22AIE203 – Data Structures and Algorithm - 2

LAB EXP 2

Prim’s algorithm

NAME :GURUPRASATH M R

ROLLNO:CH.EN.U4.AIE22015

CLASS:2 YEAR 3 SEM

Code

#include <stdio.h>

#include <stdbool.h>

#include <limits.h>

#define V 4 // Number of vertices in the graph

// Function to find the vertex with the minimum key value

int findMinKey(int key[], bool Set[]) {

    int min = INT\_MAX, min\_index;

    for (int v = 0; v < V; v++) {

        if (Set[v] == false && key[v] < min) {

            min = key[v];

            min\_index = v;

        }

    }

    return min\_index;

}

// Function to print the prim

void print(int parent[], int graph[V][V]) {

    printf("Edge \tWeight\n");

    for (int i = 1; i < V; i++) {

        printf("%d - %d \t%d\n", parent[i], i, graph[i][parent[i]]);

    }

}

// Function to construct and print the  using Prim's algorithm

void prim(int graph[V][V]) {

    int parent[V]; // Array to store constructed

    int key[V];    // Key values used to pick the minimum weight edge

    bool Set[V]; // To represent the set of vertices included in

    // Initialize all key values as infinite and Set[] as false

    for (int i = 0; i < V; i++) {

        key[i] = INT\_MAX;

        Set[i] = false;

    }

    // Start with the first vertex

    key[0] = 0;

    parent[0] = -1; // First node is always the root of

    //  will have V-1 edges

    for (int count = 0; count < V - 1; count++) {

        int u = findMinKey(key, Set);

        // Add the picked vertex to the  Set

        Set[u] = true;

        // Update key and parent index of the adjacent vertices

        for (int v = 0; v < V; v++) {

            if (graph[u][v] && Set[v] == false && graph[u][v] < key[v]) {

                parent[v] = u;

                key[v] = graph[u][v];

            }

        }

    }

    // Print the constructed

    print(parent, graph);

}

int main() {

    int graph[V][V] = {

        {0, 1, 3, 0},

        {1, 0, 2, 2},

        {3, 2, 0, 0},

        {0, 2, 0, 0}

    };

    // Print the  using Prim's algorithm

    prim(graph);

    return 0;

}

Output

