Practical no: 3

Name: Saloni Vishwakarma
Batch-Roll no: C1-13
Subject: DAA Lab
Date of execution: 9 June 2023

Aim: Simplifying Business travel Agency aim is to provide the best path which takes less time to travel all the cities which visitor wants to visit. No of cities Ci and the travel time Ti between cities is given. Suggest the best suited greedy algorithm to find the travel time which takes less time, implement and provide the solution.

Prim's Algorithm (Code and Output):

```
#include <stdio.h>
#include <stdbool.h>
#define MAX_VERTICES 20
int minKey(int key[], bool mstSet[], int vertices) {
   int min = __INT_MAX__, min_index;

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

void printMST(int parent[], int graph[MAX_VERTICES][MAX_VERTICES], int vertices) {
   printf("\n Minimum Spanning Tree:\n");
   printf("  Edge \t  Weight\n");
   int minCost = 0;
   for (int i = 1; i < vertices; i++) {
      printf("  %d - %d     %d\n", parent[i], i, graph[i][parent[i]]);
      minCost += graph[i][parent[i]];
   }
```

```
    printf("\n Minimum Cost: %d\n", minCost);
}

void primMST(int graph[MAX_VERTICES][MAX_VERTICES], int vertices) {
    int parent[MAX_VERTICES];
    int key[MAX_VERTICES];
    bool mstSet[MAX_VERTICES];
    for (int i = 0; i < vertices; i++) {
        key[i] = __INT_MAX__;
        mstSet[i] = false;
    }
    key[0] = 0;
    parent[0] = -1;
    for (int count = 0; count < vertices - 1; count++) {
        int u = minKey(key, mstSet, vertices);
        mstSet[u] = true;
        for (int v = 0; v < vertices; v++) {
            if (graph[u][v] && mstSet[v] == false && graph[u][v] < key[v]) {
                parent[v] = u;
                key[v] = graph[u][v];
            }
        }
    }
    printMST(parent, graph, vertices);
}

int main() {
    int vertices;
    int graph[MAX_VERTICES][MAX_VERTICES];

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

    printf("\n Enter the adjacency matrix:\n");
    for (int i = 0; i < vertices; i++) {
        for (int j = 0; j < vertices; j++) {
            if(j==0)
            printf(" ");
```

```
            scanf("%d", &graph[i][j]);
        }
    }
    primMST(graph, vertices);
    return 0;
}
```

```
Enter the number of vertices: 6

Enter the adjacency matrix:
0 4 0 0 0 8
4 0 20 0 0 16
0 20 0 10 6 5
0 0 10 0 25 0
0 0 6 25 0 7
8 16 5 0 7 0

Minimum Spanning Tree:
 Edge    Weight
 0 - 1     4
 5 - 2     5
 2 - 3     10
 2 - 4     6
 0 - 5     8

Minimum Cost: 33


...Program finished with exit code 0
Press ENTER to exit console.
```

Conclusion: We have successfully studied and implemented MST (Minimum spanning tree) i.e. Prim's algorithm using C.