[v];
            minIndex = v;
        }
    }

    return minIndex;
}
void printMST(int parent[], int graph[MAX_VERTICES][MAX_VERTICES], int
vertices) {
    printf("Edge \tWeight\n");
    for (int i = 1; i < vertices; i++) {
        printf("%d - %d \t%d\n", parent[i], i, graph[i][parent[i]]);
    }
}
void primMST(int graph[MAX_VERTICES][MAX_VERTICES], int vertices) {
    int parent[MAX_VERTICES];
    int key[MAX_VERTICES];
    int mstSet[MAX_VERTICES];
    for (int i = 0; i < vertices; i++) {
        key[i] = INT_MAX;
        mstSet[i] = 0;
    }
    key[0] = 0;
    parent[0] = -1;
    for (int count = 0; count < vertices - 1; count++) {

        int u = minKey(key, mstSet, vertices);
        mstSet[u] = 1;
        for (int v = 0; v < vertices; v++) {
            if (graph[u][v] && !mstSet[v] && graph[u][v] < key[v]) {
                parent[v] = u;
                key[v] = graph[u][v];
            }
        }
    }
    printMST(parent, graph, vertices);
}

int main() {
    int vertices;
    printf("Input the number of vertices: ");
    scanf("%d", &vertices);
    if (vertices <= 0 || vertices > MAX_VERTICES) {
        printf("Invalid number of vertices. Exiting...\n");
        return 1;
    }

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

    return 0;
}
```

Output :

```
Input the number of vertices: 3
Input the adjacency matrix for the graph:
1 4 3
6 3 0
9 7 0
Edge  Weight
0 - 1      6
0 - 2      9
```